



ORACLE
NETSUITE

NetSuite Connector

2022.2

March 22, 2023



Copyright © 2005, 2022, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document may change and remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

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 <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided "as is" and "as available", for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer's use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle's other customers, Oracle may slow down or throttle Customer's excess use until such time that Customer's use stays within reasonable limits. If Customer's particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer's actual usage pattern.

Beta Features

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described in this document remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Send Us Your Feedback

We'd like to hear your feedback on this document.

Answering the following questions will help us improve our help content:

- Did you find the information you needed? If not, what was missing?
- Did you find any errors?
- Is the information clear?
- Are the examples correct?
- Do you need more examples?
- What did you like most about this document?

Click [here](#) to send us your comments. If possible, please provide a page number or section title to identify the content you're describing.

To report software issues, contact NetSuite Customer Support.

Table of Contents

NetSuite Connector Overview	1
NetSuite Connector Business Flow	1
NetSuite Connector Data Management	2
NetSuite Connector Communication with Internal IDs and Field IDs	3
Data Manipulation in NetSuite and NetSuite Connector	3
NetSuite Connector Customer Record Handling	4
NetSuite Connector Items Handling	4
NetSuite Connector Concurrency and Data Limits	5
NetSuite Connector Tax Handling	6
Gift Cards and Gift Certificates in NetSuite Connector	7
Concession and Refund Handling in NetSuite Connector	7
NetSuite Connector Carrier Handling	7
Types of Shipping Cost Values in NetSuite Connector	8
TLS Protocol Support in NetSuite Connector	8
Compliance and Fraud Prevention in NetSuite Connector	8
NetSuite Connector User Interface	8
User Event Scripts in NetSuite Connector	9
NetSuite Connector Authentication	10
NetSuite Connector Syncs	10
NetSuite Connector Product Sync	10
NetSuite Connector Order Sync	22
NetSuite Connector Refund Sync	24
NetSuite Connector Third-Party Logistics (3PL) Sync	24
Best Practices for NetSuite Connector Syncs	25
Sync Frequency and Directional Flows	33
Number of Logic Levels Supported in NetSuite Connector Mappings	33
NetSuite Connector Third Party Licensing	34
NetSuite Connector Setup	35
NetSuite Connector Prerequisites	35
NetSuite Connector IP Addresses to Safelist (Allowlist)	36
Creating a NetSuite Connector Account	37
Adding a User to a NetSuite Connector Account	38
Changing the NetSuite Connector Password in the Account Settings	38
Resetting the NetSuite Connector Password	39
Notification and Email Settings for NetSuite Connector	39
Setting Up Token-Based Authentication for NetSuite Connector	40
Migrating to the NetSuite Connector SuiteApp	41
Setting Up NetSuite Connector Order Sync	51
Setting Up NetSuite Connector Product Sync	52
Setting Up NetSuite for NetSuite Connector Syncs	54
Setting Up NetSuite Connector for Syncs	54
Managing Connector and Account Settings in NetSuite Connector	56
Accessing Connector and Account Settings in NetSuite Connector	56
Adding a Connector to a NetSuite Connector Account	57
Updating the Connector Account Nickname in NetSuite Connector	57
Disabling or Removing a Connector from NetSuite Connector	58
NetSuite Connector Sync Tests	59
NetSuite Connector Order and Fulfillment Sync Tests	59
NetSuite Connector Test Orders Prerequisites	59
Creating Test Orders in Marketplace or Cart	60
Testing NetSuite Connector Order Sync	60
Testing NetSuite Connector Fulfillment Sync	61
Activating NetSuite Connector Order and Fulfillment Sync	62

NetSuite Connector Product Sync Tests	62
NetSuite Connector Test Products Prerequisites	63
Mapping NetSuite Connector Test Products	64
Mapping Inventory and Price for NetSuite Connector Test Products	65
Testing NetSuite Connector Product Sync	65
Activating NetSuite Connector Product Sync	66
NetSuite Connector Refund Sync Tests	66
NetSuite Connector Test Refund Prerequisites	66
Testing NetSuite Connector Refund Sync to NetSuite	67
Activating NetSuite Connector Refund Sync	68
NetSuite Connector Third-Party Logistics (3PL) Sync Tests	69
NetSuite Connector 3PL Sync Test Prerequisites	69
Testing NetSuite Connector ShipStation Sync	69
NetSuite Connector Settlement Sync Tests	71
NetSuite Connector Settlement Sync Prerequisites	71
Testing NetSuite Connector Settlement Sync	72
Activating NetSuite Connector Settlement Sync	73
NetSuite Connector Manual Sync Tests	73
NetSuite Connector Deployment to Production	77
Checking Syncs for NetSuite Connector Deployment	77
Checking Accounts for NetSuite Connector Deployment	79
Switching to NetSuite Connector from an Existing Integration	80
Deploying NetSuite Connector to Production	80
NetSuite Connector Deployment to Production FAQ	80
Integrating NetSuite Connector with NetSuite	82
Verifying a NetSuite Connector Integration Record	82
Working with NetSuite Records in NetSuite Connector	83
Viewing Records and Mappings In NetSuite Connector	84
Splitting an Order into Multiple Item Fulfillments Using NetSuite	85
Changing the Primary User Information in NetSuite Connector	85
Exposing Internal IDs and Field IDs	85
Finding the Internal ID of Records	86
Viewing Internal IDs in NetSuite	87
Setting Form Fields Visible Or Editable for NetSuite Connector	87
Showing the Fulfilled Field on Sales Orders for NetSuite Connector	87
NetSuite SKU and Storefront SKU in NetSuite Connector	88
Setting the SKU Fields in NetSuite Connector	88
Verifying NetSuite SKU Matches the Storefront SKU	89
Managing Fields for NetSuite Connector	90
Adding NetSuite Custom Fields	90
Making Custom Fields in NetSuite Available for Global Search	90
Enabling Order Email Messages in NetSuite Connector	90
Handling Inventory Items with Alias SKU	91
Viewing Inventory History in NetSuite	92
Enabling Quantity-Based Pricing in NetSuite	92
Using NetSuite Connector for Data Migration	93
Configuring VAT in NetSuite Connector	93
Handling Incorrect Tax Rates, Tax Codes, or Tax Items in NetSuite Orders	94
Creating or Reopening an Accounting Period in NetSuite	95
Finding the Cost of Goods Sold (COGS) Account on Items	95
NetSuite Connector Saved Search Export	95
Undeploying Scripts Installed by NetSuite Connector	99
Troubleshooting Common NetSuite Issues in NetSuite Connector	99
Troubleshooting Common NetSuite Configuration Issues	99
Troubleshooting Fields Not Being Visible or Editable Errors	100

Troubleshooting Last Name is Missing Errors	100
Troubleshooting Character Limit Errors in NetSuite	101
Troubleshooting Issues When Using Avalara or Scripts on Sales Orders	102
Troubleshooting Connection Time Out Issues	102
Troubleshooting Sync Script Errors	103
Integrating NetSuite Connector with Supported Storefronts or 3PLs	104
Integrating NetSuite Connector with Amazon	104
Integrating NetSuite Connector with eBay	104
Integrating NetSuite Connector with Magento 2	105
Integrating NetSuite Connector with Shopify	105
NetSuite Connector Sync Management	107
Managing Sync Operations in NetSuite Connector	107
Disabling a Mapping in NetSuite Connector	109
Order and Product Sync Error Messages	109
NetSuite Connector Order Sync Management	110
Order Mappings in NetSuite Connector	111
Mapping Order Location in NetSuite Connector	116
Setting up Custom Order Mappings in NetSuite Connector	117
Viewing Order Mapping in NetSuite Connector	126
Setting Up Customer Deposits	129
Mapping a Subsidiary on Orders in NetSuite Connector	131
Setting Up Order Sync to Handle Multiple Subsidiaries in NetSuite	131
Appending a Set Value to Order Numbers in NetSuite Connector	132
Reloading List Values for Order Mappings in NetSuite Connector	133
Assigning a Fixed Customer to All Orders in NetSuite Connector	133
Selecting the First Available Lot Number on Items During Order Sync in NetSuite Connector	134
Managing Order Discounts in NetSuite Connector	135
Displaying a Discount Item or Rate Field on the Order Form for NetSuite Connector	136
Handling Incorrect Tax Totals on Discounted Orders	137
Ensuring That NetSuite and Storefront Order Totals Are Matching	137
Posting an Order with Total Variance into NetSuite	138
Configuring Accounting for NetSuite Connector Order Discounts	138
Changing the User That is Assigned the Role for Token Authentication	139
Writing to Custom Segment Fields on Orders in NetSuite Connector	139
Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite	140
Checking if an Order was Imported by NetSuite Connector	141
Finding Order Data that NetSuite Connector Sent To NetSuite	142
Finding an Available Order Data to Map in NetSuite Connector	142
Editing an Order in NetSuite Connector	143
Deleting and Canceling Orders in NetSuite Connector	143
Enabling Order Cancellations	144
Posting a Canceled Order	145
Using Batch Operations in the NetSuite Connector Order Portal	145
Changing Synced Orders	146
Identifying the NetSuite Form that NetSuite Connector Uses for Order Sync	146
Changing the Preferred Order Form for NetSuite Connector	147
Changing the Script Execution Order in NetSuite Connector When TaxJar is Enabled	147
Setting Order Filters in NetSuite Connector	148
Checking for Item Filters in an Order Form Used by NetSuite Connector	150
Setting Order Cutoff Dates in NetSuite Connector	150
Handling a High Volume of Orders	151
Importing Orders from the Storefront in NetSuite Connector	152
Setting Order Statuses for Automatic Import in NetSuite Connector	152
Delaying Order Imports in NetSuite Connector	153
Searching for an Order in NetSuite Connector	153

Searching for Orders Posted By NetSuite Connector	154
Setting Customers As Taxable By Default	154
Handling Line Level Tax Rates in NetSuite Connector	154
Configuring Value-Added Tax (VAT) in NetSuite Connector	155
Omitting Remitted Tax in NetSuite Connector	155
Assigning Items to Tax Groups When Syncing Orders in NetSuite Connector	156
Configuring Gift Cards for NetSuite Connector	156
Mapping the Shipment or Payment Methods in NetSuite Connector	157
Handling Invalid Shipping Tax Code Reference Key in NetSuite Connector	164
Fixing Incorrect Shipping Tax on the Order in NetSuite	165
Configuring Shipping for NetSuite Connector	165
Posting an Order After Resolving the Error on the Order	166
Exporting Order Sync Errors from NetSuite Connector	166
Troubleshooting Order or Fulfillment Sync Errors in NetSuite Connector	167
NetSuite Connector Order Sync FAQ	179
NetSuite Connector Refund Sync Management	184
Refunds and Exchanges in NetSuite Connector	184
Syncing Refund for Orders Deleted from NetSuite	185
Stopping Cash Refunds from Restocking Refunded Items	186
Deleting or Canceling Refunds in NetSuite Connector	187
NetSuite Connector Refund Sync FAQ	189
NetSuite Connector Fulfillment Sync Management	190
NetSuite Connector Fulfillment Sync Triggers	190
Creating Mappings For Fulfillment Sync in NetSuite Connector	190
Enabling Multiple Item Fulfillments from a Single Order in NetSuite Connector	191
Pushing Fulfillments Manually from NetSuite in NetSuite Connector	192
Configuring Custom Fields for Fulfillments	192
Troubleshooting Common Fulfillment Sync Errors in NetSuite Connector	192
NetSuite Connector Product Sync Management	193
Setting Up Full Product Sync Mapping	194
Setting the Real-Time Price and Quantity Sync Preference	195
Disabling Real-Time Sync	195
Creating a Flag Field for Product Sync	196
Flagging Existing NetSuite Items to Sync	196
Creating a List-Based Storefront Flag Field Manually	197
Creating a Single-Character Storefront Flag Field Manually	198
Mapping Product or Inventory Sync in NetSuite Connector	199
Viewing Product Mapping in NetSuite Connector	204
Mapping Required and Optional Fields for Product Sync	205
Mapping Cart Categories in NetSuite Connector	206
Mapping Inventories in NetSuite Connector	207
Mapping Custom Field Inventories in NetSuite Connector	207
Blank Value Field for Category Product Mapping in NetSuite Connector	208
Removing or Resetting Data for a Field in the Storefront Using Product Sync	208
Updating the SKU of Items Posted with NetSuite Connector	209
Fixing SKUs that Do Not Match the Storefront SKUs	209
Setting Maximum Quantity for Inventory During Product Sync	210
Setting Minimum Quantity for Inventory During Product Sync	210
Adding Product Images for NetSuite Connector	211
Configuring Assembly Item Quantities in NetSuite Connector	213
Batch Updating Items in NetSuite	214
Searching for a Product in NetSuite Connector	214
Reloading Products in NetSuite Connector	215
Removing an Item from a Marketplace or Cart	216
Deleting a Variation Option in NetSuite Connector	216

Mapping Prices in NetSuite Connector	216
Configuring Price Calculations Before Sync in NetSuite Connector	219
Accommodating Repricer in NetSuite Connector Product Sync	220
Configuring Quantity Calculations Before Sync in NetSuite Connector	220
Subtracting Backordered Quantities for Multiple Locations in NetSuite Connector	221
Information for Troubleshooting Inventory Sync Issues in NetSuite Connector	222
Troubleshooting Product Sync Issues in NetSuite Connector	222
NetSuite Connector Product Sync FAQ	227
Managing Connectors	230
Managing the Amazon Connector in NetSuite Connector	230
Amazon Seller Central vs. Vendor Central	231
Supported Amazon Orders	231
Default Mappings for Amazon Connector	232
Amazon Category Inheritance for NetSuite Connector	234
Amazon Inventory Adjustment in NetSuite Connector	236
Setting the Handling Time for Amazon in NetSuite Connector	237
Setting Amazon Gift Wrap Items in NetSuite Connector	237
Checking Valid Variations in Amazon	238
Amazon Merchant Shipping Groups in NetSuite Connector	238
Delays in Amazon Order Imports	238
Automatic SKU Creation For Amazon Settlement Charges	238
Manually Creating Amazon Settlement SKUs in NetSuite Connector	239
Resolving Amazon SKU Data Conflicts in NetSuite Connector	243
Importing Amazon Settlement Reports in NetSuite Connector	243
Syncing Sale Price for Amazon Items in NetSuite Connector	247
Assigning a Fixed Customer to All Amazon Orders in NetSuite Connector	247
Adding a New Value to the Amazon Product Type Custom Field in NetSuite	247
Supporting Multiple Unified Amazon Accounts on One Connector	248
Syncing Orders of Other Amazon Marketplaces in the Same Region	248
Managing Amazon Fulfillments Without Tracking Numbers in NetSuite Connector	249
Configuring NetSuite Connector to Use Amazon Standard Identification Number (ASIN) When	
Posting Products on Amazon	249
Using ASIN to Identify an Amazon Product	250
Amazon Tax Remittance	250
Categories, Product Types, and Item Types in the Amazon Flat File Template	250
Managing Fulfillment by Amazon (FBA) Items in NetSuite Connector	251
Verifying that Amazon and NetSuite Connector MWS Authentication Tokens Are Matching	257
Managing Amazon Settlement Sync on NetSuite Connector	257
Amazon Seller Fulfilled Prime Add-On in NetSuite Connector	262
Troubleshooting Amazon Connector Issues	263
Amazon Connector FAQ	265
Managing the BigCommerce Connector in NetSuite Connector	272
Mapping Categories for Full Product Sync in BigCommerce	273
Mapping BigCommerce Call For Pricing Message	273
Default Mappings for BigCommerce Connector	274
BigCommerce Order Status	275
Refund Syncing Issues in BigCommerce Connector	276
Troubleshooting the Issue That An Item has No Inventory, But Customers can Still Order the	
Item in BigCommerce	276
BigCommerce Connector FAQ	276
Managing the eBay Connector in NetSuite Connector	276
Default Mappings for eBay Connector	277
Managing eBay Categories in NetSuite Connector	279
Mapping the eBay Return Profile ID in NetSuite Connector	281
Mapping eBay Buyer IDs to Customer Records in NetSuite	281

Mapping eBay Images in NetSuite Connector	282
Posting 0 Quantity During Product Sync in NetSuite Connector	283
Troubleshooting eBay Product Sync Errors in NetSuite Connector	283
Troubleshooting eBay Sync Issues	284
eBay Connector FAQ	284
Managing Magento 2 Connector in NetSuite Connector	285
Default Mappings for Magento 2 Connector	285
Mapping Magento 2 Custom Product Attributes in NetSuite Connector	288
Creating a Magento 2 Order Invoice in NetSuite Connector	290
Troubleshooting Order Status Not Changed in Magento 2 After Shipping	290
Managing the Shopify Connector in NetSuite Connector	290
Setting Up Real-Time Order Sync for Shopify in NetSuite Connector	291
Default Mappings for Shopify Connector	291
Shopify Inventory Management Mapping	293
Mapping Shopify Multilocation Inventory in NetSuite Connector	294
Mapping Multilocation Orders in Shopify	294
Mapping Shopify Images in NetSuite Connector	295
Mapping Shopify Order Risk Data to NetSuite	296
Mapping Shopify Metafields in NetSuite Connector	297
Tracking Shopify Item Inventory in NetSuite Connector	297
Handling Shopify Cancel and Refund Emails in NetSuite Connector	298
Syncing Canceled Orders to Shopify Using NetSuite Connector	298
Shopify Point of Sale (POS) Orders	299
Assigning Default Values to Shopify Point-of-Sale (POS) Orders in NetSuite Connector	299
Shopify Gift Cards	299
Shopify Product IDs	300
Using the Published Scope and Published Fields for Mapping in Shopify	300
Using Shopify's Compare At Feature in NetSuite Connector	300
Troubleshooting Shopify Sync Errors in NetSuite Connector	301
Shopify Connector FAQ	302
Managing the Walmart Connector in NetSuite Connector	305
Walmart State Tax Remittance	305
Configuring NetSuite Connector for Walmart Two-Day Shipping Delivery Program	306
Walmart Connector FAQ	306
Managing the WooCommerce Connector in NetSuite Connector	307
Default Mappings for WooCommerce Connector	307
Mapping WooCommerce Images in NetSuite Connector	308
Troubleshooting Saving WooCommerce Credentials Failing with Invalid Signature Error	309
Managing Third-Party Logistics (3PL) Connectors in NetSuite Connector	309
Third-Party Logistics (3PL) Order Statuses in NetSuite Connector	309
Mapping Orders and Fulfillments for a 3PL Connector in NetSuite Connector	310
Retrieving and Syncing an Order from NetSuite to 3PL Manually Using NetSuite Connector	310
Creating Custom Filters for 3PLs in NetSuite Connector	311
Syncing Item Fulfillments to a 3PL from NetSuite Connector	312
Mapping 3PL Shipment Methods in NetSuite Connector	313
Handling Partially Fulfilled Orders in NetSuite Connector	313
Troubleshooting Order Import Issues for 3PL Connector in NetSuite Connector	314
Troubleshooting 3PL Fulfillments Posting to NetSuite Without Tracking Numbers	314
3PL Connectors FAQ	315
Managing the ShipStation Connector in NetSuite Connector	316
Fields that can be Mapped to and from ShipStation	316
Resending Orders from ShipStation to NetSuite Connector	317
Checking the Order Status in ShipStation Connector	317
ShipStation Connector FAQ	318
Troubleshooting Common Connector Issues	320

NetSuite Connector FAQ	322
General NetSuite Connector FAQ	322

NetSuite Connector Overview

NetSuite Connector (formerly called as FarApp) provides point integration solutions for leading ecommerce, logistics, and point-of-sale providers, enabling businesses to automate the data transfer between systems.

This chapter covers the following overview sections about NetSuite Connector.

- [NetSuite Connector Data Management](#)
- [User Event Scripts in NetSuite Connector](#)
- [NetSuite Connector Syncs](#)
- [NetSuite Connector Third Party Licensing](#)

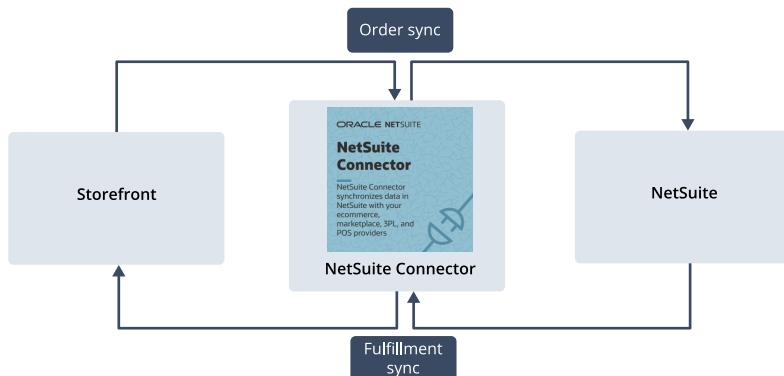
NetSuite Connector Business Flow

NetSuite Connector connects NetSuite to various carts, marketplaces, third-party logistics (3PLs), and shipping software. Each connection between NetSuite and the ecommerce platform is facilitated by a specific connector. Each connector has several syncs that facilitate the import and export of transaction and item-related data between NetSuite and the ecommerce, logistics, and point-of-sales providers.

NetSuite Connector pulls the data from one end and pushes the data to the other. The data flow is dependent on the availability, accessibility, integrity of the data on either endpoint: storefront or NetSuite. The transfer of data is facilitated by syncs that pull data into NetSuite Connector and pushes the data to the destination at defined intervals. This ensures that the data between the storefront and NetSuite are synchronized. After a successful sync, the data can be verified in the storefront, NetSuite Connector, and NetSuite. Understanding the data flows is useful in troubleshooting when there is missing or incorrect data.

Order and Fulfillment Management

NetSuite Connector exports orders from storefronts into NetSuite. When items are fulfilled in NetSuite, the shipping data is exported from NetSuite into storefronts. The following diagram illustrates the flow of basic order management with NetSuite Connector.

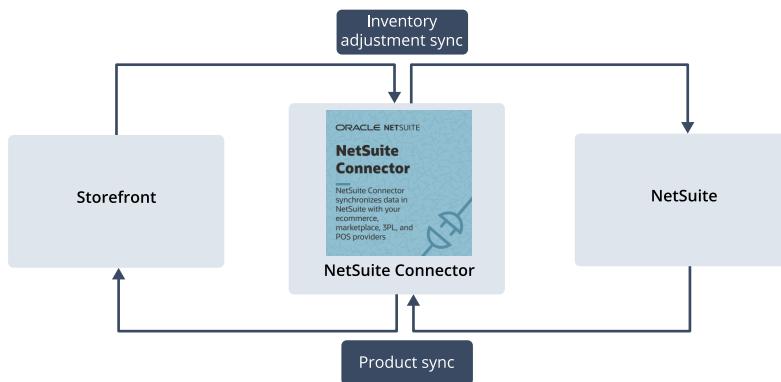


The order and refund syncs ensure that orders transactions are properly exported from the ecommerce channel to NetSuite. The fulfillment sync then exports fulfillment data from NetSuite to the storefront.

- NetSuite Connector Order Sync
- NetSuite Connector Refund Sync
- NetSuite Connector Fulfillment Sync Management

Inventory Management

For inventory management, NetSuite Connector exports product information from NetSuite into commerce channels. The following diagram illustrate the flow of basic inventory management with NetSuite Connector.



Inventory management syncs ensure that the inventory in NetSuite and the storefront match.

- Price and Quantity Sync
- Full Product Sync
- Real-Time Price/Quantity Sync
- Amazon Inventory Adjustment in NetSuite Connector

NetSuite Connector Data Management

This section discusses how NetSuite Connector handles different record and data types. The following topics are covered in this section:

- NetSuite Connector Communication with Internal IDs and Field IDs
- Data Manipulation in NetSuite and NetSuite Connector
- NetSuite Connector Customer Record Handling
- NetSuite Connector Items Handling
- NetSuite Connector Concurrency and Data Limits
- NetSuite Connector Tax Handling
- Gift Cards and Gift Certificates in NetSuite Connector
- Concession and Refund Handling in NetSuite Connector
- NetSuite Connector Carrier Handling
- Types of Shipping Cost Values in NetSuite Connector

- TLS Protocol Support in NetSuite Connector
- Compliance and Fraud Prevention in NetSuite Connector
- Fraud Prevention and NetSuite Connector



Note: NetSuite Connector cannot create or update a support case record in NetSuite. NetSuite Connector can only post data to transaction, item, and customer records.

NetSuite Connector Communication with Internal IDs and Field IDs

NetSuite Connector communicates with NetSuite using internal IDs and field IDs. For example, an order ID like SO12345 is the order ID you see in NetSuite. When you view the order, the internal ID (an integer) is displayed in the URL. This is the internal ID. Likewise, a field like **Item Name/Number** has a field ID of **itemId**. NetSuite Connector cannot communicate with NetSuite using the visible transaction numbers, entity IDs, or field labels you displayed on the NetSuite user interface.

If you change the labels of NetSuite standard fields, NetSuite Connector will not be able to reference the correct field and its corresponding field ID. For example, you modify the **Category** field (with corresponding Category field ID) on a standard customer record, and rename it to **Class**, NetSuite Connector cannot detect that the **Class** field now represents the Category field ID. So you'll need to let us know what the field ID is and not the label.

Every NetSuite record has an internal ID and every field has a field ID. These IDs are the permanent IDs associated with the records and fields and are often visible in the URL when viewing a record.

Field IDs are in SuiteScript format, and the field IDs of standard NetSuite fields (non-custom fields) are in lowercase. However, NetSuite Connector communicates with NetSuite using SuiteTalk, and not SuiteScript. In SuiteTalk, all field IDs are in camel case. For example, **Item Name/Number** field ID is **itemid** in SuiteScript, but it is **itemId** in SuiteTalk.

All the standard SuiteTalk field IDs and internal IDs on all NetSuite records are listed here: [Schema and Records Browser](#).

To reference records, click the **Records Browser** tab.

To reference fields, click the **Schema Browser** tab.

To find an entity, click the initial letter of the name of the entity from the alphabetical row. For example, to find the valid field IDs of inventory items, click the **Schema Browser** tab and click the **I**. Then, in the left column, click **InventoryItem**. All valid field IDs on inventory item records are in the **Name** column.

Data Manipulation in NetSuite and NetSuite Connector

In NetSuite, you can create workflows and scripts to add automation on your data. For example, you can create a workflow that automatically creates a cash sale when a sales order is in **Pending Fulfillment** state. Another example is you can create a script that calculates an item's profitability and automatically populates the profitability on the sales order line item.

NetSuite Connector also has certain capabilities to transform data through calculations and post to NetSuite. However, if an automation or calculation is dependent on data in NetSuite, you should implement a workflow or script in NetSuite. For example, assume that you have a custom record in NetSuite and want to populate information from that record on the sales orders. In such case, you should set up a script in NetSuite because the custom record data is in NetSuite.

Use NetSuite Connector when the automation or calculation is dependent only on the data imported from the storefront. In such case, NetSuite Connector may be able to manipulate the data prior to the sale order posting to NetSuite. You can adjust the order or fulfillment mappings or change the connector settings in the NetSuite Connector. For example, assume that you want to automatically populate the **Department** field on the sales order based on the type of order in the storefront. To achieve this automation, set up a logic mapping for the **Department** field in NetSuite Connector so that the correct value posts to NetSuite.

NetSuite Connector Customer Record Handling

This section covers the topics about NetSuite Connector customer record handling:

- [Limitations on Records and Mapping Export Support for NetSuite Connector](#)
- [Duplicate Record for a Customer](#)

Limitations on Records and Mapping Export Support for NetSuite Connector

NetSuite Connector does not provide the ability to export your mappings, orders, refunds, products, or anything else into a downloadable file. You can only export sync errors.

Duplicate Record for a Customer

By default NetSuite Connector matches customers either by name or company name and email address. Depending on your settings, this matching may look different for your account.

Duplicate customer records usually get created because of mistyped customer information in one of the fields that NetSuite Connector matches. Due to this error, NetSuite Connector is not able recognize an existing customer. For example, Joe at ABC Corporation already has an existing customer record in NetSuite with the email address joe@abc.com. However, during the entry of a future order, Joe accidentally types in his email address as joe@abc.co. In this scenario, NetSuite Connector sees a different email address and therefore may create a new record depending on your settings.

To resolve the duplicate record, merge the new customer record into the existing customer record. For more information, read the help topic [Merging Customer Records](#).

NetSuite Connector Items Handling

This topic discuss handling of different item types in NetSuite Connector.

Handling Serialized or Lot Numbered Kit or Package Items in NetSuite Connector

If NetSuite Connector is configured to post cash sales, some orders still go into NetSuite as sales orders.

If an order contains an SKU that is a kit or package item with a member SKU that is set to use bins, NetSuite will not allow a standalone cash sale or invoice to be created with that SKU. Likewise, if an order contains a serialized or lot numbered kit item, NetSuite will not allow a cash sale to be created.

In both cases, NetSuite Connector automatically compensates by creating a sales order for the item that you must fulfill.

However, the fallback functionality may not be enabled or may not work in older accounts. In this case, contact NetSuite Connector Support.

Matrix Items

Matrix items are also known as family items or parent-child items. Matrix items let you build item families. Families are typically used for products like clothing, where the same item comes in various sizes and colors. If the items feature is enabled, you may create the item as a matrix item when creating an item in NetSuite.

For more information, read the help topic [Matrix Items](#).

NetSuite Connector Concurrency and Data Limits

This section gives details about concurrency and other data limits in NetSuite and their effect on NetSuite Connector. This section includes the following topics:

- [NetSuite Concurrency Limits](#)
- [Throttling Limitation or Limit Change Email from Marketplace or Cart in NetSuite Connector](#)

NetSuite Concurrency Limits

NetSuite's concurrency limits let you allocate a part of your account's concurrent request limits to specific integrations. An account has limitations to the number of times requests can be made to the web services and RESTlet. For more information, read the help topic [Concurrency Limit per Integration](#).

You can monitor your concurrency limit usage from the Concurrency Monitor. For more information, read the help topic [Concurrency Monitor Overview](#).

If you see **Failed** status for the NetSuite integration, NetSuite indicates that the concurrency limit for the integration may need to be adjusted. The **Failed** status for the NetSuite Connector integration does not always mean that the integration is not working. The status merely indicates that the request made at that specific time had failed and NetSuite Connector keeps trying until a **Finished** status is received. The process of trying applies to only NetSuite Connector integration, other integrations may not do the same and may stop at the first **Failed** status. In that case, contact NetSuite Customer Support.

By default, NetSuite Connector limits the concurrent sessions it uses to five. To change this limit, contact NetSuite Customer Support.

Throttling Limitation or Limit Change Email from Marketplace or Cart in NetSuite Connector

You may receive an email from your marketplace or cart regarding throttling change to your automated updates. For example, you might be informed that your number of feeds for updating products has been reduced from 15 to 10 per hour.

If you receive such an email, there is no configuration change required to your NetSuite Connector setup. NetSuite Connector monitors for updates for each marketplace or cart connections and makes necessary updates. You can continue to use the syncs as usual.

In case you notice any errors or anything unusual after such updates, contact NetSuite Customer Support.

NetSuite Connector Tax Handling

This topic covers how NetSuite Connector handles different tax concepts.

Taxable Items

Items in NetSuite are assigned a tax schedule. The assigned tax schedule determines whether the item is taxable.

You can determine the tax schedule of an item from the **Tax Schedule** field in the **Accounting** subtab of the item record. By default, NetSuite assigns **S1** as a taxable tax schedule and **S2** as non-taxable tax schedule.

For more information, read the help topic [Setting Up Tax Schedules](#).

Tax Setting for Customers

NetSuite enables a taxable customer to associate with both taxable as well as non-taxable orders. However, a non-taxable customer can only be associated with non-taxable orders. In NetSuite Connector, the marketplace or cart determines whether the customer pays tax on the order. Therefore, it is essential to allow for a taxable order on every customer even if the tax is charged by error.

A new customer created by NetSuite Connector is always set to taxable. If NetSuite does not set customers taxable by default, an error occurs when NetSuite Connector attempts to create a new customer as taxable. Thus, the order fails to import in NetSuite.

Applying Tax to Settlement Items

If you manually set up settlement item SKUs, make sure you assign them to a nontaxable schedule.

When using NetSuite Connector's automated creation of settlement SKU process, NetSuite Connector attempts to automatically assign the nontaxable tax schedule to the settlement item SKU. However, if such a schedule does not exist, then the item SKUs are assigned taxable tax schedules. Therefore, make sure you create a nontaxable tax schedule and then assign the tax schedule to your settlement item SKUs.

For more information, read [Tax Schedule Errors](#).

Value Added Tax (VAT) in NetSuite Connector

Value Added Tax (VAT) is the sales tax method used in most parts of the world. With VAT, when you show an item price to customers, the price already includes the tax amount in it. Therefore, a customer knows the final price of the item without having to perform any calculations.

In NetSuite, VAT is the same as any other tax. In a typical configuration, NetSuite applies tax to the item and calculates the final item price. Therefore, item prices synced to NetSuite must not include the tax.

In all current marketplaces NetSuite Connector connects to and in most shopping carts, the item price sent on the order includes the VAT. You can configure some shopping carts to provide the pre-VAT prices.

NetSuite Connector can handle both pre-VAT and post-VAT prices. To record the correct item price, you must confirm whether the prices include VAT in the orders that your storefront provides.

VAT and Tax Rounding

You must configure rounding precision in NetSuite, else you may see order total variance due to tax rounding. NetSuite provides different tax rounding options for different countries:

- For United Kingdom (UK), NetSuite does not provide a rounding option and a fixed rounding precision of four decimal places is applied. NetSuite Connector matches NetSuite's precision by rounding four decimal places for orders shipping to the UK.
- For orders shipping to United States (US), the taxes that are used are rounded to a four decimal place precision by both NetSuite and NetSuite Connector.
- For countries that allow a configurable rounding precision, NetSuite sets the rounding precision by default to two decimal places. NetSuite Connector assumes a two decimal place rounding precision for orders shipping to those countries.

You can check and set the rounding precision from the **Tax Rounding Precision** field in the Set Up Taxes page of NetSuite. The field is available only for the countries that allow configuring rounding precision.

For more information about configuring NetSuite Connector with VAT, read [Configuring VAT in NetSuite Connector](#).

NetSuite Connector VAT Calculation

Before sending orders into NetSuite, NetSuite Connector can subtract Value Added Tax (VAT) from orders. You can configure this using the tax settings for each account on the connector.

For more information about VAT settings, read [Configuring VAT in NetSuite Connector](#).

Gift Cards and Gift Certificates in NetSuite Connector

NetSuite Connector treats card data for gift cards in a manner consistent with credit card data. Like credit card, the marketplace or cart does not allow NetSuite Connector to access the gift card information. Therefore, NetSuite Connector cannot sync the remaining balance of a gift card back to NetSuite.

For more information, read [Configuring Gift Cards for NetSuite Connector](#).

Concession and Refund Handling in NetSuite Connector

When posting modifications to the original order for settlement report processing, NetSuite Connector performs the following actions by default :

- A concession or similar entity is posted as an adjustment to the original transaction but does not affect the inventory for any SKU on the order.
- A refund is created from the billed record of the order and affects the inventory of the original SKU.

NetSuite Connector Carrier Handling

NetSuite Connector derives the carrier value from the name of the shipment method on the item fulfillment. For example, if the shipping method is called **FedEx 2 Day**, NetSuite Connector derives the

carrier value as **FedEx**. However, for generic shipment methods from which NetSuite Connector cannot derive the carrier, usually NetSuite Connector sends the shipment method as the carrier. If you have multiple generic shipment method names or uncommon regional carriers, you must create mappings for the **Carrier** field or use the custom **Carrier** field.

When you sync orders to a 3PL for fulfillment, you can map shipment methods coming from 3PL to a different shipment method in NetSuite. However, make sure the shipment method mapped in NetSuite uses the same carrier as the one used with the shipment methods in the 3PL.

Types of Shipping Cost Values in NetSuite Connector

There two different shipping cost values used, one during order creation and other during order fulfillment.

Shipping cost in reference to order creation can be more accurately referred to as shipping revenue. This amount is the payment received from the customer for shipping. The amount is posted to the **Shipping Cost** field on the sales order in NetSuite.

Shipping cost in reference to order fulfillment is the amount that cost you to ship the order. This amount is posted on the item fulfillment record.

If you have a third-party logistics (3PL) providers or shipping software like ShipStation connected, this value is automatically posted on fulfillment data sync from the 3PL or shipping software.

TLS Protocol Support in NetSuite Connector

NetSuite Connector supports the latest encryption standards. If a system that you are using is deprecating older encryption standards like TLS 1.0 or TLS 1.1, the NetSuite Connector service will not be affected.

Compliance and Fraud Prevention in NetSuite Connector

NetSuite Connector imports credit card tokenization details by default to maintain PCI compliance. Most storefronts cannot capture full payment details. However, if a storefront supports this feature and NetSuite Connector imports this data to NetSuite, it would be a breach of PCI compliance.

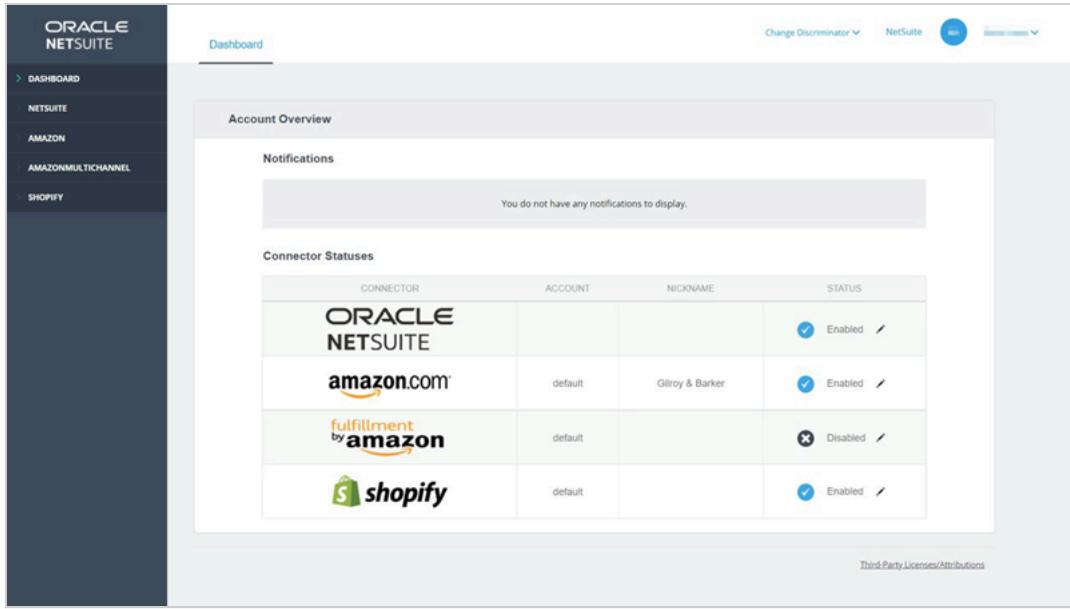
Fraud Prevention and NetSuite Connector

NetSuite Connector does not have the ability to detect frauds. Fraud prevention should be done directly in the marketplace or cart or through the payment gateway where the customer enters the payment information.

For more information about how NetSuite Connector handles credit card information and PCI compliance, read [Compliance and Fraud Prevention in NetSuite Connector](#).

NetSuite Connector User Interface

The NetSuite Connector dashboard provides a quick overview of the status of the account's connectors and notifications. The dashboard is the default view upon logging in to NetSuite Connector and is accessible from the navigation bar on the left of the page.



You can manage your profile, account settings, and billing information by clicking your profile name on the top right of NetSuite Connector and selecting the relevant menu option.

The connectors of the currently logged in NetSuite Connector account can be accessed and managed on the navigation bar on the left of the page. When clicked, the connector menu expands to display the menus for Data Flows, Settings, and Mappings. If the connector has multiple accounts, you need to select the relevant account from the Select Account list of the connector accordingly. If no account is selected, the default account is used.

You can also use the NetSuite Connector UI to set up other users who can access the NetSuite Connector account. The views and permissions of these added users can be configured to ensure that they can only access features and information relevant to their tasks.

It is possible for users to be added to multiple accounts or discriminators as Partner Users. This type of access is usually granted to consultants for setup and configuration or troubleshooting support. Partner User access expires after 14 days. For implementations that last longer than 14 days, you need to add the user as Partner User again.

A user added to multiple accounts can switch from the available accounts in NetSuite Connector by clicking **Change Discriminator**, then selecting the desired account or discriminator.

Contact NetSuite Customer Support to add a new connector.

User Event Scripts in NetSuite Connector

The NetSuite Connector SuiteApp includes the following user event scripts:

- **FA | Fix Shipping Tax** – When NetSuite creates a transaction using the data from NetSuite Connector, this script runs and checks the shipping tax rate in the transaction. If required, the script fixes the shipping tax to the rate set in the data coming from the NetSuite Connector.
- **FA | Fix Tax Rate** – When NetSuite creates a transaction using the data from the NetSuite Connector, this script runs and checks the tax rate in the transaction. If required, the script fixes the tax rate to the rate set in the data coming from the NetSuite Connector.
- **FA | UE Fix Order Item Group Prices** – When NetSuite creates a transaction using the data from NetSuite Connector, this script runs and checks the prices of item groups in the transaction. If required, the script fixes the prices to the prices set in the data coming from the NetSuite Connector.

- **FA | UE Order Total Validation** – When NetSuite creates a transaction using the data from NetSuite Connector, this script runs and validates the transaction data. The transaction data such as tax total, transaction subtotal, and cost of shipping is validated with the order total data sent by NetSuite Connector. If the validation passes, the transaction is created in NetSuite. If the validation fails, the transaction is not created and you get an error message in NetSuite Connector.
- **FA | UE Sync - Update FA** – When you create, update, or delete an item in NetSuite, this script runs to sync the changes to NetSuite Connector. The script runs only when the **Disable Real-Time Sync** setting in the NetSuite Connector Setup page is disabled.

Order of Script Execution

These user event scripts must run in the following order:

1. FA | Fix Shipping Tax
2. FA | Fix Tax Rate
3. FA | UE Fix Order Item Group Prices
4. FA | UE Order Total Validation

NetSuite Connector Authentication

NetSuite Connector data flows rely on active connections to the storefront accounts, NetSuite account, and NetSuite Connector account.

- **NetSuite Connector account** – The account you use to log in to NetSuite Connector (farapp.com). You use this account to manage your connectors, syncs, and mappings in your NetSuite Connector dashboard.
- **NetSuite account** – The account you use to log in to NetSuite. You use this account to connect to the NetSuite side endpoint of the data flow.
- **Storefront account** – The account you use to log in to your storefront web site. Each connector needs this account to be able to push and retrieve data to and from the specific storefront. Depending on the storefront, you can use token-based or credential-based authentication to authorize NetSuite Connector to connect to your storefront accounts.

NetSuite Connector Syncs

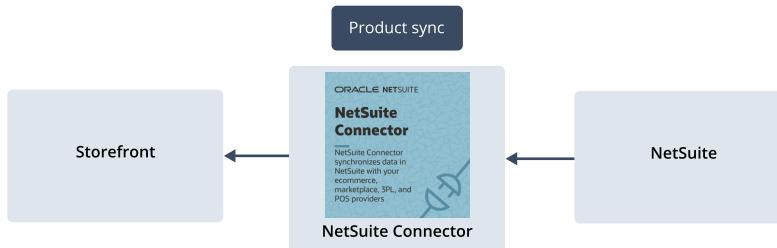
NetSuite Connector enables you to work with following syncs:

- Product sync – Read [NetSuite Connector Product Sync](#)
- Order sync – Read [NetSuite Connector Order Sync](#)
- Refund sync (or Amazon settlement sync) – Read [NetSuite Connector Refund Sync](#)

NetSuite Connector Product Sync

Product sync synchronizes data from NetSuite to the storefront. This enables you to maintain items and inventory numbers in multiple storefronts by using NetSuite as the primary data.

The flow for product syncs is illustrated in the following diagram.



Note: Products sync from NetSuite.

This section discusses the following about product sync:

- Product Sync Types
- Product Sync Mechanisms
- Product Mapping Types
- NetSuite Connector Product Sync for Items
- NetSuite Connector Product Sync Triggers
- NetSuite Connector Storefront Flag Fields
- NetSuite Connector Product Categories
- Syncing Quantity Updates
- NetSuite Connector Push Item Update Scripts in NetSuite
- Buffered Quantity
- NetSuite Connector Product Image Handling

Product Sync Types

NetSuite Connector provides the following two types of product syncs:

1. Full product sync
2. Price and quantity sync (sometimes referred to as Price/Qty sync)

Full Product Sync

NetSuite Connector syncs full product data such as title, description, images, price, and quantity from NetSuite to the storefront.

Note: Price and quantity sync is part of full product sync. Therefore, when you do a full product sync, you need not enable price and quantity sync separately.

In full product sync, you should create new items in NetSuite and rely on NetSuite Connector to create new listing in the storefront.

Price and Quantity Sync

NetSuite Connector only syncs the data of price, quantity, or both from NetSuite to the storefront. In this sync, NetSuite Connector sends only the price, quantity, and the SKU information. Therefore, the SKUs using this sync must already exist in the storefront along with the title, description, images and other information.

Mappings

You can configure both types of product sync mappings from the Product Mappings page of NetSuite Connector.

For more information, read [Mapping Product or Inventory Sync in NetSuite Connector](#).

Difference between Price/Quantity Sync and Real-Time Price/Quantity Sync

There are differences between normal Price/Quantity Sync and Real-Time Price/Quantity Sync. Although both syncs may update the same fields, the main differences between the two is the timing.

Price/Quantity Sync

The normal Price/Quantity Sync only syncs the price or quantity of an item from NetSuite to your marketplace/cart. After the items are posted to NetSuite Connector, the Price/Quantity sync will post item updates to the marketplace/cart at 60-minute intervals.

Real-Time Price/Quantity Sync

Likewise, Real-Time Price/Quantity Sync only syncs the price or quantity of an item from NetSuite to your marketplace/cart. However, Real-Time Price/Quantity Sync depends on the scripts in NetSuite. Real-Time Price/Quantity Sync will run on every updated record and posts the data to NetSuite Connector almost immediately after the record has been updated. If you have the real-time sync enabled, NetSuite Connector will post the item updates almost immediately to the marketplace./cart.

These two syncs have limitations based on the volume of items updated and other factors.

For more information, see [Product Sync Mechanisms](#).

Leading Practices for Full Product Sync

NetSuite Connector syncs items data in NetSuite with existing SKUs on your storefront. NetSuite Connector overwrites the mapped fields with the data from NetSuite but does not change any fields that are not mapped for sync. For some storefronts, if nothing is sent on the sync for a field, the storefront uses a default value for the field.

NetSuite Connector considers NetSuite as the primary record for your SKU data. Therefore, you should enter all the product data you want to see on an SKU into NetSuite and let NetSuite Connector sync the data.

Price/Quantity Sync vs. Full Product Sync

NetSuite Connector provides two types of product sync, namely, Full Product sync and Price/Quantity sync. At a time, you can run only one type of sync on a connector and account combination. This topic provides details all the sync types to enable you to decide the type of sync for your account.



Important: You cannot switch between Price/Qty sync and Full Product sync by clicking the buttons on the Product Mappings page or the Manage Data Syncs page. If you want to change the product sync type, contact NetSuite support.

Price/Quantity Sync

Following are the key features of the price/quantity sync:

- You can sync either price, quantity, or both.
- Only items with a valid value in the designated flag field sync.
- When using Price/Quantity sync, the SKU must exist in the marketplace or cart for the price and quantity to sync.
- Connectors with certain price tiers also include real-time price and quantity sync on supported connectors. Real-time price and quantity sync enables the price and quantity updates reach the marketplace or cart immediately without waiting for the scheduled sync to run.
- The real-time sync takes place on the servers and does not show in the NetSuite Connector product dashboard.
- For matrix items, you need not sync the parent items.

Full Product Sync

Following are the key features of the full product sync:

- All item fields available through API can sync.
- Only items with a valid value in the designated flag sync.
- Do not switch to price/quantity sync to send the price or quantity values. If you have mapped the price and quantity fields, the values will be sent as part of the full product sync.
- Each marketplace or cart defines certain set of fields required for a full product sync. Other than the required fields, you can optionally include or exclude other fields.
- Not mapping a field or not populating a mapped field does not change the data for the field in the marketplace or cart. If you do not create a mapping for a field or have a blank mapped field, NetSuite Connector does not send data for that field.
- NetSuite Connector uses the SKU to sync to the existing product in the marketplace or cart. If an SKU does not exist, NetSuite Connector creates a new product using the available data from the fields mapped in NetSuite.
- For matrix items in NetSuite, set a value in the flag field of the parent item.
- For Amazon, values that you send for many fields must match the existing values in Amazon. If these values do not match, the sync fails and NetSuite Connector provides details on which fields should be changed and to what values. You can significantly reduce the fields Amazon requires to match by sending the ASIN product identifier.

For more information, read [Configuring NetSuite Connector to Use Amazon Standard Identification Number \(ASIN\) When Posting Products on Amazon](#).

Product Sync Mechanisms

Pushing a large amount of items to sync at a time causes delay in product updates. A large CSV product upload or mass update in NetSuite, which may update more item records than needed, may cause this issue.

When you send item updates to NetSuite Connector, these two product sync mechanisms govern the process.

Sync Mechanism	Description
Real-Time Sync	As the mass update changes the item records, the real-time sync processes every updated record and pushes the data to NetSuite Connector. In some cases, the real-time sync will not run with mass updates, which is beyond the control of NetSuite Connector. Instead, the process relies on NetSuite's SuiteScript governance model, which is based on usage units intended to optimize application performance. For more information about NetSuite's script usage limits, read the help topic Script Type Usage Unit Limits .
Scheduled Script (Runs every 15 minutes)	The scheduled script looks for any missed item updates and pushes them as quickly as possible. However, NetSuite imposes a limit on the number of products the scheduled script can process, which is about 1000 items per script execution. The script will push what it can every 15 minutes. If the number exceeds the limit of units allowed, the process terminates the script execution. If you do not adjust your workflow or number of items for syncing, the script loops in a cycle of timeouts and terminated processes before it completes.

The item data upload time depends on whether the marketplace/cart has a limitation on the number of times you can access the API within a certain period. Like NetSuite, most APIs have a limit to prevent an overload of requests that may end up crashing the system.

When you set the API limits, you must use the lowest limit as the determining factor. For example, if NetSuite can handle 1000 units per execution, but Shopify can only handle 500, then the limit per execution from NetSuite to Shopify should be 500.

When NetSuite Connector repeatedly fails to process your entire product queue, the first thing to check is the number of items updated per hour. The greater the number of product updates, the longer it will take for NetSuite, NetSuite Connector, and the marketplace/cart to process.

If you notice that you have more than 1000 items updating per hour, adjust the conditions to your workflow in NetSuite. The workflow condition may include items that do not require an update. When you set the workflow conditions correctly, the process updates only the relevant items, and not your full item catalog.

You should add a workflow condition that checks whether the field is already set and updated. This way, the workflow only runs when it needs to.

For example, if you have a workflow configured to set a field to True, the workflow should first check whether the field is already set to True. If you skip this step, the workflow will trigger an update on the item record even though there are no changes. This step will also cause the SuiteApp to push an update to NetSuite Connector. For more information, read the help topic [Workflow Conditions](#).

Product Mapping Types

There are several different types of product mappings you can create. The type of mapping is identified in the left-most column of the Product Mappings page.

When you create a product mapping, there are several mapping type options:

- **Use Default**

This is the simplest type of mapping. Enter a value in the **Default Value** column. When creating or updating products for the current channel and category, NetSuite Connector will automatically enter the value in the corresponding field. Do not use a NetSuite field name with this mapping type, else the literal field name will be posted into the destination field.

- **Item Field Mapping**

This lets you map a field from NetSuite to an item field on your sales channel by entering its Field ID under the **NetSuite Field ID** column. A common example is mapping **itemId** (Item Name/Number) to the SKU, or **storeDisplayName** to the title. You can still enter a default value in this case, which will be used if the field in the NetSuite Field ID column is not entered for an item.

■ **Item Field Translation Mapping**

Sometimes, the value you have in NetSuite for a field may not be the value you want to map to your sales channel. This option lets you set up correspondences, or translations, between NetSuite values and the values that NetSuite Connector posts to the sales channel for a field. When you select Item Field Translation Mapping, the NetSuite Field ID column value will change to a link. Clicking this link opens a popup and lets you set up the translations.

■ **Logic**

This mapping uses conditional check or other logical computation. These are usually IF or THEN type mappings. Although they cannot be selected from the mapping type list, two additional types are displayed if applicable.

■ **Special**

This type is shown for certain mappings that provide a popup window, like Shopify multi-location inventory mappings. You can edit these mappings, but whether a field will receive a special mapping will be automatically determined by NetSuite Connector.

■ **Custom**

If NetSuite Connector support has previously worked on your mappings, you may see some mappings with this type. This indicates that something about your mapping for the field is atypical. If you need to edit one of these mappings, contact NetSuite Customer Support.



Note: Custom mappings are no longer supported with NetSuite Connector. Existing custom mappings can remain and change requests to custom mappings will be limited to removing the mapping and replacing it with a standard mapping.

For an example of mapping a field please see [Mapping Required and Optional Fields for Product Sync](#).

NetSuite Connector Product Sync for Items

This section describes the item types, their difference and how NetSuite Connector handles their quantity in syncs.

- [Kit Items and Item Groups](#)
- [Assembly Items](#)

Kit Items and Item Groups

Kits are individually sold items composed of other items. You can add the following to your kits or packages:

- Description
- Inventory
- Non-inventory
- Other charge
- Service
- Kit
- Gift certificates

- Assembly items

NetSuite Connector syncs the inventory level of the component with the lowest inventory, because that value is the maximum number that you can completely create for the kit or item group.

For example, a shower or bath kit may contain body wash, a loofa, shampoo and conditioner, and various other items. These items are all sold together as the kit item, shower or bath kit, but the components can also be sold as individual items as well.

Kit items are posted to your marketplace or cart like any other item, the only difference is that the kit item itself does not have any quantity. Kit item inventory is not tracked by the kit, it is tracked by individual component members. The kit item record does not have an available quantity. The quantity of a kit item is determined by its components' lowest denominator. Therefore, if you view a kit item SKU in NetSuite, the quantity is 0. But when NetSuite Connector posts the kit to the marketplace or cart, it will use the quantity of the component with the least inventory amount. The following is an example:

Kit item: Shower or Bath Kit

Components: body wash, loofa, shampoo, conditioner

Quantities available per component item:

- body wash: 5
- loofa: 10
- shampoo: 13
- conditioner: 21

The quantity of the kit item, shower or bath kit, will be 5.

For more information, see the help topic [Kit/Package Items](#).

Assembly Items

An assembly item is an inventory item containing several components but is identified as a single item. Assemblies are manufactured by combining raw materials you stock. After you create assembly item records that become the members of an assembly, you can track both the raw materials and the assembled items separately. An example of this is a bike. The bike is assembled with various components like wheels, frame, handlebars, seat, and other parts.

Assembly items also post to your marketplace or cart like any other item. Assembly items sync inventory like kit items by using the quantity of the item in the group of components that has the least inventory. But the assembly item can also have quantity. NetSuite Connector can sync that quantity to the marketplace or cart, instead of the component quantity. The following is an example:

Assembly item: Bike

Components: wheels, frame, handlebars, seat

Quantities available per component item:

- wheels: 8
- frame: 3
- handlebars: 10
- seats: 21

In this example, if the bike has a quantity of 2, NetSuite Connector will either sync 2 or 5.

For more information, see the help topic [Assembly Items](#).

NetSuite Connector Product Sync Triggers

If you properly set up the product sync, the following events trigger NetSuite Connector to retrieve items from NetSuite, and sync each item to the storefront.

Items will sync when either a primary or a secondary trigger occurs. The following section describes each trigger type.

Primary Triggers

Primary triggers are changes to the item attributes that trigger the product sync. When a primary trigger occurs, NetSuite Connector retrieves the item data and syncs it to the storefront.

A primary trigger occurs when one of the following scenarios takes place:

- The **lastModifiedDate** or **Updated At** field changes on the item record in NetSuite to a date that is later than the last date NetSuite Connector checked for changed items.
- The inventory for the item changes in one of the NetSuite locations.

Secondary Triggers

Secondary triggers happen when an item associated with another item receives a primary trigger. Secondary triggers cause items to re-sync to the storefront, but do not cause NetSuite Connector to get the item data from NetSuite. Retrieving item data from NetSuite is unnecessary because there are no changes to the item record itself. Only the item record that created the secondary trigger has changes, and NetSuite Connector only needs to update the data for that item.

A secondary trigger occurs when one of the following scenarios takes place:

- An item in the same family (either a parent of the children SKUs or one of the child SKUs) receives a primary trigger
- A component of a syncing Kit or Assembly Item receives a primary trigger

The following table provides examples of how primary and secondary triggers occur:

Action	Result
When the Class field on an item changes	When NetSuite saves the item record, a primary trigger occurs for the item and updates the lastModifiedDate field.
When the inventory in Location 1 for an inventory item changes	When NetSuite Connector detects an inventory change on a location assigned to the item, a primary trigger occurs.
When the Department field on a child item SKU Shirt-Red in a matrix family changes	When NetSuite saves the item record, a primary trigger occurs for SKU Shirt-Red and updates the lastModifiedDate field. When NetSuite saves the item record, a secondary trigger occurs for the parent SKU Shirt-Red and any other child SKU in that same family (Shirt-White and Shirt-Blue).
When the SKU CMP1 item has an inventory change in Location 1. This item is a component of SKU MyKit, which is a Kit item set to sync.	When NetSuite saves the item record, a primary trigger occurs for SKU CMP1 and updates the lastModifiedDate field.

Action	Result
	Note that even if CMP1 is not set to sync, NetSuite Connector still retrieves the item record for the SKU because it is a component of SKU MyKit, which is set to sync.
	When NetSuite saves the item record, a secondary trigger occurs for the parent SKU Shirt-Red and any other child SKU in that same family (Shirt-White and Shirt-Blue).

NetSuite Connector Storefront Flag Fields

For product sync, NetSuite Connector requires a custom field for each account of a connector to use as a storefront flag. By setting the value of this field in NetSuite, you can sync data to items on the storefront.

If you install NetSuite Connector SuiteApp, NetSuite Connector creates flag fields automatically.

The following table lists the values that you can select for flag fields:

Value	Definition	Notes
(blank) or Ignore Item	Ignored by NetSuite Connector.	NetSuite Connector does not load data from NetSuite if the flag field is blank.
Add/Update Item	Add or update product data to the storefront.	.
Remove Item	Delete product or listing from storefront.	If an item is flagged as Y , and later you decide to delist the item, you must flag it to N . Note: If you are using price and quantity sync, setting the item to Remove Item does not delete the item from the marketplace or cart. The Remove Item setting only stops syncing the item.
Post Children as Stand-Alone	Parent of stand-alone items. C means post only children.	Used for NetSuite matrix item parents. Child items are posted as stand-alone (non-variation) items. NetSuite Connector needs the parent data to populate child items. This setting enables NetSuite Connector to load, but not post the parent data. To use this setting, set the parent item to C and child item to Y .
Post Inventory Only	Inventory-only flag. Supported only for eBay connector.	This value is used for posting only inventory updates to eBay when you do not want full product updates.
Post Price Only	Price-only flag. Supported only for eBay connector.	This value is used for posting only price updates to eBay when you do not want full product updates.
Post Price/Inventory Only	Price and inventory flag. Supported only for eBay connector.	This value is used for posting only price and inventory updates to eBay when you do not want full product updates.

Note: Do not change the values that are selectable from this dropdown list. Changing the values can cause syncing issues.

For leading practices on using storefront flag field, read the help topic [Storefront Flag Field Leading Practices](#).

If you are not using one of the following marketplaces or carts, you will have to create the storefront flag field manually:

- Amazon
- BigCommerce
- ChannelAdvisor
- eBay
- Jet
- Magento
- Magento2
- Miva
- Shopify
- Walmart
- WooCommerce

For more information, read [Creating a Single-Character Storefront Flag Field Manually](#).

To manually create a list-based storefront flag field, read [Creating a List-Based Storefront Flag Field Manually](#).

Storefront Flag Field Leading Practices

Follow these leading practices when using storefront flag field for syncing items from NetSuite:

- To list an item, change the item's flag field value from blank to **Add/Update Item** (or **Y**).
- To remove an item, change the item's flag field value from **Add/Update Item** to **Remove Item** (or from **Y** to **N**).
- Do not change the flag field from blank to **Remove Item** (or **N**). This change can cause errors.
- To remove an item, do not change the flag field from **Add/Update Item** (or **Y**) to blank.
- When using matrix items in Price/Qty sync, do not set the parent item to sync. Set the child item flag field to **Add/Update Item** (or **Y**).
- When using matrix items in full product sync, set the flag fields of both parent as well as child items to **Add/Update Item** (or **Y**).

For additional information, read the following topics:

- For information about flagging an item to sync, read [Flagging Existing NetSuite Items to Sync](#).
- For flag fields created by the NetSuite Connector SuiteApp, read [NetSuite Connector Storefront Flag Fields](#)
- For manually created single character flag fields, read [Creating a Single-Character Storefront Flag Field Manually](#).
- To manually create a list-based storefront flag field, read [Creating a List-Based Storefront Flag Field Manually](#).

NetSuite Connector Product Categories

When importing items from NetSuite, NetSuite Connector determines the set of mapping fields that should be used for the product. This determination of mapping fields is also called as categorization and you must create mappings for the **Category** fields for each product.

After you import an item to NetSuite Connector, you can see how the item is categorized, or the set of fields used by the item. You can use only a single category for all items at a time.

The following table provides details of category mapping for each connector type:

Product Type or Connector	Details
Wayfair, OverStock, and shopping carts (such as Shopify, Magento, and others)	<p>In shopping carts, there are two choices for categories, namely Full Product Sync category and Price/Qty category. Full product sync category names are denoted as <cart name> and Price/Qty category names are denoted as <cart name>PriceQty.</p> <p>For category mapping information, read the topic Mapping Cart Categories in NetSuite Connector.</p>
Marketplaces (Amazon, Walmart, NewEgg)	<p>When using full product sync, marketplaces create either of the following:</p> <ul style="list-style-type: none"> ■ For Amazon, creates a custom category. ■ For other Walmart and NewEgg, if required, adds a category with a name that includes the product type, like WalmartHomeOther3.2. <p>For Price/Qty mapping for a marketplace, the category name is <marketplace name>PriceQty. For example, Walmart uses the name WalmartPriceQty.</p> <p>For Amazon, the category names is AmazonInventoryPrice.</p> <p>For category mapping information of Amazon connector, read the topic Mapping Amazon Categories in NetSuite Connector.</p>
eBay	<p>For eBay, category mappings are not required. The products are automatically categorized based on whether the items are variation items.</p> <p>Products that are variation items, also known as matrix items, are categorized under eBayVariations category.</p> <p>Products that are standalone items are categorized under eBayFixedPrice category.</p>

Syncing Quantity Updates

NetSuite Connector supports syncing of item quantity updates to a storefront, for the following scenarios:

- If warehouses or locations have not been set up in NetSuite, the native **quantityAvailable** field in NetSuite, or a custom field, can be used.
- If warehouses or locations have been set up in NetSuite, the quantity available in a given warehouse can be used.
- If warehouses or locations have been set up in NetSuite, the quantity available from multiple warehouses can be added.

For more information about mapping inventory and quantity, see [Mapping Inventories in NetSuite Connector](#).

NetSuite Connector Push Item Update Scripts in NetSuite

The NetSuite Connector push item update scripts help NetSuite Connector find records that affect inventory. With this information, NetSuite Connector adjusts the appropriate records to ensure that the item inventory in NetSuite is updated. If you have product sync or price and qty sync enabled, the scripts ensure that the inventory is correctly synced back to the marketplace or cart.

If you do not want to use the scripts, you can disable the scripts. For more information about disabling the scripts, read [Undeploying Scripts Installed by NetSuite Connector](#).

Buffered Quantity

NetSuite provides you the capability to assign or map quantity available to the marketplace/cart. This is the total number of units for a particular SKU that are currently available for use in filling new orders. Assigning a quantity buffer causes the marketplace/cart to display a lower quantity available than in stock, which can help prevent unintentional overselling of product.

There are two ways you can set up a buffered quantity:

- **Specific amount for the buffer.** Set the buffer to 50 makes the quantity posted to the marketplace/cart would be 50 less than the quantity available in NetSuite.
- **Percentage-based amount buffer.** Setting the buffer to 50% makes the quantity posted to the marketplace/cart half the quantity available in NetSuite.

You can complete these through arithmetic mapping. For more information about arithmetic mapping, see [Configuring Quantity Calculations Before Sync in NetSuite Connector](#).

Another feature is to set up a box on your items that determines if the buffered quantity mapping would apply to that particular item. If selected, the buffer quantity mapping would apply. If not selected, the mapping would not apply. You can control which items the buffer quantity mapping will affect through this box.

To set up a buffered quantity through this feature, contact NetSuite Customer Support and provide the NetSuite field ID for the box, along with two SKUs to test the mappings. One of the SKUs will have the box selected and the other not selected.

NetSuite Connector Product Image Handling

This section contains best practices to ensure the proper sync of product images from NetSuite to the marketplace.

NetSuite Configuration

NetSuite has a native item image field that you can use for images. However, if you want to sync more images, you can create custom fields.

For more information about adding custom fields, read [Adding NetSuite Custom Fields](#).

Ensure that your products are set up properly in NetSuite and their image fields are populated with the images or image URLs you want to sync. For more information about setting images on your NetSuite items, see [Adding Product Images for NetSuite Connector](#).

NetSuite Connector Configuration

With the images set up in the NetSuite, you must update the mappings to pull the image fields into NetSuite Connector to post them to the marketplace or cart. Most connectors have a standard image field that you can select from the NetSuite Connector mapping dropdown fields.

Updating Images Hosted in NetSuite

The best practice for updating images is to delete the old images first and then add the new images, instead of only updating existing images in the NetSuite File Cabinet. If you do not delete the old images first, the marketplace or cart will not recognize that a change has been made and will keep the old image displayed. To avoid this, you must delete the images entirely from NetSuite and then upload new images so the internal IDs of the images can be reset as well.

For more information, see the help topics [Deleting Images](#) and [Uploading Images in Images](#).

Preserving Existing Images in the Marketplace or Cart

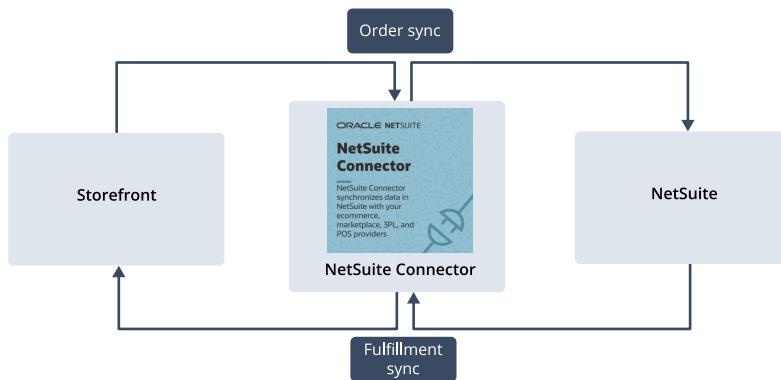
You can preserve existing images by not mapping them in NetSuite Connector. With this, the syncs will not add or update the images and therefore will not update the image field in the marketplace or cart.

If you do not want to delete the image in the marketplace or cart, you can delete the data in the mapped field for images on the product, and leave it blank. This will prevent NetSuite Connector from sending the data for that image.

NetSuite Connector Order Sync

Order Sync is the process in which orders are placed in the storefront, flow into NetSuite Connector, and post to NetSuite.

The flow of order and fulfillment syncs is illustrated the following diagram.



Note: Orders sync to NetSuite, and fulfillments sync from NetSuite.

NetSuite Connector Order Statuses and View Filters

The NetSuite Connector dashboard displays information that helps you determine the stage of processing of your order. Additionally, an advanced view filter is also provided to specify the criteria for the orders currently visible in the NetSuite Connector order dashboard.

Advanced view filters are different from order filters. Order filters control the import of orders from the storefront (or from NetSuite when using 3PL connector).

Order Status

The following are the order statuses:

- **Scheduled to Post to NetSuite** – Appears when an order is in queue to post to NetSuite. This status appears in the following cases:
 - When an order is first pulled into NetSuite Connector and NetSuite Connector attempts to post to NetSuite.
 - When there was an error in an order previously and you are attempting to post again with the newest batch of orders.

- **Error Posting to NetSuite** – Appears when there is an error posting an order to NetSuite. Hover the pointer over this error to view details.
- **Order Posted, Waiting for Shipment** – Appears when the order successfully posts from your storefront to NetSuite. NetSuite Connector checks if the order was fulfilled. If yes, then NetSuite Connector syncs the fulfillment data back to the storefront. If you do not find the order in NetSuite, click **Show NetSuite Transaction ID** to view the NetSuite order ID.
- **Error Posting Fulfillment** – Appears when there is an error posting a fulfillment to an order. The status appears in the following cases:
 - A fulfillment in your 3PL failed to post to NetSuite.
 - A fulfillment in NetSuite failed to post to the storefront.
 Hover the pointer over this error to view details.
- **Complete** – This status indicates that the order has been posted from the storefront to NetSuite and the fulfillment data is posted from NetSuite back to the storefront.
- **Canceled** – Orders with this status neither attempt to post to NetSuite nor attempt to post fulfillment data back to the storefront. The orders are canceled in the following cases:
 - When you click the pencil icon next to an order and click **Cancel**.
 - When you post an order too many times, the order is automatically canceled.
- **Held** – Orders with this status do not attempt to post to NetSuite. The orders are held in the following cases:
 - When you click the pencil icon next to an order with the status **Schedule to Post to NetSuite** or **Error Posting to NetSuite** and click **Hold**.
 - If your account settings have the setting to hold an order if there are any errors.



Note: You cannot hold orders that are posted to NetSuite.

Filters

You can select filters on the Orders page by clicking **Filter Orders**. Following are the options and the status for each option:

Filter	Status
All	<ul style="list-style-type: none"> ■ Scheduled to Post to NetSuite ■ Error Posting to NetSuite ■ Order Posted, Waiting for Shipment ■ Error Posting Fulfillment ■ Complete ■ Held ■ Canceled
Open	<ul style="list-style-type: none"> ■ Scheduled to Post to NetSuite ■ Error Posting to NetSuite ■ Order Posted, Waiting for Shipment
Error	<ul style="list-style-type: none"> ■ Error Posting to NetSuite ■ Error Posting Fulfillment
Complete	Complete

Filter	Status
Canceled	Canceled
Held	Held

Advanced Order Filter

You can further filter orders by a storefront order field and a fixed value. For example, you can filter orders by Status/Region when the field equals to North Carolina. This will show all orders from North Carolina.

You can also apply multiple advanced filters by clicking **Add Filter Row** to further limit the order you want to view. When you apply multiple filters, an order appears in the list only when all statements are true for the order.

NetSuite Connector Refund Sync

Refund Sync is when a client refunds an order in NetSuite and it posts back to the storefront, or reverse. Some connectors permit either direction to be selected but no connector permits both directions at the same time.

The flows for refund syncs are illustrated in the following diagrams

For Marketplaces (Amazon, eBay, and others)



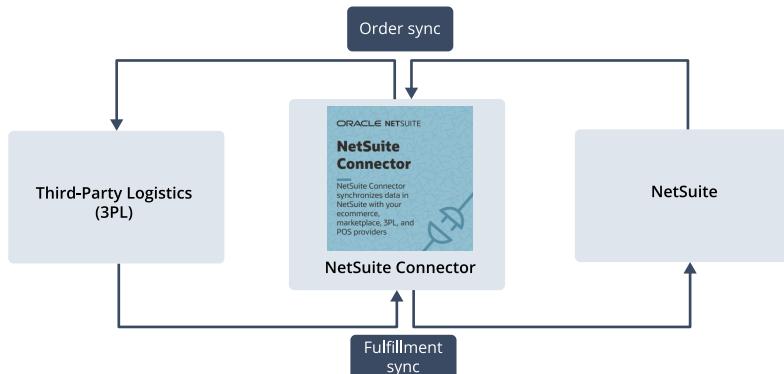
For Shopping Carts (BigCommerce, WooCommerce, and others)



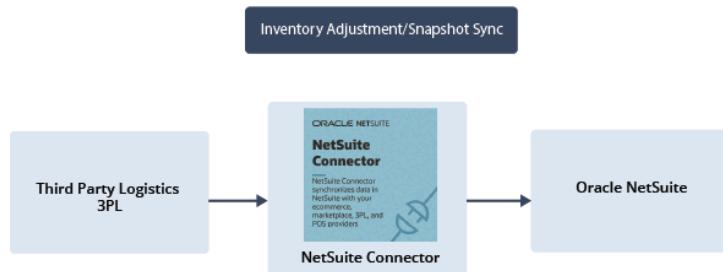
NetSuite Connector Third-Party Logistics (3PL) Sync

3PLs can have several different types of syncs including Order and Fulfillment, Inventory Adjustment and Transfer Order. The flows of these syncs are illustrated in the following diagrams.

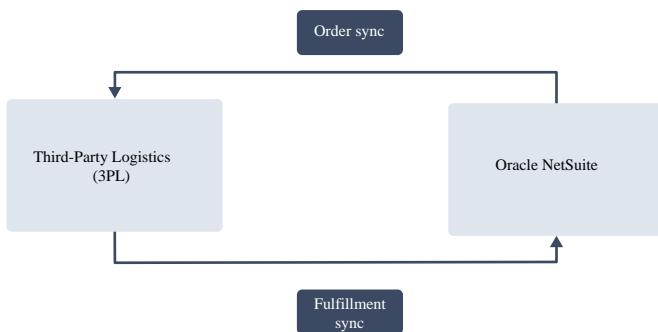
Order and Fulfillment Sync



Inventory Adjustment and Snapshot Sync



Transfer Order Sync



Note: Not all syncs are supported for all 3PLs.

Best Practices for NetSuite Connector Syncs

Best practices for NetSuite Connector syncs are based on experiences in configuring syncs for businesses or companies with diverse processes and workflows. These best practices are the most efficient ways to

achieve standard configurations of NetSuite Connector for integration projects and syncs. Best practices also reduce the likelihood of issues occurring. When issues do occur, they ensure that the issues will be resolved in the most efficient and focused manner.

The following topics discuss the best practices for NetSuite Connector syncs:

- Best Practice for Product Sync
- Best Practice for Order and Fulfillment Sync
- Best Practice for Refund Syncs
- Best Practice for 3PL Sync

Best Practice for Product Sync

Product sync applies to both Full Product Sync and Price and Quantity sync.

Flag Field

NetSuite Connector does not sync every item to the storefront with Product Sync. It uses a flag field on the item that allows you to indicate which items should sync to the storefront. This also lets you determine, by connector or account, which items will or will not sync. Therefore, there is flexibility to different sets of items for different connectors or accounts.

SKU Values

Storefronts identify items by an SKU value. NetSuite Connector uses this value as a unique identifier to match to an existing item in NetSuite, so the source and target item can be determined during product sync. A field on the NetSuite item record containing the SKU value on the item record is designated in the Product Mappings. When NetSuite Connector retrieves an item from NetSuite, it uses the value in the mapped SKU field to find a matching item in the storefront. This requires the SKU values to uniquely match, one-to-one, between the storefront and NetSuite.

Consider the following:

- The best practice is to use the **Item Name/Number** field as the SKU field.
- The NetSuite Connector requirement for unique SKUs applies to all storefronts even if the storefront allows duplicate SKU values.
- NetSuite SKU values must be unique across all item types and across both active and inactive items.
- SKU values are case-sensitive.
- If you are using Order Sync, the same SKU field must be used for both Order Sync and Product Sync, otherwise you would be syncing products that are not selling in orders.

Mapping Logic

All complex logic for determining the field values to sync to the storefront must be handled in NetSuite using a formula field, script, or some other automation. NetSuite Connector works best when mappings are relatively simple. NetSuite Connector either directly syncs data from NetSuite to the storefront or uses a simple translation mapping to produce an output for the storefront field. Constructing complex mapping logic directly in NetSuite Connector complicates troubleshooting, makes issues unclear, and renders some mappings only editable at the backend, which then requires you to rely on support for editing or even viewing your mapping. Keeping complex logic in NetSuite allows you to have full control and visibility into such logic. If an unexpected value is written to a field on the product in the storefront, determining where the unexpected data came from is greatly simplified.

Best Practice for Order and Fulfillment Sync

Order and fulfillments are closely related and so are their syncs.

For details of order and fulfillment syncs, see the following topics:

- [Best Practices for Order Sync](#)
- [Best Practice for Fulfillment Sync](#)

Best Practices for Order Sync

The following discusses the details of best practices for order syncs.

Customers

Customer Matching

When an order sync runs, the customer is synced first, so that the order can post against that customer. NetSuite Connector matches customers on orders to existing customer records in NetSuite by company name or name and email. This ensures that NetSuite Connector does not match to the incorrect customer record. If NetSuite Connector finds multiple customers that match by both company name and name and email, NetSuite Connector will match to the oldest (by date created) customer record. It is important to ensure that your company must not have duplicate customer records in NetSuite to avoid confusion regarding which customer NetSuite Connector will post against.

Address Forms

NetSuite Connector supports the use of only the default NetSuite address form when syncing customer address information. The default form provides the fields for the required address information and accept values that various storefronts provide for that information.

Taxes

Tax Rates

The best practice for taxes is to have NetSuite calculate the correct tax code and rate for an order based on that order's shipping address. Then, allow NetSuite Connector to overwrite the tax rate with the rate that was charged on the marketplace or cart. NetSuite does not allow explicitly overwriting the tax amounts, but only the rates. This is the reason that this is the best practice.

Tax rates in your marketplace or cart and in NetSuite on synced orders will always match.

There can be situations where your marketplace or cart charges a slightly different rate than NetSuite does, but even in this case, or if your marketplace or cart's calculation is incorrect, the order totals in NetSuite (including taxes, discounts, and others) should match what your customer paid. This is the reason that this is the leading practice.

Taxability

All customers, items, and shipping items should be assigned to a taxable tax schedule. This allows NetSuite Connector to tax or not tax orders, and the items in them, in a way that matches the order on the storefront. Whether an order or item should be taxable has already been determined by the time NetSuite Connector retrieves the order. Therefore, NetSuite Connector must have the ability to import the order as both taxable and non-taxable to match the storefront.

Marketplace Remittance

For marketplaces that remit tax on your behalf (Amazon, Walmart, and others) the best practice is for NetSuite Connector to omit the tax from the orders when they import to NetSuite. This functionality is

enabled by default. The tax is omitted because the marketplace never provides the tax revenue to you, but instead remits the tax directly to the tax authority on your behalf.

VAT

For storefronts that charge Value Added Tax (VAT), the best practice is for NetSuite Connector to subtract the VAT amount from the item before syncing the order since NetSuite will assign tax to the item upon import. If NetSuite Connector does not subtract the VAT from the item, then NetSuite will add an additional VAT to the item resulting in an overtaxing of the item.

Setup

For information about the best practice in setting up taxes in NetSuite and NetSuite Connector, see [Setting Up NetSuite Connector Order Sync](#).

Discounts

The best practice for discounts is to post all discount, promotion, or coupon amounts to a generic discount item. In most cases, the discount will be posted at the order header in NetSuite. But certain connectors will default to the line level because of specific differences for the storefront. Using this approach ensures that the discount total on the order will post to NetSuite resulting in the NetSuite order total matching the marketplace or cart order total. Posting the discount to the order header allows the discount total to show up in NetSuite order summary, on which you can easily view the discount on the order.



Note: When you install the NetSuite Connector, the generic discount item will be created for you and configured in NetSuite Connector automatically.

For information about the best practice for setting up discount items, see [Managing Order Discounts in NetSuite Connector](#).

Payments

Payment Methods

NetSuite Connector allows a translation mapping between the storefront payment methods and NetSuite payment methods. Then, your orders in NetSuite can have a payment method, and potentially, tokenization data.

Capturing Payment

The best practice is to capture payment in your marketplace or cart at the time of order (most are either preconfigured to do this and many do not have an option to do external capture). NetSuite Connector will import the tokenized credit card data on orders to NetSuite when such data is available in the storefront (if the mapped NetSuite payment method makes those fields available). The credit card data can later be used to refund orders in NetSuite. It data can also be used to capture payment in NetSuite if the state you sell in requires that you capture payment after fulfillment and not at the time of the order. The following information will be populated on the order:

- P/N Ref.
- Auth. Code
- Credit Card Approved - This field ensures payment that is not captured in NetSuite.



Note: This approach may not fully apply if the Payment Instruments feature is enabled in NetSuite. Changes with that feature require special handling.

Payment Processors or Gateways

NetSuite Connector does not integrate to payment gateways or processors, which means that any data related to the payment must arrive directly from the storefront. NetSuite Connector does not authorize or capture payments, it only syncs available payment data from the storefront to NetSuite. This allows you to use any payment processor or gateway in your storefront that the host platform supports.

Shipment Methods

The best practice for shipment methods is to map the storefront shipment method to a NetSuite shipment method so that the shipping cost and (if applicable) shipping tax can be properly accounted for on the synced order. Without a shipping method, NetSuite will not accept a shipping cost. If shipping was paid on the order in the storefront, but no shipping cost is imported to the order in NetSuite, then a variance in the order total between the storefront and NetSuite will result. For this reason, it is important to map a shipping method on your orders.

Shipping methods or items must be configured in a taxable tax schedule in NetSuite so that NetSuite Connector can assign a tax rate to the shipment method if the storefront charged tax on the shipping cost. Assigning a shipping method to nontaxable tax schedule creates a order total variance between your storefront and NetSuite when the storefront charges tax on the shipping cost. For this reason, it is important to place the shipping method or item in a taxable tax schedule.

For more information about setting up Shipping Methods or Items, see [Configuring Shipping for NetSuite Connector](#).

Mapping Logic

All complex logic for determining the field values on the order in NetSuite must be handled in NetSuite using a formula field, script, or some other automation. NetSuite Connector works best when mappings are relatively simple. NetSuite Connector either directly syncs data from the storefront to a field in NetSuite, or uses a simple translation mapping to produce a NetSuite field output. Constructing complex mapping logic directly in NetSuite Connector complicates troubleshooting, makes issues unclear, and renders some mappings only editable at the backend, which then requires you to rely on support for editing or even viewing your mapping. Keeping complex logic in NetSuite allows you to have full control and visibility into such logic. If an unexpected value is written to a field on the order in NetSuite, determining where the unexpected data came from is greatly simplified.

Order Items

SKU Field

Storefronts identify items on orders by an SKU value. NetSuite connector uses this value as a unique identifier to match to an existing item in NetSuite. A field on the NetSuite item record containing the SKU value is designated in the Order Settings (configurable by connector and account). NetSuite Connector searches NetSuite for an item record with a value in the designated SKU field that matches the SKU value on the item from the storefront. When it finds a match, it uses that matched item on the order in NetSuite. This requires the SKU values to uniquely match, one-to-one, between the storefront and NetSuite.

Consider the following:

- The best practice is to use the **Item Name/Number** field as the SKU field.
- The NetSuite Connector requirement for unique SKUs applies to all storefronts even if the storefront allows duplicate SKU values.
- NetSuite SKU values must be unique across all item types and across both active and inactive items.
- SKU values are case-sensitive.
- If you are using Product Sync, the same SKU field must be used for both Order Sync and Product Sync, otherwise you would be syncing products that are not selling in orders.

Price Level

The best practice is for NetSuite Connector to sync the price level on order items to the Custom price level in NetSuite. This allows NetSuite Connector to set the price of the item to the same price that was charged on the storefront regardless of what the price may be in NetSuite. This ensures that the tax for that item and the order total match the storefront.

Price, Quantity, and Tax

NetSuite Connector syncs the price, quantity, and tax rate that appears on the order in the storefront to the order in NetSuite. You must not modify any of these values as it alters the order total to create a variance, which can cause problems with other syncs like refund sync or settlement sync.

POS (Point of Sale) and FBA (Fulfilled By Amazon) Orders

The best practice for orders that are already paid and fulfilled when they are received by NetSuite Connector such as POS and FBA orders, is to sync them as Cash Sales. Since payment has already been collected and fulfillment has already taken place before the sync, no Sales Order or Item Fulfillment is necessary. Therefore, skipping to the Cash Sale record eliminates unnecessary records and work in NetSuite. Additionally, since customer information is typically not provided by the storefront with POS and FBA orders, a fixed customer is used for all such orders. NetSuite requires a customer for all orders and without customer data, neither existing customer match will be found nor can a new customer be created. Using a fixed customer allows NetSuite Connector to meet NetSuite's requirement that the order must have a customer without having customer data provided by the storefront.

Unit of Measure

NetSuite Connector does not sync the Unit of Measure for items because the storefront does not provide the unit of measure when the item is sold. The reason for this is that NetSuite has internal logic built around properly identifying the various unit of measure types on a particular item that should not, and often cannot, be overwritten during order sync. NetSuite will automatically populate the unit of measure on your line items on the order with the unit that you have specified on the item record for that particular item as the Primary Sale Unit type. Therefore, if you are using Units of Measure in NetSuite, it is important that you properly configure the item's Primary Sale Unit type with the the unit type you sell in your storefront. The Base Unit should be set for each item (which is essentially a single unit).

For example, if you sell water bottles in cases of 24, the base unit should be 'each' and the sale unit should be 'Case of 24'. This way, when you receive an order for 1 of the 24 case, your inventory will properly decrement 24 units. If your sales unit is incorrect and set to 'each' even though you sell cases; for example, when the order comes in, only one unit will be decremented leaving 23 single water bottles in stock that have already been sold.

Best Practice for Fulfillment Sync

All relevant fulfillment information such as tracking number and shipment method, automatically syncs by default when the Item Fulfillment for the order is set to a shipped status. For the tracking number, NetSuite Connector uses the NetSuite Tracking Number field on the Item Fulfillment as the source value. For the Carrier field sent to those storefronts that require it, NetSuite Connector derives the value from the shipment method name to determine the Carrier. This means it is important to provide the correct data on the Item Fulfillment record to allow NetSuite Connector to accurately sync the fulfillment information.

Best Practice for Refund Syncs

Because of the limitations or features of certain storefronts, or the way that NetSuite Connector integrates with those storefronts, the best practice for certain storefronts may differ from the general approach. If a connector has both workflows (in the diagrams) available in the Manage Data Sync page

in NetSuite Connector, then you must use the best practice workflow unless told otherwise by specialists or support personnel. Do not apply both workflows at the same time because this will cause issues. If a connector only has workflow available, then that is the best practice for that connector.

NetSuite Records

When syncing from NetSuite, NetSuite Connector will sync from Cash Refunds or Credit Memo records created from orders that it previously synced. This ensures that NetSuite Connector associates the refund with the correct order in the storefront.

When syncing refunds to NetSuite, the best practice is for NetSuite Connector to sync a Cash Refund or Credit Memo record only. Most storefronts only provide the refunded amount and, if applicable, the items refunded. Typically, NetSuite Connector will not receive the refund until the funds have really been refunded. Therefore, because the real refund has usually occurred, but whether the item has been returned, may or may not have necessarily occurred. Perhaps the return may even be expected to occur. NetSuite Connector has no source data to sync to NetSuite that accurately represent any other record type besides the Cash Refund or Credit Memo. Therefore, syncing only those records is the best practice. You can use NetSuite's Return Authorization and Item Receipt records to track the return of your products (if applicable) to accurately reflect that information.

Generic Refund Items

For storefronts that offer partial or nonitem-specific refunds, NetSuite Connector lets you designate specific items in NetSuite to account for the refunded amount on the Refund record. The best practice is to create Other Charge For Sale items to represent those partial or nonitem-specific refunds. Such items must NOT be set to fulfillable and in a taxable tax schedule. The latter enables NetSuite Connector to account for any tax amount that the storefront may have included in the refund.

Processing the Payment Refund

NetSuite Connector does not integrate to payment gateways or processors. This means the real refunding of the customer's payment is not done by NetSuite Connector. Refund records that NetSuite Connector creates in the storefront or (if capturing payment in NetSuite) in NetSuite may result in a refund being triggered (and often do). But NetSuite Connector does not initiate that process with any sort of communication to the payment processor or gateway. This lets you use any payment processor or gateway in your storefront that the host platform supports to capture and subsequently refund payments.

Best Practice for 3PL Sync

This topic covers best practices for the following in 3PL sync

- [Mapping Logic](#)
- [Actual Shipping Cost](#)
- [Inventory Detail](#)

Because some parts of 3PL sync are essentially Order and Fulfillment sync, for best practices related to 3PL order and fulfillment sync, refer to [Best Practice for Order and Fulfillment Sync](#).

Mapping Logic

All complex logic for determining the field values to sync to 3PL must be handled in NetSuite in a formula field, script, or other automation. NetSuite Connector works best when mappings are relatively simple, either directly syncing data from NetSuite to the 3PL or using a simple translation mapping. Keeping

complex logic in NetSuite allows you to have full control over the logic. If unexpected data is entered to a field on the order in the 3PL, determining where the unexpected data came from is greatly simplified.

Actual Shipping Cost

NetSuite Connector automatically syncs the actual cost paid to ship the order from the 3PL to the native shipping cost field on the Item Fulfillment. This syncing enables you to track the amount the customer paid for shipping on the Sales Order (incoming revenue) and the amount you paid to ship the order on the Item Fulfillment (outgoing revenue), as they are often not the same.

Inventory Detail

You must select the inventory detail on the source record (Sales Order, Transfer Order, etc.) so that NetSuite will automatically carry over the inventory detail to subsequent records that require it. If this is not possible, then you must configure NetSuite such that the inventory detail is automatically populated on the final record (Item Fulfillment, Item Receipt, Inventory Adjustment, or others) using a script or other ways. Most 3PLs do not send inventory detail in the data they return to NetSuite Connector, and while some do for certain data sets (like fulfillments), they do not with others (like inventory quantities). NetSuite should automatically provide the expected inventory detail. This ensures that the syncs related to inventory detail will work without error when NetSuite requests them, even if NetSuite Connector does not have the data to provide such detail. This also ensures that you will have the inventory detail you want on the final record.

 **Note:** When you select the inventory detail on the record, NetSuite automatically carries it over to subsequent records. Do not assign the inventory detail in NetSuite Connector.

Best Practices for Syncing Inactive Items in NetSuite Connector

If an item is marked as inactive in NetSuite, NetSuite Connector cannot see the item. Therefore, consider the following when marking an item inactive.

Order and Fulfillment Sync

After marking an item as inactive, if NetSuite Connector attempts to sync an order or a fulfillment that contains the item, you get an error. Because, when NetSuite Connector attempts to match the SKU in the order to an item in NetSuite, NetSuite Connector is unable to see the item. This results in NetSuite Connector not being able to point to the item in NetSuite and you get an order or fulfillment sync error.

Product Sync

When you change item fields, including storefront flag fields in NetSuite, NetSuite Connector is not aware of these changes instantly. There are processes that run at set intervals that provide the changes to NetSuite Connector. If you mark an item as inactive, any change to the item that was not updated to NetSuite Connector will never be known to NetSuite Connector.

For example, you change the storefront flag field on an item to **Remove Item** or **N** and immediately set the item as inactive in NetSuite. Then, there is a possibility that NetSuite Connector does not see the change of the storefront flag field. Also, if there is a pending sync to the marketplace or cart for the item in NetSuite Connector, the sync attempt will still occur.

Handling Marking Items Inactive

Mark items as inactive only when needed. If you must mark the items inactive, make sure you do the following:

- For order sync, make sure all syncing orders that may contain the inactive item have been marked with a **Complete** status in NetSuite Connector.
- For product sync, change the storefront flag field on the item to **Remove Item** or **N** for each marketplace or cart that you are syncing to.
For more information about storefront flag field, read [NetSuite Connector Storefront Flag Fields](#).
- For full product sync type of product sync, on the next sync, the item will be removed from syncing with the marketplace or cart.
- For price and quantity sync type of product sync, remove the item manually from each syncing marketplace or cart.
- Wait for two hours from the completion of the previous steps and then mark the item as inactive in NetSuite.

Sync Frequency and Directional Flows

Use the following table to determine the frequency and directional flow of the NetSuite Connector syncs:

Sync Name	Sync Source	Sync Target	Default Frequency
Order Sync	Storefront	NetSuite	20 minutes
Fulfillment Sync	NetSuite	Storefront	90 minutes
3PL Sync- Orders	NetSuite	3PL	20 minutes
3PL Sync- Fulfillments	3PL	NetSuite	90 minutes
Product Sync	NetSuite	Storefront	60 minutes
Refund Sync	(directional flow varies by storefront)	(directional flow varies by storefront)	90 minutes
Amazon Settlement Sync	Amazon	NetSuite	One time a day
Amazon Inventory Adjustment Sync	Amazon	NetSuite	One time a day
Amazon Inbound Shipping Sync	Amazon or NetSuite	NetSuite or Amazon	20 minutes

To determine the frequency of your specific sync, read the help topic [Viewing the Date, Time, and Frequency of Syncs](#).

Number of Logic Levels Supported in NetSuite Connector Mappings

NetSuite Connector mappings support two levels of logic. For example, the following logic is supported:

- Check If **myField** on an Order has the value **A**.
- If yes, then check if **myField2** has the value **B**.
- If yes, output **123** and if not, output **456**.

More than two levels of logic like in the example below is not supported:

- Check If **myField** On an Order has value **A**.

- If yes, then check if **myField2** has the value **B**.
- If **myField2** is not **B**, then output **789**.
- If **myField2** is **B**, then check if **myField3** has the value **C**.
- If yes, output **123** and if not, output **456**.

Even though you can construct more than two levels of logic in the UI mapper, you should limit your mapping to two levels of logic. Any level beyond that may not work.

NetSuite Connector Third Party Licensing

A NetSuite Connector implementation can include software governed by licenses from third parties ("Third Party Software" and "Third Party License"). All third party software licensed for use with NetSuite Connector is subject to the terms and conditions of the corresponding Third Party License, notwithstanding anything to the contrary in the agreement governing the NetSuite Connector Software. See the following PDF for specific information.

NetSuite Connector makes no representation or warranty concerning Third Party Software and shall have no obligation or liability with respect to Third Party Software.

 [NetSuite Connector Third Party Licenses.](#)

NetSuite Connector Setup

This chapter gets you started with the setup of NetSuite Connector using the following sections:

- [NetSuite Connector Prerequisites](#)
- [NetSuite Connector IP Addresses to Safelist \(Allowlist\)](#)
- [Creating a NetSuite Connector Account](#)
- [Setting Up Token-Based Authentication for NetSuite Connector](#)
- [Migrating to the NetSuite Connector SuiteApp](#)
- [Setting Up NetSuite Connector Order Sync](#)
- [Setting Up NetSuite Connector Product Sync](#)
- [Setting Up NetSuite for NetSuite Connector Syncs](#)
- [Setting Up NetSuite Connector for Syncs](#)
- [Managing Connector and Account Settings in NetSuite Connector](#)

NetSuite Connector Prerequisites

Before you begin to activate NetSuite Connector, you must first meet the following prerequisites. You need to complete each prerequisite to avoid potential errors with your NetSuite Connector account.

General Prerequisites

- You must have an account for NetSuite, your marketplace or cart, and third-party logistics (3PL) service, if applicable. You must be able to successfully log in to each of these accounts.
- You need to be familiar with using NetSuite, your marketplace or cart, and 3PL, if applicable.

Prerequisites for NetSuite

- You should be able to manually enter orders into NetSuite. The record should include the shipment and payment methods and any other information you expect NetSuite Connector to post, such as discounts.
- You should configure NetSuite to correctly calculate your tax codes and rates. Specifically, your tax nexuses must be configured so that orders entered into NetSuite will properly reflect the tax.
- You need to manually create a taxable order in NetSuite to verify that tax is assigned properly to your items.
- You should be able to fulfill orders, print labels, generate tracking numbers, and do other actions in NetSuite or in your 3PL as applicable.
- You should be able to bill orders in NetSuite.

Prerequisites for Marketplace or Cart

- All items you sell in your marketplace or cart must be in NetSuite. The stock-keeping units (SKUs) must exactly match 1 to 1 from the marketplace or cart to NetSuite and the item hierarchy.
- You must identify what data will be synced between NetSuite and the marketplace or cart. For example, order information, SKUs, quantity, price, and product description, and other relevant information must be synced.
- You should be able to place orders on your marketplace or cart because you will create test orders for NetSuite Connector during integration.

Prerequisites for NetSuite Connector

- Make sure you have signed up on the app.farapp.com website.
- You should be able to supply NetSuite Connector with sample or test data as needed. For example, if NetSuite Connector requests sample orders to manually sync to NetSuite for you as tests, you should be able to provide the order IDs for these orders. The most common test data needed are orders, fulfilled orders, products ready to sync, and refunded orders.
- If you are setting up the product sync, you should be able to create the item both in NetSuite and in the marketplace or cart with the correct data.
- If you are setting up the refund sync, you should be able to refund the order both in NetSuite and in the marketplace or cart with the correct data.

Enabling Features

Enable features from the Enable Features page.

To enable features:

1. Go to Setup > Enable Features.
2. Enable the features listed in the following table:

Subtab	Feature
CRM	Customer Relationship Management
Items & Inventory	Inventory
SuiteCloud	<ul style="list-style-type: none"> ■ Custom Records ■ Client SuiteScript ■ Server SuiteScript ■ SOAP Web Services ■ Token-Based Authentication

3. Click **Save**.

NetSuite Connector IP Addresses to Safelist (Allowlist)



Important: Oracle NetSuite does not support the use of NetSuite IP addresses to access or manage access to any NetSuite services. There are better alternatives than using a list of allowed IP addresses to manage access to NetSuite. Using IP addresses (either as part of a URL or in your Domain Name Server, or DNS) to access NetSuite services prevents dynamic DNS routing. Understand the following:

- The IP addresses of NetSuite services may change at any time without notice.
- Using IP addresses to directly access NetSuite services can result in unpredictable service outages or significant performance degradation.

When using a cart, server, or other service that is behind a firewall, you may need to add NetSuite Connector IP Addresses to your firewall's safelist or allowlist.

If necessary, include the following IP addresses:

- 158.101.44.97
- 158.101.27.21
- 141.148.149.162
- 52.10.94.111
- 50.112.172.4
- 35.161.153.181
- 44.224.106.183
- 192.241.198.156

Including the above IP addresses to your safelist will enable you to connect to any NetSuite Connector service.

i Note: If you view any of the following issues for a connector, the reason most likely is that the IP addresses are blocked:

- Long delays before a failing test or timed out test.
- 403 Forbidden error.

Creating a NetSuite Connector Account

If you have not implemented NetSuite Connector, contact NetSuite sales for the same. If you have an old NetSuite Connector account that was deactivated, contact NetSuite Customer Support to reactivate your account.

To create a NetSuite Connector account:

1. Go to <https://app.farapp.com/>.
2. Click **Sign Up**.
3. On the Sign Up page, enter or select values in the required fields.
4. Read the subscriber agreement and click **I Agree**.
5. Click **Submit**.

You should receive an activation email in the email address you provided on the Sign Up page.

6. Check your inbox for an email message from NetSuite Connector.
7. Follow the link provided in the email to activate your NetSuite Connector subscription.

You should now be logged in to your newly activated NetSuite Connector account and see the NetSuite Connector dashboard.

In the dashboard, you will be prompted to add an ERP connector.

8. Select NetSuite or SKUVault to add the ERP connector.



Note: If you want to remove the added ERP connector, contact NetSuite Customer Support.

Adding a User to a NetSuite Connector Account

With the implementation of Two-Factor Authentication (2FA), you may find it difficult to share accounts with other users. Follow this procedure to add a new user who needs access to your account.



Note: You must be logged in as the Primary user to add, edit or delete users.

To add a user to a NetSuite Connector account:

1. Log in to app.farapp.com.
2. On the top-right corner of the page, hover over your account name to show the dropdown list. Then, select **Account Settings**.
3. Click the **Manage Users** tab.
4. Click **Add New User**.
5. Fill out the following fields:
 - Username
 - Email
 - Full Name
 - Password
 - Verify Password
6. Click **Save User**.

The new user appears on the list.

Changing the NetSuite Connector Password in the Account Settings

Your NetSuite Connector password enables you to log in to your account, and the system also uses it in other aspects of your syncs.

If you forgot your password and cannot log in, go to app.farapp.com. To reset your password, use the **Forgot your password?** link and follow the instructions.

Alternatively, you can also change your password in the account settings.

To change your NetSuite Connector password in the account settings:

1. Log in to app.farapp.com.
2. On the top-right corner of the page, hover over your account name to show the dropdown list. Then, select **Account Settings**.
3. Go to the **Change Password** tab.
4. Enter the value for the following fields:
 - Current Password
 - New Password
 - Confirm New Password
5. Click **Change Password**.

A confirmation message appears, which indicates that you successfully updated your password.

If you change your password, you must also update the password in the secret ID created for accessing NetSuite Connector. For more information, read the help topic [Creating Secrets](#).

Resetting the NetSuite Connector Password

There are two ways to change your NetSuite Connector password. The first is through the **Forgot your password?** link, and the second is through the change password option in the account settings.

If you forgot your password and cannot log in, follow the instructions for resetting your password.

To reset your NetSuite Connector password using the **Forgot your password?** link:

1. Go to app.farapp.com.
2. Click **Forgot your password?** link.
3. Enter the email address for your NetSuite Connector account.
4. Click **Reset Password**.

A popup message appears, which indicates that an email was sent to your email address.

5. Check your inbox and follow the instructions in the email about resetting your password.

To change your NetSuite Connector password in the account settings:

1. Log in to app.farapp.com.
2. On the top-right corner of the page, hover over your account name to show the dropdown list. Then, select **Account Settings**.
3. Go to the **Change Password** tab.
4. Enter the value for the following fields:
 - Current Password
 - New Password
 - Confirm New Password
5. Click **Change Password**.

A confirmation message appears, which indicates that you successfully updated your password.

Notification and Email Settings for NetSuite Connector

You can manage various notifications you receive from NetSuite Connector in the following areas:

- For NetSuite and general/billing notifications, go to NetSuite. In the NetSuite Connector menu, select Settings > Notification and configure the settings as needed.
- For notifications to a particular connector, go to NetSuite Connector. Select the connector, then click **Notifications**.

NetSuite Connector sends the notification only when your account is currently processing that type of feed.

For example, if you do not set up NetSuite Connector to process custom record updates for NetSuite, setting the notification will have no effect. On the Notifications page for NetSuite, even if you enter your email address in the **Send NetSuite Custom Record Update Error Notifications To** field, you will not receive any notifications.

Setting Up Token-Based Authentication for NetSuite Connector

NetSuite Connector connects to your NetSuite account using token-based authentication. With the proper setup, you can generate a token and enter the generated token in NetSuite Connector.

To set up token-based authentication:

1. Create the NetSuite access token and connect NetSuite Connector.
 - a. Verify the installation of NetSuite Connector integration record
 - i. Go to Setup > Integration > Manage Integrations.
NetSuite Connector should be listed on the Integrations page.
 - ii. (Optional) If NetSuite Connector is not listed, the integration must be blocked. Follow these steps to enable a blocked integration:
 - A. On the Integrations page, check the **Show Inactives** box.
 - B. Click **Refresh**.
 - C. If you still do not see the NetSuite Connector record, try to uninstall, and reinstall the SuiteApp.
 - D. When the NetSuite Connector record displays in the Integrations page, click **NetSuite Connector**.
The NetSuite Connector record opens.
 - E. Click **Edit**.
 - F. From the **State** list, select **Enabled**.
 - G. Click **Save**.
2. Assign the role for token authentication.
 - a. Go to Setup > Users/Roles > Manage Users.
 - b. Click the employee name to which you want to assign the token-authentication role.
 - c. On the Employee page, click **Edit**.
 - d. Click the **Access** subtab.
 - e. In the **Role** field, select **NetSuite Connector Web Services** and click **Add**.
 - f. Click **Save**.
3. Verify access of the assigned role to subsidiaries.
 - a. Go to Setup > Users/Roles > Manage Roles.
 - b. Click the **Edit** link of the NetSuite Connector Web Services role.
 - c. Under the Subsidiary Restrictions section, check the **Allow Cross-Subsidiary Record Viewing** box.



Note: The employee that you select must have the Administrator role assigned.
Also, to avoid breaking the connection, you must not delete or deactivate the user ID.

- d. Click **Save**.
4. Create token authentication credentials for NetSuite Connector.
 - a. Go to Setup > Users/Roles > Access Tokens > New.
 - b. On the Access Token setup page, in the **Application** field, select **NetSuite Connector**. This references the integration record.
 - c. In the **User** field, select the user you assigned the NetSuite Connector Web Services role.
 - d. In the **Role** field, select **NetSuite Connector Web Services**.

The Token Name is automatically filled out.

- e. Click **Save**.

A confirmation page is displayed indicating the Token ID and Token Secret. Copy the Token ID and Token Secret values and keep them for later use.



Important: This is the only time that the Token ID and Token Secret values will be displayed. When you close the page, you will not be able to retrieve or have access to the values anymore. If you lose or forget these credentials, you must create a new token.

After generating the token, do not delete or deactivate the user with the token authentication role. If the user account associated with a token is deleted or deactivated, the token will not work anymore and any connection that uses that token will fail.

Adding NetSuite Account ID to NetSuite Connector

The Account ID is the first part of the URL in the browser when logged into NetSuite. Alternatively, in NetSuite, you can find the ID in the **Account ID** field in the Company Information page, accessible from Setup > Company > Company Information.

To add NetSuite Account ID to NetSuite Connector:

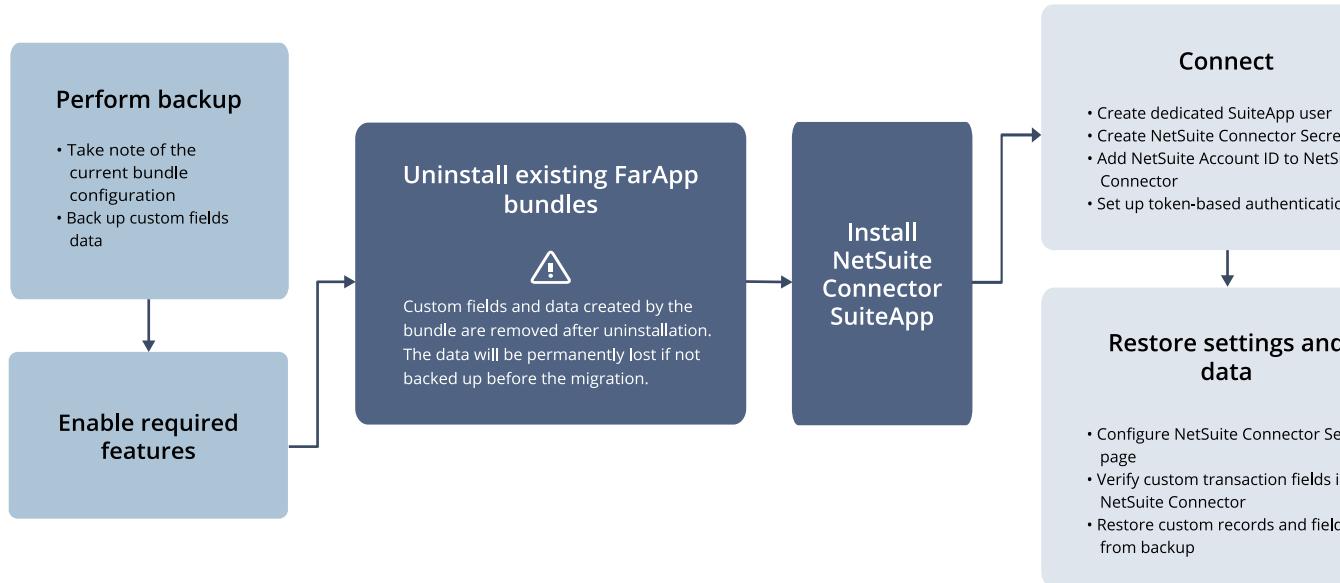
1. Log in to app.farapp.com.
2. Go to NetSuite > Settings > Credentials.
3. On the Credentials settings page, in the **NetSuite Account ID** field, enter the NetSuite Account ID. You can find your NetSuite account ID at the beginning of the NetSuite URL. For example, if the URL is <https://1234567.app.netsuite.com/>, your account ID is 1234567.
4. In the **NetSuite Token ID** field, enter the value of the **Token ID** you generated and copied.
5. In the **NetSuite Token Secret** field, enter the value of the **Token Secret** you generated and copied.
6. Click **Save and Test Connection**.

If the testing is successful, a confirmation message appears. If the testing fails, check the values you entered and retry.

Migrating to the NetSuite Connector SuiteApp

If you are using the old FarApp or NetSuite Connector bundles, you must upgrade to the latest NetSuite Connector SuiteApp.

✖ Warning: During migration to the NetSuite Connector SuiteApp, none of the syncs will run and no data will sync.



Preparing for Migration

Before starting the migration, make the following preparation:

Task	Help Topic
Create a dedicated SuiteApp user.	Creating a Dedicated SuiteApp User
Check the current bundle configuration settings.	Checking the Current Bundle Configuration Settings
Check fields used in business processes and automations.	Checking Fields Used in Business Processes and Automations
Take backup of existing field data.	Taking Backup of Existing Field Data

Creating a Dedicated SuiteApp User

Creating a dedicated SuiteApp user is a leading practice for NetSuite Connector.

To create a dedicated user:

- Log in to app.farapp.com as a Primary user.
- On the top-right side, hover over your profile name and select **Account Settings**.
- Click the **Manage Users** tab.

4. Click **Add New User**.
5. In the Add New User popup window:
 - a. In the **Login Name** field, enter a login name that is unique across all NetSuite Connector accounts.
To make the name unique, you should include your company name in the login name, without any space.
 - b. In the **Email** field, enter an email address that is unique across all NetSuite Connector accounts.
This email address should not be of a user who may now or in future use NetSuite Connector. You do not need to access any emails sent to this user. Therefore, you can use something like dedicated@<yourdomain>. For example, dedicated@mucompany.com.
 - c. In the **Full Name** field, enter a unique full name for the user.
For example, **NSC SuiteApp Dedicated User - DO NOT DELETE**. Such username clarifies what the user is and that nobody should delete it.
 - d. In the **Password** and **Verify Password** fields, enter a password that contains:
 - At least 8 characters
 - An uppercase, a lower case, and a number
 - A special character
 - e. Click **Save User**.



Important: Do not enable multi-factor authentication for this user.

Watch the following help video for a demonstration on how to create a dedicated SuiteApp user in NetSuite Connector.

[NetSuite Connector: Creating a Dedicated SuiteApp User](#)

Checking the Current Bundle Configuration Settings

If the FarApp Token Based Authentication bundle or NetSuite Connector Token Based Authentication bundle is the only FarApp or NetSuite Connector bundle installed in your account, you can skip this section.

To check the bundle configuration settings:

1. Go to Setup > Company > General Preferences.
2. Click the **Custom Preferences** subtab.
3. In the **NetSuite Connector** section:
 - a. If **Disable Real-Time Sync** box is checked, note this status.
 - b. If the **Ensure Line-Item Tax Rates Match Created From Transactions (Beta)** box is checked, note this status.

Checking Fields Used in Business Processes and Automations

When you uninstall the FarApp or NetSuite Connector bundles, any fields installed by the bundles are deleted along with the data in those fields. If you use any custom fields in your orders or other business processes, make sure those fields are not created by one of the uninstalled bundles.

To check the bundle from which a field is created:

1. Perform one of the following actions:
 - To check transaction body fields, go to Customization > Lists, Records & Fields > Transaction Body Fields.
 - To check transaction line fields, go to Customization > Lists, Records & Fields > Transaction Line Fields.
2. On list page, the **From Bundle** column populates the bundle ID from which the field is created.

Taking Backup of Existing Field Data

If you have business processes or automations that use fields created from one of the FarApp or NetSuite Connector bundles, backup the data of such fields. You should create new fields and move the data to the new fields with appropriate adjustments to processes, automations, or mappings. With this method, you can verify that all data is as expected before losing the original data during uninstallation.



Important: NetSuite Connector does not update data in closed accounting periods. During migration, you should not allow general ledger changes to closed accounting periods to limit the necessary data backup to only open accounting periods. To do this, remove the Manage Accounting Periods permissions from the FarApp or NetSuite Connector Web Services role.

You can take backup by exporting the custom records or fields to a CSV file. When uninstalling the bundle, you need to back up the following fields:

- Document Number
- FarApp Line Item Amount (Custom Column)
- FarApp Line Item Tax (Custom Column)
- FarApp Marketplace/Cart (Custom Body)
- FarApp Marketplace/Cart Order Number (Custom Body)
- FarApp Order Total (Custom Body)
- FarApp Shipping Tax (Custom Body)

For more information, read the help topic [Extracting Data from NetSuite](#).

Watch the following help video for a demonstration on how to back up NetSuite Connector custom field data.

[Exporting NetSuite Connector Custom Field Data](#)

Migrating to the NetSuite Connector SuiteApp

Note the following points before you start the migration to the NetSuite Connector SuiteApp:

- The estimated duration of performing this procedure is 20 minutes.
- To get timely support from NetSuite Customer Support, you should perform the migration between 8 a.m. to 6 p.m., Pacific Standard Time (PST) on working days.
- Uninstalling a bundle deletes the custom record types or fields, token-based authentication, and any other setup done by the uninstalled bundles. The procedures in this migration process enable you to set these features again.
- When you save a record in NetSuite during migration, an error may appear indicating that the SuiteApp is not configured. However, when the migration is complete, the error disappears automatically.

- After the migration is completed and NetSuite Connector connects to NetSuite, NetSuite Connector initiates a series of automated processes. These processes set up your account and then resync NetSuite Connector with NetSuite. The resync takes place in the backend and takes some time.

Refer to the following table for steps to migrate and set up the NetSuite Connector SuiteApp.

Step	Task	Help Topic
1.	Enable the required features.	Enabling the Required Features
2.	Create NetSuite Connector Secret ID to access NetSuite Connector.	Creating NetSuite Connector Secret ID
3.	Uninstall existing FarApp or NetSuite Connector bundles.	Uninstalling the Existing FarApp or NetSuite Connector Bundles
4.	Install the NetSuite Connector SuiteApp.	Installing the NetSuite Connector SuiteApp
5.	Configure the NetSuite Connector Setup page.	Configuring the NetSuite Connector Setup Page
6.	Set up token-based authentication for NetSuite Connector.	Setting Up Token-Based Authentication for NetSuite Connector
7.	Add NetSuite account ID to NetSuite Connector.	Adding NetSuite Account ID to NetSuite Connector
8.	Set the Custom Transaction Fields in NetSuite Connector.	Verifying and Setting the Custom Transaction Fields in NetSuite Connector
9.	Restore custom records and fields.	Restoring Custom Records and Fields

Enabling the Required Features

Enable the features required for using NetSuite Connector.

To enable the required features:

- Go to Setup > Enable Features.
- Enable the features listed in the following table:

Subtab	Feature
CRM	Customer Relationship Management
Items & Inventory	Inventory
SuiteCloud	<ul style="list-style-type: none"> Custom Records Client SuiteScript Server SuiteScript REST Web Services SOAP Web Services Token-Based Authentication SuiteFlow

- Click **Save**.

Creating NetSuite Connector Secret ID

Create NetSuite Connector Secret ID to access NetSuite Connector.

To Create NetSuite Connector Secret ID:

1. Go to Setup > Company > API Secrets.
2. Click **Create New**.
3. In the Create New Secret popup window, enter a name for the API secret.
4. In the **ID** field, enter an ID for the secret.
You will have to enter this ID in the **NetSuite Secret ID** field of the NetSuite Connector Setup page.
5. In the **Password** and **Confirm Password** fields, enter the password for accessing NetSuite Connector.
If you created a user using the procedure in the section [Creating a Dedicated SuiteApp User](#), enter the password set for that user.
6. Click the **Restrictions** tab.
7. Check the **Allow for All Scripts** box.
8. Check the **Allow for All Domains** box.
9. Click **Save**.
10. Refresh the page to view the new secret ID and note it.

You will need this secret ID when configuring the NetSuite Connector SuiteApp.

Watch the following help video for a demonstration on how to create a secret ID for NetSuite Connector.



[Creating a NetSuite Connector Secret ID](#)

Uninstalling the Existing FarApp or NetSuite Connector Bundles

Uninstall the bundles listed in this procedure.



Warning: Uninstalling a bundle deletes the custom record types or fields, token-based authentication, and any other setup done by the uninstalled bundles.

To uninstall the existing FarApp or NetSuite Connector bundles:

1. Go to Customization > SuiteBundler > Search & Install Bundles > List.
2. In the Installed Bundles page, if you have any of the following bundles listed, select **Uninstall** from the **Action** dropdown list for those bundles:

Bundle ID	Bundle Name
401685	NetSuite Connector
402043	NetSuite Connector Marketplace/Cart
401819	NetSuite Connector for SuiteSuccess
283676	FarApp
123322	
283678 (Managed)	FarApp for SuiteSuccess - Item Fields
135118 (Unmanaged)	
283682 (Managed)	FarApp Marketplace/Cart - Item Fields
173319 (Unmanaged)	
283691 (Managed)	FarApp Token-Based Authentication

Bundle ID	Bundle Name
208429 (Unmanaged)	



Note: Do not uninstall any of the following bundles:

- FarApp Saved Search Export
- NetSuite Connector Saved Search Export (bundle ID – 402066)

3. In the confirmation popup window, click **OK**.
4. After couple of minutes, click **Refresh** to confirm that the bundle is uninstalled.



Important: A role that cannot be deleted can cause the uninstallation process to fail. If this occurs, the error states that the record cannot be deleted because it has dependent records. To resolve this, change the internal ID of the role specified in the error message. Then, run the uninstallation process again.

5. Repeat steps **2** to **4** for each FarApp or NetSuite Connector bundle.

Watch the following help video for a demonstration on how to uninstall the previous NetSuite Connector version.



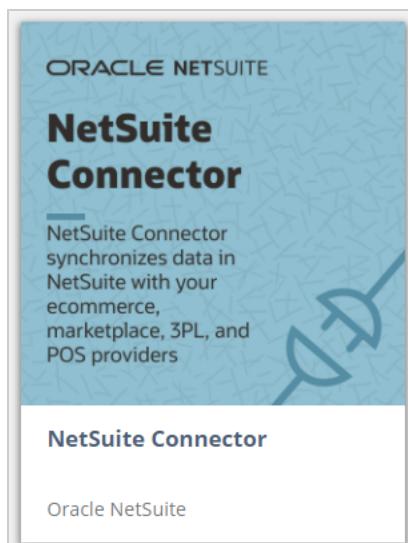
[Uninstalling the NetSuite Connector Bundles](#)

Installing the NetSuite Connector SuiteApp

Install the NetSuite Connector SuiteApp from the SuiteApp marketplace.

To install the NetSuite Connector SuiteApp:

1. Click the **SuiteApps** tab.
2. On the SuiteApp marketplace, in the **Search** field, enter **NetSuite Connector**, and press Enter. The SuiteApps are filtered to show the NetSuite Connector SuiteApp tile.
3. Click the **NetSuite Connector** tile as shown below:



4. On the NetSuite Connector page, click **Install**.
5. In the confirmation popup window, click **Install**.

The SuiteApp installation may take some time. Wait until the SuiteApp shows the status as **Installed** in the top-right area of the page. For more information, read the help topic [Installing from the SuiteApp Marketplace](#).

Watch the following help video for a demonstration on how to install the NetSuite Connector SuiteApp.



Configuring the NetSuite Connector Setup Page

The NetSuite Connector SuiteApp creates the NetSuite Connector Setup page. Use the following procedure to configure this page.

To configure the NetSuite Connector Setup page:

1. Go to Connector > Configuration > Setup.
2. On the NetSuite Connector Setup page, configure the following fields:

Field	Description
Username	If you created a user using the procedure in the section Creating a Dedicated SuiteApp User , enter the login name of that user.
Secret ID	Enter the secret ID you created in the section Creating NetSuite Connector Secret ID for accessing NetSuite Connector.
Disable Real-Time Sync	If the old FarApp or NetSuite Connector bundle had this box checked, then check this box.
Match Line-Item Tax Rates with Tax Rates Created from Transaction	If the old FarApp or NetSuite Connector bundle had the Ensure Line-Item Tax Rates Match Created From Transactions (Beta) box checked, then check this box.

3. Click **Submit**.

Watch the following help video for a demonstration on how to configure NetSuite Connector SuiteApp setup page.



Setting Up Token-Based Authentication for NetSuite Connector

You must connect your NetSuite account to NetSuite Connector using token-based authentication. With the proper setup, you can generate a token and connect to NetSuite Connector.

To set up token-based authentication:

1. Create the NetSuite access token and connect NetSuite Connector.
 - a. Verify the installation of NetSuite Connector integration record
 - i. Go to Setup > Integration > Manage Integrations.
NetSuite Connector should be listed on the Integrations page.
 - ii. (Optional) If NetSuite Connector is not listed, the integration must be blocked. Follow these steps to enable a blocked integration:
 - A. On the Integrations page, check the **Show Inactives** box.

- B. Click **Refresh**.
 - C. If you still do not see the NetSuite Connector record, try to uninstall, and reinstall the SuiteApp.
 - D. When the NetSuite Connector record displays in the Integrations page, click **NetSuite Connector**.
The NetSuite Connector record opens.
 - E. Click **Edit**.
 - F. From the **State** list, select **Enabled**.
 - G. Click **Save**.
2. Assign the role for token authentication.
- a. Go to Setup > Users/Roles > Manage Users.
 - b. Click the employee name to which you want to assign the token-authentication role.
- i Note:** The employee that you select must have the Administrator role assigned. Also, to avoid breaking the connection, you must not delete or inactivate the user ID.
- c. On the Employee page, click **Edit**.
 - d. Click the **Access** subtab.
 - e. In the **Role** field, select **NetSuite Connector Web Services** and click **Add**.
 - f. Click **Save**.
3. Verify access of the assigned role to subsidiaries.
- a. Go to Setup > Users/Roles > Manage Roles.
 - b. Click the **Edit** link of the NetSuite Connector Web Services role.
 - c. Under the Subsidiary Restrictions section, check the **Allow Cross-Subsidiary Record Viewing** box.
 - d. Click **Save**.
4. Create token authentication credentials for NetSuite Connector.
- a. Go to Setup > Users/Roles > Access Tokens > New.
 - b. On the Access Token setup page, in the **Application** field, select **NetSuite Connector**. This references the integration record.
 - c. In the **User** field, select the user you assigned the NetSuite Connector Web Services role.
 - d. In the **Role** field, select **NetSuite Connector Web Services**.
- The Token Name is automatically filled out.
- e. Click **Save**.
- A confirmation page is displayed indicating the Token ID and Token Secret. Copy the Token ID and Token Secret values and keep them for later use.

! **Important:** This is the only time that the Token ID and Token Secret values will be displayed. When you close the page, you will not be able to retrieve or have access to the values anymore. If you lose or forget these credentials, you must create a new token.

After generating the token, do not delete or inactivate the user with the token authentication role. If the user account associated with a token is deleted or inactivated, the token will not work anymore and any connection that uses that token will fail.

Watch the following help video for a demonstration on how to set up token-based authentication in NetSuite Connector.



Adding NetSuite Account ID to NetSuite Connector

The Account ID is the first part of the URL in the browser when logged into NetSuite. Alternatively, in NetSuite, you can find the ID in the **Account ID** field in the Company Information page, accessible from Setup > Company > Company Information.

To add NetSuite Account ID to NetSuite Connector:

1. Log in to app.farapp.com.
2. Go to NetSuite > Settings > Credentials.
3. On the Credentials settings page, in the **NetSuite Account ID** field, enter the NetSuite Account ID. You can find your NetSuite account ID at the beginning of the NetSuite URL. For example, if the URL is <https://1234567.app.netsuite.com/>, your account ID is 1234567.
4. In the **NetSuite Token ID** field, enter the value of the **Token ID** you generated and copied.
5. In the **NetSuite Token Secret** field, enter the value of the **Token Secret** you generated and copied.
6. Click **Save and Test Connection**.

If the testing is successful, a confirmation message appears. If the testing fails, check the values you entered and retry.

Verifying and Setting the Custom Transaction Fields in NetSuite Connector

Use this procedure to verify and if required, set the custom transaction fields in NetSuite Connector.

To verify and set the custom transaction fields in NetSuite Connector:

1. Log in to app.farapp.com.
2. Go to NetSuite > Settings > General.
3. On the NetSuite Settings page, click **Advanced Options**.
4. If the field **Custom Transaction Body Storefront Field** does not contain the value **custbody_fa_channel**, enter that value in the field.
5. If the field **Custom Transaction Body Storefront Order Field** does not contain the value **custbody_fa_channel_order**, enter that value in the field.
6. Click **Save**.

Restoring Custom Records and Fields

If you had taken the backup of custom records and fields, restore those custom record and fields. Import the CSV file that you created when taking the backup.

The custom fields from the backup file needs to be imported and mapped to the new custom fields created by the NetSuite Connector SuiteApp.

Fields from Backup	NetSuite Fields
Document Number	Sales Order : Order #
FarApp Line Item Amount	Sales Order - Items : NetSuite Connector Line Item Amount
FarApp Line Item Tax	Sales Order - Items : NetSuite Connector Line Item Tax
FarApp Marketplace/Cart	Sales Order : NetSuite Connector Storefront
FarApp Marketplace/Cart Order Number	Sales Order : NetSuite Connector Storefront Order Number
FarApp Order Total	Sales Order : NetSuite Connector Order Total
FarApp Shipping Tax	Sales Order : NetSuite Connector Shipping Tax

For more information about importing CSV files, read the help topic [Importing CSV Files with the Import Assistant](#).

The time of processing and the amount of records can affect the import job. If your import job is processing slowly, see the help topic [Why is my import job processing so slowly?](#).

Watch the following help video for a demonstration on how to restore NetSuite Connector custom field data from backup.



Setting Up NetSuite Connector Order Sync

You must complete the following procedures before using NetSuite Connector order sync or fulfillment sync:

1. Enable the **Show Internal IDs** box in the Set Preferences page. For more information, read the help topic [Setting the Show Internal IDs Preference](#).
2. [Configuring Taxes](#)
3. [Configuring Shipping Methods](#)
4. [Configuring Payment Methods](#)

Configuring Taxes

Configure taxes for the United States in NetSuite. For other countries, keep the default setup.

To configure taxes:

1. Go to Setup > Accounting > Set Up Taxes.
2. (Optional) If you have tax set up for multiple countries, select the **United States** subtab.
3. Check the following boxes:
 - Enable Tax Lookup on Sales Transactions
 - Customers Default to Taxable

For more information, read the help topic [Tax Setting for Customers](#).

If you have existing customers in NetSuite, make sure they are set to taxable.
 - Respect Discount Item Tax Preference.

- Verify that **Charge Sales Tax on Store Orders** is set to **Always**.

Note: The above tax settings do not indicate that tax will always be charged to items or orders. The above settings only enable NetSuite Connector to assign sales tax when the marketplace or cart assigns tax for an order.

- Click **Save**.

Note: Any item that may be charged tax on marketplace or cart must be set to use a taxable tax schedule in NetSuite. For more information, read the help topic [Taxable Items](#).

Note: If you use VAT in your marketplace or cart, read [Configuring VAT in NetSuite Connector](#).

Configuring Shipping Methods

Configure shipping methods or shipping items in NetSuite to map storefront shipping methods to the corresponding shipping methods in NetSuite. Mapping shipping methods is important because without a shipping method on an order, NetSuite does not accept shipping cost.

Note: NetSuite may provide several default shipping items that are not completely set up. Therefore, even though these shipping items appear in the list, you cannot use them. You can edit and save a shipment item in NetSuite before using the shipment item with order sync.

For more information, read [Configuring Shipping for NetSuite Connector](#).

Configuring Payment Methods

If you do not have payment method set on an order, some NetSuite operations can be impacted. For example, a sales order without a payment method will always bill as an invoice and not as a cash sale. Therefore, you should configure payment methods for use with NetSuite Connector order sync.

Note: NetSuite may provide several default payment methods that are not completely set up. Therefore, even though these payment methods appear in the list of payment methods, you cannot use them. You can edit and save a payment method in NetSuite before using the payment method with order sync.

For more information, read [Specifying Payment Methods](#).

Syncing Amazon FBA Orders

To sync Amazon FBA orders, set a fixed customer in NetSuite for such orders.

For more information, read [Managing Fulfillment by Amazon \(FBA\) Items in NetSuite Connector](#).

Setting Up NetSuite Connector Product Sync

Follow these procedures before using NetSuite Connector's product, price, or quantity sync features.

1. [Installing from the SuiteApp Marketplace](#)
2. [Setting a Test Item to Sync in NetSuite](#)

Install the NetSuite Connector SuiteApp from the SuiteApps Marketplace. For more information, read the help topic [Installing from the SuiteApp Marketplace](#).

If you experience issues after installing the SuiteApp, ensure that you completed the installation steps. Leaving the installation steps incomplete may cause issues in creating invoices for orders, or other order sync errors.

After installation, verify that your SKU field is automatically mapped for product sync. If not, map the SKU field.

Setting a Test Item to Sync in NetSuite

Select a test item in NetSuite to test product sync.

For more information, read [NetSuite Connector Product Sync Tests](#).

Note: During the test, the data on your marketplace or cart may change and the data can be incorrect. Do not select an item for which an incorrect data sync can result in meaningful negative consequences.

Review the following guidelines for price and quantity version of product sync:

- The test item must exist in the marketplace or cart with an SKU matching the value on that item in NetSuite. Price or quantity sync can update only existing items.
- To verify data sync, ensure that the price and quantity data on the item in marketplace or cart is different from the data in the order.

Review the following guidelines for full product sync version of product sync:

- You should test with an item that exists in your marketplace or cart with an SKU matching the value on the same item in NetSuite.
- To verify data sync, ensure that the price and quantity data on the item in marketplace or cart is different from the data in the order.
- You should not use matrix items during your initial testing.

To set a test item to sync to NetSuite:

1. In the global search, enter the SKU of your test item.
2. Click on the matching item search result to open the item record..

Note the value in the **Item Name/Number** field. This is the default SKU of the item. You will need this value when testing product sync. You can change the SKU field.

3. Click **Edit**.
4. Click the subtab with the name of your marketplace or cart that you are syncing (for example, Shopify).

The subtab is created when you install the NetSuite Connector SuiteApp.

5. Under the **Required** subtab, from the **Flag** list, select **Add/Update Item**.
6. Click **Save**.

Note: For full product sync testing with a matrix item, repeat the above procedure for the parent and all child items of the matrix family.

Setting Up NetSuite for NetSuite Connector Syncs

Before proceeding to NetSuite Connector Activation, you must perform the tasks listed in this topic.

For guidance on performing these tasks, read:

-  https://system.netsuite.com/help/helpcenter/shared_resources/PDF/SalesOrdersCashSales.pdf
-  https://system.netsuite.com/help/helpcenter/en_US/PDF/ItemRecordManagement.pdf

For order or fulfillment sync, perform the following tasks:

- Create a taxable sales order with shipment and payment methods along with any other information you expect to see in the order, such as discounts.
- Fulfill the sales order
- Bill the sales order
- Note the order number as a working example order.

For refund or settlement sync, perform the following tasks:

- Refund the order created in NetSuite. After the sales order is fulfilled and billed, NetSuite creates an invoice or cash sale record. Refund the record that was created.
- If you are syncing Amazon FBA orders or POS (Point of Sale) as cash sale, refund the cash sale.
- Note the order number as a working example order.

For full product sync, perform the following tasks:

- Create an item in both NetSuite and the marketplace or cart. If you plan to use matrix items, create a matrix family in NetSuite.
- Note the SKU as working example item.

Setting Up NetSuite Connector for Syncs

After setting up NetSuite for integration projects or syncs, you must also set up NetSuite Connector.

To set up NetSuite Connector for integration or syncs:

1. Log in to NetSuite Connector.
2. On the left panel, click the **NetSuite**.
3. Click Settings > General.
4. Under **Import Marketplace/Cart Orders To NetSuite Using**, select the status you want NetSuite Connector to set on synced orders by clicking any of the following buttons:
 - **Default Order Status**
 - **Pending Fulfillment Status**

■ Pending Approval Status

You should click the **Default Order Status**, which means that NetSuite Connector will not specify a status, so NetSuite will assign its configured default. You can create a manual order directly in NetSuite to determine what that default status will be.

5. (Optional) If you have a SuiteCloud Plus License, check the **You Have a SuiteCloud Plus License For NetSuite** box. If you are not sure you have SuiteCloud Plus License, do not check this box because it will cause NetSuite Connector to exceed the limit on concurrent sessions to NetSuite.
6. (Optional) If you have enabled line-item taxes or are using a NetSuite tax plug-in that does this, then check the **Line-Item Taxes Are Enabled In Your NetSuite** box. If you are not sure, do not check this box because it will result in an error upon order import by NetSuite Connector, as it will try to set a tax rate at the line level when no such field exists.
7. (Optional) If you are using the integrated shipping carriers in NetSuite, check the appropriate boxes under the Shipping Accounts:

- **USPS**
- **FedEx**
- **UPS**

Checking a box means you have enabled Shipping Label Integration in NetSuite and has set up an account on the carrier. In addition, you can create and be billed for labels for the carrier directly in NetSuite.

8. If you enabled the Pick, Pack And Ship feature in NetSuite, check the **Pick/Pack/Ship Enabled in Your NetSuite** box.
9. (Optional) If you enabled SuitePromotions and want to use them with the orders that NetSuite Connector syncs, check the **SuitePromotions Is Enabled In Your NetSuite** box. Checking this box does not make NetSuite Connector use the promotions, because you must make that designation on the Discount settings on the connector.

For more information, see [Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite](#).

10. In the **NetSuite Subsidiaries Used For FarApp Connectors** field, enter the internal IDs of the subsidiaries you will use with NetSuite Connector. Separate multiple entries with commas.



Note: Do not enter every subsidiary you have in NetSuite. Enter only the subsidiaries you will sync orders to with NetSuite Connector. In most cases, you will only have as many subsidiaries listed as you have connectors.

11. Click **Save**.

A message is displayed confirming the changes you made.

Marketplace or Cart Connectors

The minimum amount of data typically needed to create an order in NetSuite will be synced by default with no mappings needed. The particular configuration of your NetSuite (fields designated as required, and others) may require you to create additional mappings. But when syncing test orders, usually at least the first order is synced with default data.

The exception is if you are using multiple subsidiaries in NetSuite. If you are using subsidiaries you will likely need to set subsidiary mappings on each one of your connectors in NetSuite Connector.

For more information about creating a subsidiary mapping, see [Mapping a Subsidiary on Orders in NetSuite Connector](#).

Managing Connector and Account Settings in NetSuite Connector

You can perform following operations on connectors and accounts:

- [Accessing Connector and Account Settings in NetSuite Connector](#)
- [Adding a Connector to a NetSuite Connector Account](#)
- [Updating the Connector Account Nickname in NetSuite Connector](#)
- [Disabling or Removing a Connector from NetSuite Connector](#)

Accessing Connector and Account Settings in NetSuite Connector

You can view and manage settings for NetSuite Connector and also for individual connector accounts.

Accessing NetSuite Connector Account Settings

The NetSuite Connector account settings are distributed across following three tabs:

- Profile
- Manage Users
- Change Password

To access NetSuite Connector account settings:

1. Log in to [app.farapp.com](#).
2. Hover over your account name on the top-right corner and click **Account Settings**.

Accessing Individual Connector Account Settings

You can also manage individual connector account settings.

To access connector account settings:

1. Log in to [app.farapp.com](#).
2. From the left-side navigation menu, select the desired connector and then the account.
3. Go to one of the following:
 - To view or edit your account credentials, go to Settings > Credentials.
 - To view or edit notification settings, go to Settings > Notifications.
 - To view or edit order sync settings, go to Settings > Orders.
 - To view or edit other transaction settings, go to Settings > Other Transactions.
 - To view or edit fulfillment sync settings, go to Settings > Fulfillments.
 - To enable or disable syncs, or change the account nickname, go to Data Flows > Manage Data Syncs.

Adding a Connector to a NetSuite Connector Account

Add connectors for the marketplaces or carts to transfer data between the marketplaces or carts and NetSuite.

To add a new connector or to add an account to an existing connector, contact your NetSuite Account Manager.

Authorizing a Connector

After creating a connector, you must authorize the connector so that NetSuite Connector can transmit data to and from the endpoint.

To authorize a connector:

1. Log in to app.farapp.com.
 2. Select a connector and then select the relevant account.
 3. Go to Settings > Credentials.
- The Credential Settings page opens. The required credentials vary from connector to connector. The Credential Settings page provides a link that takes you to the instructions on obtaining credentials from the marketplace or cart.
4. Provide the appropriate information.
 5. Click **Save Settings**.

Enabling the Connector and Account

To test the sync, you must first enable the connectors and account.



Note: Enabling connectors does not enable the individual syncs.

To enable a connector and an account:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Manage Data Syncs.
4. Click the **Toggle this connector active/inactive** button on the top right to enable the activation.

Updating the Connector Account Nickname in NetSuite Connector

If you have multiple accounts under one connector, you may want to differentiate the accounts by setting a nickname for each account.

To update the connector account nickname:

1. Log in to app.farapp.com.
2. Select the connector and relevant account.

3. Go to Data Flows > Manage Data Syncs.
4. Below the **Nickname** field, click **Edit** (pencil icon).
5. In the Update Account nickname popup window, in the **Nickname** field, enter the nickname.
6. Click **Update Nickname**.

Disabling or Removing a Connector from NetSuite Connector

You can disable a connector or remove a connector from NetSuite Connector. To disable a connector, you need to make the connector inactive.

To disable or remove a connector:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account that you want to disable or remove.
3. Go to Data Flows > Manage Data Syncs.
4. Click the **Toggle this connector active/inactive** button to switch it to **OFF**.
The connector disables.
5. To remove the connector, click the **Trash** icon that appears next to the **Toggle this connector active/inactive** button.

NetSuite Connector Sync Tests

Before using syncs in your live production environment, you should first test the syncs with test or sample orders.

When providing a test order or test product to your NetSuite Connector integration specialist, follow these guidelines:

- Carefully select the order or product for testing because there is a chance that something may not sync correctly. Do not select your top-selling SKU product for product sync testing. There is a chance that the price, quantity, or some other important data does not sync correctly.
You should not provide a test order containing items that are out of stock in NetSuite, because such orders will not be fulfilled in NetSuite.
- The order or product you select should be a good representation of your data. For example, if 90% of your SKUs are matrix items, make sure to test with a product that is a matrix item in NetSuite.
Your test order should have data populated for the fields you expect NetSuite Connector to sync to NetSuite. For example, assume you want the Memo field to sync over from the marketplace or cart orders. Make sure your test order contains data in the field you map to the Memo field in NetSuite.

NetSuite Connector Order and Fulfillment Sync Tests

Before using order and fulfillment sync in your live production environment, you should first test the processes with test or sample orders.

Do the following procedures to test and activate order and fulfillment syncs:

- [Testing NetSuite Connector Order Sync](#)
- [Verifying NetSuite Connector Order Sync](#)
- [Testing NetSuite Connector Fulfillment Sync](#)
- [Activating NetSuite Connector Order and Fulfillment Sync](#)

NetSuite Connector Test Orders Prerequisites

To test order and fulfillment sync, you will need one or more test orders.

The test orders will be retrieved from your marketplace or cart and posted to NetSuite manually to ensure the sync works as expected.

Ensure the following prerequisites are met when preparing your test orders:

- The test order contains tax.
You can ignore this if you are not required to collect tax in any jurisdictions or you sell only on marketplaces that remit tax on your behalf.
- The customer of your test order does not exist in NetSuite.
If you will be using a designated customer record in NetSuite, you can ignore this requirement.
For more information on designating a customer, see [Assigning a Fixed Customer to All Orders in NetSuite Connector](#).
- All items in the test order exist in NetSuite and have SKUs that exactly match, one-to-one.

By default, NetSuite Connector uses the Item Name/Number field (field ID: itemId) as the SKU field on your item records in NetSuite. Also, these items must have quantity available at one or more locations in NetSuite so the test order can be fulfilled.

NetSuite Connector can use the same test orders to confirm that refund sync between NetSuite and your marketplace or cart works as expected. If you will be setting up refund sync, you should refund the test order after it has been fulfilled and billed.

For information on how to create a test order for a specific marketplace or cart, see [Creating Test Orders in Marketplace or Cart](#).

Creating Test Orders in Marketplace or Cart

The test orders generation process differs from storefront to storefront. Some storefronts permit you to create orders specifically for testing, whereas, others require you to generate a normal order.

Most marketplaces do not permit you to create a test order. To set up order sync for a marketplace, create a live order for a low value item, using yourself as a customer. Then, use that order for testing.

Most marketplaces require orders to be shipped within a certain time frame and penalize you if you fail to meet that requirement. If the marketplace for which you are creating a test order has such a requirement, wait to create the test order until you know it will be marked shipped in time. Doing so will avoid any penalties.

Most carts permit you to create test orders. Refer to the respective storefront's documentation for more information. However, if you cannot find test order documentation for the cart, or you are unable to create a test order, then create a live order. The live order can be for a low value item, using yourself as a customer. Then, use that order for testing.

Testing NetSuite Connector Order Sync

After setting up NetSuite Connector and NetSuite for integration and syncs, you can perform syncing of a test order to NetSuite.

Before proceeding with the testing, consider the following:

- It is recommended that you create or identify a test order, as problems may be encountered during the sync.
- Most carts let you create an order specifically for testing. For more details, refer to your cart documentation. But if your cart does allow you to create a test order, you can place an actual order from your cart and then make up a sample customer.
- For Amazon and eBay testing, place an order for your own product in the marketplace to use as a test order.
- Do not select a Shopify POS order or Amazon FBA order for testing, as those orders do not need to be fulfilled in NetSuite and therefore will not be suitable for fulfillment sync testing.
- Do not create any mappings until after your initial order syncs. All commonly required data will sync by default if the marketplace or cart provides that data in the order.

To sync a test order to NetSuite:

1. Create or identify an order for testing in your marketplace or cart.
2. Log in to NetSuite Connector.
3. On the left panel, select the relevant connector and account.
4. Click **Data Flows > Orders**, to display the Order Dashboard.

5. Click **Retrieve**.
6. On the Retrieve Order Into FarApp window, in the **Order ID** field, enter the test order ID and click **Retrieve**.

NetSuite Connector will import the order from your marketplace or cart. The order will appear in the Order Dashboard with the status, In Farapp (Automatic Sync Disabled). If you cannot see the order refresh the page.

7. On the Order Dashboard, click **pencil icon** under the **Action** menu. Then, click **Post Order To NetSuite**. NetSuite Connector will post the order to NetSuite.

If your order posts successfully, the status will indicate **Order Posted - Waiting For Shipment**.

If there is any problem in posting your order to NetSuite, the status will indicate **Error Posting to NetSuite**.

Verifying NetSuite Connector Order Sync

After your order has imported to NetSuite, you should log in to NetSuite and verify that the order posted as expected.

For example, a posted item will be reflected on a Sales Order record, and you should check the following:

- The Status is **Pending Fulfillment**.
- The Summary section indicates the correct amounts for the Subtotal, Discount, Tax Total, Shipping Cost and Total.
- On the **Items** subtab, check if the line item's item number, quantity and amount match those in the marketplace or cart.



Note: As NetSuite is a customizable platform, the location of information may not be the same for every user. This example is based on the most common NetSuite configuration, but your information may display differently.

If you encounter an error, refer to the [Troubleshooting Order or Fulfillment Sync Errors in NetSuite Connector](#).

Testing NetSuite Connector Fulfillment Sync

After verifying the order in NetSuite and marketplace or cart, you must fulfill the order in NetSuite and sync the fulfillment information back to your marketplace or cart.



Note: If you are using a 3PL connector with NetSuite Connector, you should fulfill the order using your 3PL. If you do not use a 3PL connector, you must manually fulfill the order in NetSuite. Once fulfilled in NetSuite, the fulfillment data will be returned to the order source.



Important: If you will perform fulfillment order syncs using a 3PL connector, you must first complete the steps for activating or preparing 3PL sync. For more information, see [NetSuite Connector 3PL Sync Test Prerequisites](#). After activating or preparing your 3PL, you can perform the step 1 in the following procedure, but you must skip step 2.

The following steps must only be completed if you are NOT integrating or using a 3PL connector with your order sync.

To sync fulfillment:

1. Fulfill the order on NetSuite.
 - a. Go to the Item Fulfillment page for the order you want to sync. Depending on how you fulfill orders, there are several ways to open the page. For more information see the help topic [Fulfilling Orders](#).
 - b. On the Item Fulfillment page, go to the **Packages** subtab.
 - c. In the **Package Tracking Number** field, enter the tracking number. Adding the tracking number ensures that it appears on the storefront and other locations where the order appears.
 - d. Click **Save**.
2. After the order is fulfilled with a tracking number in NetSuite, you must upload the fulfillment to your marketplace or cart from the NetSuite Connector dashboard:
 - a. Log in to NetSuite Connector.
 - b. On the panel, select the relevant connector or account and click **Data Flows > Orders**.
 - c. Locate the order you fulfilled. Click the **pencil icon** under the **Action** menu and then click **Upload Shipment from NetSuite**.

After the fulfillment is synced, the order status is automatically set to **Complete**. Information on the item fulfillment record, such as shipping method and tracking number, are also synced on the marketplace or cart.

Activating NetSuite Connector Order and Fulfillment Sync

You can activate order and fulfillment sync after successfully testing them.

Before proceeding to activating your syncs, read and understand [NetSuite Connector Deployment to Production](#).

The following procedure activates order and fulfillment sync only.

To activate order and fulfillment sync:

1. Log in to NetSuite Connector.
2. On the left panel, select the relevant connector or account.
3. Click **Data Flows > Manage Data Syncs**.
4. Set **Order Sync** to **On**.
A message appears about completing order mappings.
5. Click **Enable**.
6. Set **Fulfillment Sync** to **On**.

Your order and fulfillment sync will now run. All processes you manually completed during testing will now be automatically performed by NetSuite Connector.

Note: Before using Price/Qty or Full Product Sync, you must perform the activation procedure for that sync before using it live production.

NetSuite Connector Product Sync Tests

Before using product sync in your live production environment, you should first test the process with test or sample products.

Do the following procedures to prepare test products and use them to test the product sync process:

1. NetSuite Connector Test Products Prerequisites
2. Mapping NetSuite Connector Test Products
3. Mapping Inventory and Price for NetSuite Connector Test Products
4. Testing NetSuite Connector Product Sync
5. Activating NetSuite Connector Product Sync

NetSuite Connector Test Products Prerequisites

To set up product sync, you should create or identify one or more test products. The test products or items will have changes or adjustments to their data in the marketplace or cart. To prevent this from affecting your live online sales, you should select sample products or items that have little to no interaction with customers.

NetSuite Connector product sync has two versions — the Price and Quantity sync and the Full Product sync. You can only run one syncs at a time on a particular connector or account combination.

Price and Quantity Sync

If you are setting up price and quantity sync, you must have test items that exist in both NetSuite and your marketplace or cart. Ensure that you meet the following prerequisites:

- The SKU in NetSuite (typically the Item Name/Number field) exactly matches the SKU in the marketplace or cart.
- If syncing price, you have a price value entered in one of the price levels on your test item in NetSuite.
- If syncing quantity, you have quantity available at one of your item locations in NetSuite. If you would rather sync quantity from a custom field on the item record, ensure that your custom field has a quantity specified.

Full Product Sync

In setting up full product sync, the test items you select or create will depend on whether you sell variation items or not. If you sell variation items, you must meet these prerequisites:

- Use NetSuite's matrix item type to create your variation products. When you create a matrix item in NetSuite, you will define a parent item, child items or subitems, and variation fields like size and color.
- Create one matrix item family in NetSuite that can be used for testing.
- Limit the number of subitems to no more than 5 per test item family.



Note: If you sell standalone items or those without variation, you will need one sample standalone product in NetSuite that can be used for testing.

Item Fields

The following fields are required for nearly all marketplaces and carts. Ensure these basic fields are filled out with values or data on your test items in NetSuite, in addition to any other data you may want to sync.

- Flag field in NetSuite Connector
For more information on the Flag Field, see [NetSuite Connector Storefront Flag Fields](#).
- Product title

- Product description
- Price
- Quantity
- Image

For more information, see [Adding Product Images for NetSuite Connector](#).

- Item options (size, color, and other variations)

This is only required for variation items.

Marketplaces like Amazon and eBay have required item fields in addition to those in the list.

Creating New Product Listings

If you want NetSuite Connector to create new product listings in your marketplace or cart, ensure that your test items do not exist in your marketplace or cart.

Designating The Products To Sync

After selecting test products, you must flag the items for syncing to signal to NetSuite Connector that you want to sync them. Do not set all your items to sync at first, even if that is what you want. When you go live, you will have to flag more products to sync, but for now, flag only the test products to sync.

For more information, see [Flagging Existing NetSuite Items to Sync](#).

Mapping NetSuite Connector Test Products

Before setting up NetSuite Connector for product sync, ensure that you prepared for NetSuite product sync. For more information, read [Setting Up NetSuite Connector Product Sync](#).

The mapping category indicates to NetSuite Connector which set of fields to use for your products. You must add and set a category mapping for most marketplace or carts. The category mapping you create should depend on your use of either Full Product or Price and Quantity sync.

In choosing the appropriate type of product sync, you must consider to use only one sync per connector or account combination. Using both Price and Quantity sync and Full Product Sync on the same connector or account combination is not supported. However, you can use a different sync type across various accounts and connectors. For example, if you have Amazon connector with two accounts, you do not have to choose the same product sync type for both accounts, although you can do that if you want.

To decide which sync type you should be using, refer to [Price/Quantity Sync vs. Full Product Sync](#).

After deciding on the sync you want to set up, click the link appropriate for your company and follow the procedure.

- For carts (Shopify, Magento, etc.), see [Mapping Cart Categories in NetSuite Connector](#).
- For Amazon, see [Mapping Amazon Categories in NetSuite Connector](#).
- For Walmart and NewEgg, contact NetSuite Customer Support for assistance.
- For eBay, no category mapping is required. Your products will be automatically categorized based on whether the items have variations.

Mapping Inventory and Price for NetSuite Connector Test Products

Price and inventory fields are often the most mapped across all connectors, because most users manage their pricing and inventory in NetSuite.

Mapping Inventory Fields

To map inventory fields, see [Mapping Inventories in NetSuite Connector](#). After inventory mapping, proceed with price mapping.

Mapping Price Fields

To map price fields, see [Mapping Prices in NetSuite Connector](#). After price mapping, proceed with Product Sync Activation,

Mapping Inventory and Price for NetSuite Connector Test Products

Price and inventory fields are often the most mapped across all connectors, because most users manage their pricing and inventory in NetSuite.

Mapping Inventory Fields

To map inventory fields, see [Mapping Inventories in NetSuite Connector](#). After inventory mapping, proceed with price mapping.

Mapping Price Fields

To map price fields, see [Mapping Prices in NetSuite Connector](#). After price mapping, proceed with Product Sync Activation,

Testing NetSuite Connector Product Sync

Before syncing your test item to the marketplace or cart, you must have selected a test item, noted its SKU, and flagged it for syncing. These are prerequisites for product sync testing. For more information, read [Setting Up NetSuite Connector Product Sync](#).

Note: Product sync test will modify the data on your marketplace or cart. Ensure that you select a sample test item whose data if modified will not result in negative consequence.

To test product sync:

1. Log in to NetSuite Connector.
2. On the left panel, select the relevant connector or account and click **Data Flows > Products**.
3. Click **Retrieve**.
4. On the Retrieve Product Into FarApp window, in the **Product SKU or Internal ID** field, enter the SKU value of the product you want to sync. Then, click **Retrieve**.
The product will be imported to NetSuite Connector.
5. Close the Retrieve Product Into FarApp window and refresh the page.
If your item is not displayed, refer to [Troubleshooting Missing Retrieved Products in NetSuite Connector](#).
6. In the row of the product you imported, under the Action column, click the pencil icon.
The options under the Action column will differ according to the type of sync.

7. Do any of the following:

- If you are testing full product sync, click **Post to marketplace/cart**, where the marketplace or cart is the connector you are testing. NetSuite Connector will then post the product to your marketplace or cart.
- If you are using price and quantity sync, click **Update Inventory and Price in marketplace/cart**, where the marketplace or cart is the connector you are testing. NetSuite Connector will then post the price or quantity to your marketplace or cart.

Verifying NetSuite Connector Product Sync

After NetSuite Connector successfully syncs your test product, you should verify the result directly in the marketplace or cart. Check if the fields in the marketplace or cart you mapped over from NetSuite do match the NetSuite source field values. Check the price or inventory quantity fields as those are important values that may have great impact. Adjust or add mappings accordingly.

Activating NetSuite Connector Product Sync

After completing your product sync setup and testing, you are ready to activate your Price and Quantity sync or Full Product sync.

To activate product sync:

1. Log in to NetSuite Connector.
2. On the left panel, select the relevant connector or account.
3. Click **Data Flows > Manage Data Syncs**.
4. Set either **Price/Quantity Sync** or **Product Sync** (for Full Product sync) to **On**.



Note: Do not enable both at the same time in the same connector or account.

Your product sync is now running. When an item is set to sync in NetSuite, NetSuite Connector will automatically sync the item to your marketplace or cart.

NetSuite Connector Refund Sync Tests

Before using refund sync in your live production environment, you should first test the process with test or sample orders.

Do the following procedures to prepare test orders and use them to test the refund sync process:

1. [NetSuite Connector Test Refund Prerequisites](#)
2. [Testing NetSuite Connector Refund Sync to NetSuite](#)
3. [Activating NetSuite Connector Refund Sync](#)

NetSuite Connector Test Refund Prerequisites

Before starting NetSuite Connector activation for refund sync, you must complete a successful test of order sync and bill the test order in NetSuite. The process of billing a test order is handled as a NetSuite process. Typically, an order without a payment or with terms is billed as an invoice and an order with a payment is billed as a cash sale. The billed order record is refunded.

Next, you should decide how you want to sync refunds. The leading practice is to sync refunds for marketplaces such as eBay to NetSuite and refunds for carts such as Shopify from NetSuite.



Note: Some connectors let you choose syncing in either direction. However, you must not enable the sync in both directions.

For next steps, check one of the following:

- To sync refunds to NetSuite, read [Testing NetSuite Connector Refund Sync to NetSuite](#).

Testing NetSuite Connector Refund Sync from NetSuite

Before syncing the refund from NetSuite to the storefront, you must refund the order in NetSuite. After refunding the order in NetSuite, you should have either a cash refund or credit memo generated from the billed record of your order.

Syncing Refund to Storefront

The cash refund or credit memo is used as the source for refund to the storefront.

To sync refund to the storefront:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. From the **Source Channel** list, select **NetSuite**.
5. Click **Retrieve**.

The Retrieve Refund/Return Into FarApp window opens.

6. In the **Order/Transaction ID** field, enter the order ID.
7. Click **Retrieve**.

NetSuite Connector imports the refund and a success message appears.

If you get a failure message, check the following: and that

- Make sure the order ID is an order NetSuite Connector has previously synced to NetSuite.
- The order has a cash refund or a credit memo generated from the billed record for that order.

8. Click **Close**.
9. From the transactions list, locate the imported refund.

If you do not see the order, refresh the page.

10. Click the **Action** menu (pencil icon) and select **Post Order to NetSuite**.

NetSuite Connector syncs refund to the storefront. After the refund is synced, you get the **Posted** status.

Testing NetSuite Connector Refund Sync to NetSuite

To sync refund from the marketplace or cart to NetSuite, you must first refund the order in the marketplace or cart. The refund must be for an order that previously synced to NetSuite and has not been previously refunded in NetSuite.

Syncing a refund to NetSuite

After the order is refunded in the marketplace or cart, use the following procedure to sync the refund to the storefront.

To sync refund to NetSuite:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. From the **Transaction Type** list, select **Refunds And Returns**.
5. Click **Retrieve**.
The Retrieve Refund/Return into FarApp window opens.
6. In the **Order/Transaction ID** field, enter the order ID.
7. Click **Retrieve**.
NetSuite Connector imports the refund and a success message appears.
8. Click **Close**.
9. From the transactions list, locate the imported refund.
If you do not see the order, refresh the page.
10. Click the **Action** menu (pencil icon) and select **Post Order to NetSuite..**
NetSuite Connector syncs refund to NetSuite. If the refund is synced, you get the **Posted** status. If the refund sync fails, you get an error. Hover the mouse over the refund to view the error message and take appropriate action.

Activating NetSuite Connector Refund Sync

When validating refund sync, check the following:

- Verify that the total refund amount in the storefront matches with the total refund amount in NetSuite. When syncing refunds from NetSuite to storefront, NetSuite Connector does not control the generation of refund record in NetSuite. Consider a situation where you refund an order in NetSuite and your refund record (cash refund or credit memo) does not match the original order total. In such case, the reason for the difference lies in NetSuite and not in NetSuite Connector.
- If you refunded some line items, verify the refunded line items match in both storefront and NetSuite. Some storefronts may not allow refunds of specific line items. In such case, NetSuite Connector does not refund the specific line items but instead syncs a general refund. You can determine whether refund is possible on a line item in your storefront by attempting the refund process manually.

Enabling Refund Sync

Before enabling refund sync, make sure you perform the following steps:

1. Complete refund activation steps 1 and 2
2. Review the [NetSuite Connector Deployment to Production](#).

To activate refund sync:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Manage Data Syncs.
4. Click the **Refund Sync** button for the direction of use.

Some connectors enable refund sync in either direction. Do not enable both directions, because that is not a supported configuration and can cause issues.

NetSuite Connector Third-Party Logistics (3PL) Sync Tests

Before using 3PL sync in your live production environment, you should first test the process with test or sample orders.

Do the following procedures to prepare test orders and use them to test the 3PL sync process:

1. NetSuite Connector 3PL Sync Test Prerequisites
2. Testing NetSuite Connector ShipStation Sync

NetSuite Connector 3PL Sync Test Prerequisites



Note: Before setting up NetSuite Connector for 3PL sync, make sure you have added and authorized your 3PL connector as described in [Adding a Connector to a NetSuite Connector Account](#).

To test 3PL sync, you need a sales order in NetSuite in **Pending Fulfillment** status. If you are using NetSuite Connector to sync orders, you should choose an order that is already synced to NetSuite Connector.

When you have identified the order record to use for test, note down its transaction ID found in the **Order #** field.

Proceed to [Testing NetSuite Connector ShipStation Sync](#).

Testing NetSuite Connector ShipStation Sync



Note: This topic should be used only for 3PL connectors like ShipStation or ShipWorks that utilizes a passive connection.

NetSuite Connector has a passive connection with ShipStation, which means that NetSuite Connector does not actively push or pull data to and from ShipStation. Instead, ShipStation picks data from or drops data into NetSuite Connector. Because of this difference, the steps to activate ShipStation connector are different from other connectors.

Syncing a Test Order to ShipStation

The following procedure describes syncing a test order to ShipStation.



Note: Before proceeding, make sure you have configured ShipStation.

To sync a test order to ShipStation:

1. Log in to app.farapp.com.
2. Select the required 3PL connector and then select the relevant account.
3. Go to Data Flows > Manage Data Syncs.
4. To enable order sync, check the **Order Sync On/Off** button.



Note: In most connectors, order sync is not enabled until the last step of activation. However, because ShipStation and ShipWorks are passive connectors, order sync must be enabled earlier to successfully pull orders from NetSuite Connector.

5. Go to Data Flows > Orders.

Verify if your test order is automatically imported. If yes, proceed with Step 6. If not, perform the following substeps:

- a. Enter the order ID in the **Search** box and click the magnifying glass icon.
NetSuite Connector searches for the order in the current set of orders. If the order is not found, NetSuite Connector attempts to import the order from NetSuite. When the order is in NetSuite Connector, the status for the order is **Awaiting Retrieval from ShipStation**.
The status indicates that the order must now be picked up by ShipStation for fulfillment.
Note that for ShipStation, NetSuite Connector does not control when the orders are picked up. Most orders are picked up within 10 minutes, but the time can vary.
- b. Wait for 10 minutes and then refresh the page to check the status change.
You can also manually initiate a pull from the 3PL.
- c. When the 3PL has the order, the status in NetSuite Connector changes to **Order Posted, Waiting for Shipment**.

6. Log in to your 3PL and ship the order.

Make sure you have the tracking information to verify the sync in NetSuite Connector.

Whether ShipStation automatically notifies NetSuite Connector of the shipment depends on your ShipStation configuration.

7. If you are prompted to notify the marketplace, accept the notification. In this context, marketplace is NetSuite Connector.
ShipStation drops the order back to NetSuite Connector.
8. Wait for 10 minutes.
9. Click the **Action** icon and select **Post Shipment To NetSuite**.

NetSuite Connector creates an item fulfillment for your sales order in NetSuite and changes the status of the order in NetSuite Connector to **Complete**.

Verifying NetSuite Connector ShipStation Sync

After NetSuite Connector syncs the fulfillment to NetSuite from ShipStation, you should verify the sync in NetSuite.

To verify the ShipStation sync in NetSuite:

1. Locate your test order in NetSuite.

You should see one of the following status on the order:

- If you have shipped all items in the order, the order shows **Pending Billing** status.
- If you have yet to ship all the items, then the order shows **Partially Fulfilled/Pending Billing** status.
- If you have NetSuite automation to automatically bill orders, you may see **Billed** status.

2. Click the **Related Records** subtab.

The **Related Records** subtab may be named as **Related Transactions** in some records.

The subtab displays all records created from the sales order.

3. Click the date for the item fulfillment to open the record.
4. On the Item Fulfillment page, verify that the status is **Shipped**.
5. Click the **Packages** subtab.
6. In the **Package Tracking Number** column, verify the tracking number.

The 3PL activation is complete.

NetSuite Connector Settlement Sync Tests

Before using Amazon settlement sync in your live production environment, you should first test the process with test or sample orders.

Do the following procedures to prepare test orders and use them to test the settlement sync process:

1. NetSuite Connector Settlement Sync Prerequisites
2. Testing NetSuite Connector Settlement Sync
3. Activating NetSuite Connector Settlement Sync

NetSuite Connector Settlement Sync Prerequisites

Amazon settlement sync retrieves Amazon settlement report from Amazon Seller Central. The settlement sync imports the fees, credits, refunds, and other credits and debits that are available in your Amazon Seller Central account into NetSuite.

Setting Up Settlement Sync

Before proceeding, make sure you read and understand the steps given in the topic [Importing Amazon Settlement Reports in NetSuite Connector](#).

Identify a customer in NetSuite for associating with the transactions for fees and credits that are not related to orders, like FBA Warehouse fees. Note the internal ID of the customer.



Tip: Create a customer exclusively for settlement sync. Then all non-order modification cash sale and cash refund records can be found by looking for the transactions of that customer.

To set up settlement sync:

1. Log in to app.farapp.com.

2. Select Amazon connector and the relevant account.
3. Go to Settings > Other Transactions.
4. Click the **Settlement** tab.
5. In the **NetSuite SKU Field**, enter the field ID of your SKU field.
For information on field ID, read [NetSuite SKU and Storefront SKU in NetSuite Connector](#).
6. In the **Customer Internal ID** field, enter the internal ID of the customer that you want to associate with non-order related fees and credits.
7. Click **Save**.

Testing NetSuite Connector Settlement Sync

In most cases, Amazon releases settlement report every two weeks. The report covers transactions where settlement fees and credits were applied in the preceding two week.

To start testing, check if you have any settlement reports available to download.

 **Note:** Select a report that includes orders NetSuite Connector has synced with NetSuite. If the settlement report contains settlement data for orders that NetSuite Connector has not synced to NetSuite, errors are generated. To stop the error notification, you will have to cancel the postings for such orders.

If you do not have any reports, make a request to Amazon to generate reports. For more information, read [Importing Amazon Settlement Reports in NetSuite Connector](#).

After you get report ID of the report that you want to use for testing, proceed with the following procedure.

To test settlement sync:

1. Log in to app.farapp.com.
2. Select Amazon connector and the relevant account.
3. Go to Data Flows > Settlements.
4. Click **Retrieve**.
The Retrieve Settlement Report into FarApp window opens.
5. In the **Settlement Report ID** field, enter the report ID.
6. Click **Retrieve**.
NetSuite Connector pulls the settlement report from Amazon. The retrieval may take some time.
A success message appears on the popup window and NetSuite Connector adds the report to the list of settlement reports..
7. Click **Close**.
8. Locate the imported report and click the **Action** menu (pencil icon) and select **Post**.
NetSuite Connector syncs the report in the background. The report may take time to sync. Larger reports may take a day to sync and update the status.
After NetSuite Connector attempts to post each settlement fee and credit, the status of the report appears. If the status of the sync is **Complete**, skip the remaining part of the procedure. If there are errors in the sync, the report displays the errors.
9. If the report shows errors in the status, check each error and take one of the following actions:

- a. Cancel the posting.

If you decide to cancel the posting, the settlement data for that posting does not sync with NetSuite.

- b. Fix the error.

10. Repeat the posting from step **8**.

After the settlement report shows **Complete** status, you can activate the settlement sync.

Activating NetSuite Connector Settlement Sync

When validating settlement sync, consider the following:

- Make sure the test report shows **Complete** status in NetSuite Connector. Any errors, should be corrected or cancelled. Technically, if the order you are verifying does not have any errors, then the synced data should be accurate.
- If an order has cancelled postings, the data in that order will not be accurate.
- Check both a refunded and non-refunded order and verify that the records in NetSuite match Amazon's data.

Note: NetSuite Connector syncs data directly from the v2 Amazon settlement report. The expectation is that the Amazon Seller Central UI represents the data on the order the same as it does in the report. However, if you see a mismatch, download the report directly from Amazon Seller Central and check the data on the report.

Enabling Settlement Sync

After completing settlement sync activation steps 1 and 2, you can activate your settlement sync. You should activate the settlement sync after the Amazon connector is live, because the settlement sync needs orders in the reports that are previously synced by NetSuite Connector. However, if you are setting the Amazon order sync and settlement sync live for the first time, review the going live topics before activating the sync. For more information, read [NetSuite Connector Deployment to Production](#).

Before enabling settlement sync, you may want to adjust the cutoff date for settlement reports.

For more information, read [Setting the Amazon Settlement Report Cutoff Date](#).

To enable settlement sync:

1. Log in to app.farapp.com.
2. Select Amazon connector and the relevant account.
3. Go to Data Flows > Manage Data Syncs.
4. Click the Settlement Sync switch to enable the settlement sync.

The Settlement Sync now runs. All the processes that you manually completed during the testing will now be completed automatically by NetSuite Connector.

NetSuite Connector Manual Sync Tests

This topic covers manual testing procedures for the following:

- Order sync
- Fulfillment sync
- Price and quantity sync
- Full product sync
- Matrix items

Testing Order Sync

You must first retrieve the order into NetSuite Connector and then send the order to NetSuite.

For information about retrieving an order into NetSuite Connector, read [Importing Orders from the Storefront in NetSuite Connector](#).

To send an order to NetSuite:

1. Go to Data Flows > Orders.
 2. Click the **Orders** tab.
 3. From the orders list, locate the order that you want to send to NetSuite.
- The order should be highlighted in blue and should show the status as either **In FarApp (Automatic Sync Disabled)** or **Order Pending Posting to NetSuite**.
4. Click the pencil icon on the order line.
 5. (Optional) To preview what will be sent to NetSuite, click **Show Order Mapping**.
 6. Click **Post Order to NetSuite**.

The status of the order changes to **Posting in Progress**.

After the order is posted to NetSuite, the order status is updated on the dashboard. If the order is successfully sent as cash sales, the dashboard shows the status as **Complete**. If the order is successfully sent to sales orders, the dashboard shows the status of **Order Posted Waiting For Shipment**. If there is an error in sending the order, you will get an error message.

Testing Fulfillment Sync

This procedure assumes that you have sent an order to NetSuite through NetSuite Connector and that order is fulfilled in NetSuite. After fulfilling the order in NetSuite, use the following steps to manually retrieve the fulfillment information and send the information to the marketplace or cart.

To manually retrieve fulfillment information:

1. Go to Data Flows > Orders.
 2. From the orders list, locate the order that you want to send to NetSuite.
- The order should show the status of **Order Posted Waiting For Shipment**.
3. Click the pencil icon on the order line and select **Upload Shipment from NetSuite**.
- NetSuite Connector retrieves the fulfillment information from NetSuite and sends the information to marketplace or cart.
- If the sync is successful, the status of the order changes to **Complete**. If there is an error, the error, the error is displayed on the order line.

Testing Price and Quantity Sync

To complete the price and quantity sync test, make sure your mappings are set up.

For more information, read the following topics:

- The required Category field – [Mapping NetSuite Connector Test Products](#)
- The price and quantity fields – [Mapping Inventory and Price for NetSuite Connector Test Products](#)

To test an item, you must have the **Flag** field set to **Add/Update Item** in NetSuite.

Pulling the Product into NetSuite Connector

To test the price and quantity sync, you must first pull the product into NetSuite Connector.

To pull the product into NetSuite Connector:

1. Go to Data Flows > Products.
The Products dashboard opens.
2. Click **Retrieve**.
The Retrieve Product into FarApp window opens.
3. In the **Product SKU or Internal ID** field, enter the SKU or internal ID of the product.
4. Click **Retrieve**.
The product is added to the list of products in your Products dashboard.

Sending the Product to Marketplace or Cart

After the product is pulled into NetSuite Connector, send the product to the marketplace or cart.

To send the product to marketplace or cart:

1. Go to Data Flows > Products.
The Products dashboard opens.
2. From the products list, locate the product that you want to send to the marketplace or cart.
The product should be highlighted in blue and show the status as either **In FarApp (Automatic Sync Disabled)** or **Product Updates Scheduled to Post to X**. Where X is the marketplace or cart connector name that you are testing.
3. (Optional) To preview the data that NetSuite Connector will sync, read [Viewing Product Mapping in NetSuite Connector](#).
4. Click the pencil icon on the product line.
5. Click **Update Inventory and Price on X**. Where, X is the name of the marketplace or cart connector that you are testing.

The status of the product changes to **Posting in Progress**.

After the product is posted, the product status is updated on the Products dashboard. If the posting is successful, the status changes to **Product Synced to X**. If there is an error in posting the product, you will get an error message.

Testing Full Product Sync

To test a full product sync, make sure you have at least a category mapping and one other field set up.

For more information, see the following topics:

- The required Category field – [Mapping NetSuite Connector Test Products](#)
- The price and quantity fields – [Mapping Inventory and Price for NetSuite Connector Test Products](#)
- Other mappings – [Mapping Required and Optional Fields for Product Sync](#)

To test an item, you must have the **Flag** field set to **Add/Update Item** in NetSuite.

To test the product sync, first pull the product into NetSuite Connector and then send the product to your marketplace or cart. For more information, read [Pulling the Product into NetSuite Connector](#).

After the product is pulled into NetSuite Connector, send the product to the marketplace or cart. For more information, read [Sending the Product to Marketplace or Cart](#).

Testing Matrix Items

Testing matrix items is similar to testing standalone items, except for the following extra steps to make sure the posts to the family items are correct.



Note: The following steps do not apply for price and quantity sync.

1. Set the **Flag** field for both parent and child items appropriately. If you want to post to both parent and child items, the **Flag** fields of both must be set to **Add/Update Item**. To post child items as individual items, the parent item must be set to **Post Children as Standalone** and child item must be set to **Add/Update Item**.
2. Manually retrieve the parent item of all child items into NetSuite Connector.
3. Post only the parent item to your marketplace or cart. The child items are posted with the parent items automatically.

NetSuite Connector Deployment to Production

Deploying NetSuite Connector to production can involve activating and, at time, deactivating some processes. To know more, see the following sections:

- [Checking Syncs for NetSuite Connector Deployment](#)
- [Checking Accounts for NetSuite Connector Deployment](#)
- [Switching to NetSuite Connector from an Existing Integration](#)
- [Deploying NetSuite Connector to Production](#)
- [NetSuite Connector Deployment to Production FAQ](#)

Checking Syncs for NetSuite Connector Deployment

Validate the following configurations:

- Order and fulfilment sync
- Product or Price/Qty sync
- Refund sync (or Amazon settlement sync)
- Sandbox vs. Production NetSuite
- Multiple connectors ready to go-live
- 3PLs

Order and Fulfillment Sync

To validate the sync, verify that NetSuite has sourced and populated the sales order with correct values:

- Tax total in NetSuite matches the tax total in the marketplace or cart.
- Order total in NetSuite matches the order total on the order in the marketplace or cart.
- Line items in NetSuite match the line items on the order in the marketplace or cart.
- Quantity of line items in NetSuite matches the quantity of line items on the order in the marketplace or cart.
- All mapped order, item, and customer fields populate as expected, including the tokenized credit card information.
- If mapped in NetSuite Connector, correct shipment and payment methods populate on the order.

If you have made mapping adjustments, update the order in NetSuite to apply the mappings and confirm that the orders are posted correctly. To test, you should select orders that are in **Order Posted, Pending Fulfillment** status in NetSuite Connector. If you update orders that are marked **Complete**, their status will change back to **Pending Approval** or **Pending Fulfillment** in NetSuite. Also, NetSuite Connector will re-sync the completed order. Follow these steps to update a pending order.

To update a pending order:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Orders.
4. Search the order that you want to map by entering the order ID in the **Search** field and pressing **Enter**.
5. On the order, click the **Action** menu (pencil icon) and select **Update Order in NetSuite**.

After doing the mapping adjustments, set the order cutoff date in the **Import Orders Created After** field in the Advanced Options of the Order Settings page (Settings > Orders).

For more information, read [Initial Order Sync Activation Cutoff Date](#).

Product or Price/Qty Sync

Before enabling the product or price/qty sync, make sure you test multiple SKUs to your marketplace or cart and confirm the SKUs are posted as expected. NetSuite Connector uses flag field to control syncing updates for an item.

For information about the flag field, read [NetSuite Connector Storefront Flag Fields](#).

When you first enable product or price/qty sync, try syncing a small subset of items. After confirming the items are synced without any errors, flag the remaining items to sync. Check the following to ensure the product sync works properly:

- Make sure the items are flagged properly in NetSuite and the mapping for the flag field in NetSuite Connector is correct.
- Verify that the product information from NetSuite to NetSuite Connector is accurate and contains all the details you want NetSuite Connector to sync to the marketplace or cart.
- Verify that the data posted by NetSuite Connector shows correctly in the marketplace or cart.
- Prior to going live, populate your inventory in NetSuite before enabling the product or price/qty sync.



Note: If you are updating the products using CSV file import in NetSuite, do not check the **Run Server SuiteScript and Trigger Workflows** box. Check the box only if you do not want to enable NetSuite Connector's real-time sync.

You can make mapping adjustments from the Product Mappings page in NetSuite Connector. Make sure you select the correct sync type and category (if you have that option). For custom mappings that require support, contact NetSuite support.

Follow these steps when enabling product or price/qty sync:

1. Flag 100 items to sync.
2. Enable product or price/qty sync.
3. Wait for the 100 items to sync.
4. Confirm that data on several of the 100 items appears correctly in the marketplace or cart.
5. Flag 1000 items to sync.
6. Wait for the 1000 items to sync.
7. Confirm that data on several of the 1000 items appears correctly in the marketplace or cart.
8. Flag remaining items to sync.

Refund Sync (or Amazon Settlement Sync)

Check the following to ensure that refund sync is working properly:

- For refund sync from NetSuite to marketplace or cart:
 - Ensure the refund records in NetSuite are tied to the correct orders.
 - Verify that the order is updated in NetSuite Connector and synced to the marketplace or cart.
 - Verify that the refunds are posted correctly to the marketplace or cart.
- For refund sync from marketplace or cart to NetSuite:
 - Verify that when you process a refund in the marketplace or cart, the order is updated in NetSuite Connector and the refund is posted to NetSuite.
 - Verify that the transaction type created in NetSuite matches with what is being synced to NetSuite Connector, that is, refund authorizations, credit memos, or cash refunds.
 - Ensure that the data posted in the refund transactions is correct. For example, amount, quantity, and SKU.

You can update the refund sync settings in NetSuite Connector from the Refund Settings page.

Amazon settlement sync is tested using orders that NetSuite Connector has already synced to NetSuite. Therefore, the leading practice is to test the settlement report processing couple of weeks after NetSuite Connector goes live.

Checking Accounts for NetSuite Connector Deployment

If you are connected to sandbox account, then prior to going live you need to switch accounts before activating your syncs. Switching NetSuite account does not affect the mappings, but ensure that the field IDs and internal IDs mapped are same in the production NetSuite account.

To switch from sandbox to production account:

1. If NetSuite Connector SuiteApp is installed in your sandbox, uninstall the SuiteApp.
2. Install the NetSuite Connector SuiteApp in your production account.
3. After installing the NetSuite Connector SuiteApp in your production account, complete the connection between your production NetSuite account and NetSuite Connector.



Note: When switching from sandbox to production, NetSuite Connector purges the current data including orders, products, and refunds. This data must be purged to avoid conflict with the data on your production account. Note that only the data from NetSuite Connector is purged, no data from marketplaces or carts is purged.

Multiple Connectors Going Live

If you have multiple connectors that you want to set live, do not enable all the syncs at the same time. You should enable one connector sync at a time and maintain a time gap between enabling two connector syncs. During the time gap, check for issues with your orders and items.

3PLs

3PL shipping software like ShipStation and ShipWorks that requires you to manually schedule pulling of new orders through NetSuite Connector do not have a separate go-live process. The connector is considered as live when several days worth of orders have synced through NetSuite Connector. Other 3PLs have order and fulfillment sync buttons that can be enabled from the NetSuite Connector dashboard.

Switching to NetSuite Connector from an Existing Integration

You may have a live integration outside NetSuite Connector which is syncing your items, orders, or other data. To ensure that there are no duplicate or missing items or orders, you should leave your existing service running during the activation process with NetSuite Connector.

You may not use another service and are following manual steps to keep your items updated or your orders are entered in NetSuite and fulfilled. In this case, you should keep following the procedures mentioned in the [NetSuite Connector Deployment to Production](#) section. When activating NetSuite Connector, use some test orders and items to make sure that the functionality works in expected manner. When you are ready to activate the NetSuite Connector service, deactivate your existing service or stop running your manual processes shortly before activating the NetSuite Connector syncs.

When switching your order sync from your previous integration to NetSuite Connector, you should set an order cutoff date. You should also make a note of the timestamp of the last order that was posted by your previous integration. Then, select that data and time, plus one minute in the **Import Orders Created After** field in the NetSuite Connector Order Settings page. Adding this time prevents any orders being skipped or duplicate.



Note: NetSuite Connector imports only open orders, that is, unshipped orders. Importing already fulfilled orders is not part of standard activation process. To import older, already fulfilled orders into NetSuite, use the CSV file import feature.

Deploying NetSuite Connector to Production

After completing all the necessary checks, use the following procedure to activate NetSuite Connector.

To activate NetSuite Connector:

1. Log in to app.farapp.com.
2. On the NetSuite Connector dashboard, click the pencil icon next to the connector that you want to sync.
3. In the Overall Status page, enable or disable the required syncs.

NetSuite Connector Deployment to Production FAQ

Review the questions and answers below when you want to deploy NetSuite Connector into production.

How often does NetSuite Connector sync?

The sync frequency depends on the type and direction of sync. For information about different sync intervals, read [Sync Frequency and Directional Flows](#).

Why are my products not syncing when my mappings are correct?

If the error occurs when you initially try to sync the product, check the error message and take corrective action. After the error is addressed, let the sync run one more time. Else, try resyncing the product manually through NetSuite Connector.

Why are my products taking so long to update or sync?

If you enable real-time price and quantity sync and flag many items in NetSuite to sync, NetSuite Connector starts running the real-time price and quantity sync immediately. Therefore, the process to pick up additional changes from NetSuite must wait until the first batch of the price and quantity sync updates complete. Therefore, updates from NetSuite to NetSuite Connector may be delayed when doing mass updates.

Why are some child items not syncing with the item family?

If there are no errors in the item families and the setup is correct, the missing child items should come after the next product sync. If not, contact NetSuite Customer Support.

If I delete an item from NetSuite Connector, does it delete the item from the storefront or NetSuite?

No, if you delete the item only from NetSuite Connector, the data still exists in the storefront and NetSuite. If the item is still flagged to sync with NetSuite Connector, the item will continue to sync in the next product sync cycle.

Integrating NetSuite Connector with NetSuite

The NetSuite Connector SuiteApp enables integration of NetSuite Connector with NetSuite. This chapter includes the following integration topics:

- Handling NetSuite Data Centers Changes in NetSuite Connector
- Verifying a NetSuite Connector Integration Record
- Working with NetSuite Records in NetSuite Connector
- Splitting an Order into Multiple Item Fulfillments Using NetSuite
- Changing the Primary User Information in NetSuite Connector
- Exposing Internal IDs and Field IDs
- Setting Form Fields Visible Or Editable for NetSuite Connector
- Showing the Fulfilled Field on Sales Orders for NetSuite Connector
- NetSuite SKU and Storefront SKU in NetSuite Connector
- Managing Fields for NetSuite Connector
- Enabling Order Email Messages in NetSuite Connector
- Handling Inventory Items with Alias SKU
- Viewing Inventory History in NetSuite
- Enabling Quantity-Based Pricing in NetSuite
- Using NetSuite Connector for Data Migration
- Configuring VAT in NetSuite Connector
- Handling Incorrect Tax Rates, Tax Codes, or Tax Items in NetSuite Orders
- Creating or Reopening an Accounting Period in NetSuite
- Finding the Cost of Goods Sold (COGS) Account on Items
- NetSuite Connector Saved Search Export
- Undeploying Scripts Installed by NetSuite Connector
- Troubleshooting Common NetSuite Issues in NetSuite Connector

Handling NetSuite Data Centers Changes in NetSuite Connector

NetSuite Connector detects your data center automatically. If your data center changes in NetSuite, NetSuite Connector automatically detects this change and connects to the correct data center.

Verifying a NetSuite Connector Integration Record

The NetSuite Connector integration record is used along with the token you create in NetSuite to allow NetSuite Connector to connect. If you do not see the integration record, check the following:

- The NetSuite Connector SuiteApp creates the NetSuite Connector integration record. Make sure you have installed the NetSuite Connector SuiteApp.

You can view the SuiteApps installed in NetSuite from the Installed Bundles list page.

- If the SuiteApp is installed and you still do not see the NetSuite Connector integration record, the record may be set to blocked. Blocked integrations are not shown by default.

Enabling a Blocked NetSuite Connector Integration Record

Use the following procedure to check and enable the blocked NetSuite Connector integration record.

To check and enable a blocked NetSuite Connector integration record:

- Go to Setup > Integration > Manage Integrations.
- If you do not see the NetSuite Connector record, check the **Show Inactives** box.
- Click **Refresh**.
- If you still do not see the NetSuite Connector record, try to uninstall and reinstall the SuiteApp.
- When the NetSuite Connector record displays in the Integrations page, click **NetSuite Connector**. The NetSuite Connector record opens.
- Click **Edit**.
- From the **State** list, select **Enabled**.
- Click **Save**.

Working with NetSuite Records in NetSuite Connector

This section covers the creation and use of different NetSuite records for use in NetSuite Connector.

Employee Record Creation for NetSuite Connector Token

When you need to grant access to NetSuite Connector token, you must create an employee record in NetSuite.

For more information about creating a NetSuite employee record, see the help topic [Adding an Employee](#).

Merging Duplicate Customer Records in NetSuite

Merging duplicate customer records helps ensure NetSuite Connector matches the correct customer in NetSuite when syncing orders for existing customers.

You can individually merge duplicate customer records or do a mass find and merge of customers.

For merging two customer records, read the help topic [Merging Customer Records](#).

For mass merging of duplicate customer records, read the help topic [Duplicate Record Detection](#).

Billed Record Creation for a Sales Order in NetSuite Connector

NetSuite Connector does not control the record type that NetSuite creates when a transaction is billed. However, NetSuite looks at the field data on the sales order to determine the type of billed record to create. NetSuite Connector can populate certain fields according to your requirement using mappings.



Note: NetSuite is a customizable platform and the information in this topic applies to most of the NetSuite instances. However, it may not apply to certain NetSuite instances.

NetSuite generally decides the record type of the billed transaction based on the data in the **Billing** subtab of the order. Following are the guidelines to create an invoice as the billed record:

1. Set a value in the **Terms** field. NetSuite creates an invoice as the billed record when terms are set on the sales order. Map the **Terms** field through the NetSuite Connector user interface. You cannot place both payment method and terms on a sales order. So, if you want to map the terms, remove the payment method mappings.
2. If neither **Terms** nor **Payment Method** (or **Payment Option** if Payment Instruments feature is enabled) fields have values, NetSuite creates an invoice for the billed record. The most common reason that an invoice is created as a billed record is users fail to map the payment methods. For information about mapping payment methods, read [Mapping Order Payment Methods in NetSuite Connector](#).

To create a cash sale as the billed record, set a value for **Payment Method** (or **Payment Options**) field.

If the Payment Instruments feature is enabled but you are not capturing the payment for the order in NetSuite, create a payment method of the **Offline** type. Then map every order to that payment option.

Viewing Records and Mappings In NetSuite Connector

NetSuite connector provides the ability to view the records associated to a sync type on the page under Data Flows of that sync type.

To view records in NetSuite Connector:

1. Log in to NetSuite Connector.
2. On the navigation menu, select a connector.
3. View your records:
 - To view Orders, go to Data Flows > Orders.
 - To view Products, go to Data Flows > Products.
 - To view other transaction types, go to Data Flows > Other Transactions.

To view your mappings in NetSuite Connector:

1. Log in to NetSuite Connector.
2. On the navigation menu, select a connector.
3. View your mappings:
 - To view Order mappings, go to Mappings > Orders.
 - To view Fulfillment mappings, go to Mappings > Fulfillments.
 - To view Product mappings, go to Mappings > Products.
 - To view Refund mappings, go to Mappings > Refunds.
 - To view other transaction types, go to Mappings > Other Transactions.

Amazon connectors have a page for settlements.

Splitting an Order into Multiple Item Fulfillments Using NetSuite

The Pick, Pack, and Ship feature of NetSuite lets you fulfill different items on an order at different times. To use this feature, in NetSuite Connector you should also enable the **Pick/Pack/Ship Setting is Enabled in Your NetSuite** setting. For more information, read [Enabling Multiple Item Fulfillments from a Single Order in NetSuite Connector](#). You should use the Pick, Pack, and Ship feature to split an order because multiple item fulfillments are created from the same sales order. These fulfillments can then sync back to the storefront. For more information, read the help topic [Pick, Pack, and Ship Overview](#).

When you enable the Pick, Pack, and Ship feature in NetSuite when using NetSuite Connector to sync orders to a 3PL, some 3PL connectors have an optional setting to sync the data from the item fulfillment record instead of the sales order record.

 **Note:** NetSuite Connector cannot import a single sales order from the storefront and post it as multiple sales orders to NetSuite. This rule also applies to cash sales and invoices. You may be able to split the sales order in NetSuite. However, if you split the order, you will get an error when posting the fulfillment data on the original order back to the storefront. Therefore, you should split the order fulfillments using the Pick, Pack, and Ship feature only.

Changing the Primary User Information in NetSuite Connector

Change the primary user information from the account settings in NetSuite Connector.

To change the primary user information in NetSuite Connector

1. Log in to app.farapp.com.
2. On the top right corner of the dashboard, hover over your username and from the list, select **Account Settings**.
The **Profile** tab opens.
3. Make appropriate changes to your information on the page.
4. Click **Save**.
5. Change the password.
 - a. Click the **Password and Security** tab.
 - b. In the **Current Password** field, enter the current password.
 - c. In the **New Password** and **Confirm New Password** fields, enter the new password.
 - d. Click **Change Password**.

If you do not know the password of the primary user account, make sure you have access to the primary user email address. Then reset the password from the link <https://app.farapp.com/reset-password>.

Exposing Internal IDs and Field IDs

Every NetSuite record has an internal ID and every field has a field ID. These IDs are the permanent IDs associated with the records and fields and are often visible in the URL when viewing a record.

To expose internal IDs and field IDs:

1. Enable Client SuiteScript:
 - a. Using an Administrator role, go to Setup > Company > Enable Features.
 - b. Click the **SuiteCloud** subtab.
 - c. Under SuiteScript, ensure that the **Client SuiteScript** box is checked.
 - d. Click **Save**.
2. Enable Show Internal IDs:
 - a. Point to the Home icon and select **Set Preferences**.
 - b. On the **General** subtab, under Defaults, check the **Show Internal IDs** box.
 - c. Click **Save**.

The Field ID is displayed on the Field Level Help popup, which is displayed when you click the label of any field.

When you search for a record type, the search result list of records displays the internal ID of every record in the Internal ID column.

For more information, see the help topic [Showing Record and Field IDs in Your Account](#).

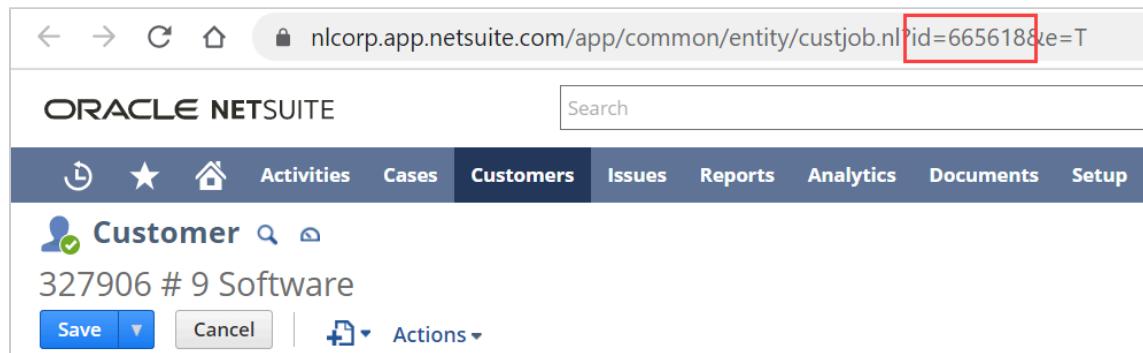
Field IDs are in SuiteScript format, and the field IDs of standard NetSuite fields (non-custom fields) are in lowercase. However, NetSuite Connector communicates with NetSuite using SuiteTalk, and not SuiteScript. In SuiteTalk, all field IDs are in camel case. For example, Item Name/Number field ID is itemid in SuiteScript, but it is itemId in SuiteTalk.

All the standard SuiteTalk field IDs and internal IDs on all NetSuite records are listed here: [Schema and Records Browser](#).

To reference fields, click the Schema Browser tab. To reference records, click the Records Browser tab. To find an entity, click the initial letter of the name of the entity from the alphabetical row. For example, to find the valid field IDs of inventory items, click the Schema Browser tab and click the 'I'. Then, in the left column, click 'InventoryItem'. All valid field IDs on inventory item records are in the Name column.

Finding the Internal ID of Records

NetSuite Connector setups may need the internal ID of certain records like orders, forms, customers, items, and others. To find the internal ID of a record, open the record and in the browser's address bar, check id=xxxx, where xxxx is the internal ID, as shown below:



For more information, see [NetSuite Connector Communication with Internal IDs and Field IDs](#).

To configure NetSuite to display internal ID values in the user interface, see the help topic [Setting the Internal ID Preference](#).

Viewing Internal IDs in NetSuite

NetSuite often refers to objects by their internal IDs when sending error messages through API. Therefore, configuring NetSuite to show internal IDs of objects can be helpful when troubleshooting issues. You can show the internal IDs of objects by enabling the **Show Internal IDs** preference.

For more information, read the help topic [Setting the Show Internal IDs Preference](#).

You can also locate the internal ID of an object by other options mentioned in the topic [Finding the Internal ID of Records](#).

Setting Form Fields Visible Or Editable for NetSuite Connector

NetSuite Connector requires your fields to be visible in your transaction forms for the connector to sync and send your transactions. If a field in your form is not visible, you will encounter errors when attempting to sync your transaction data.

When using custom forms with NetSuite Connector, you need to ensure that the following custom fields are visible or editable:

- NetSuite Connector Storefront
- NetSuite Connector Storefront Order Number
- NetSuite Connector Order Total
- NetSuite Connector Shipping Tax
- NetSuite Connector Line Item Tax
- NetSuite Connector Line Item Amount

For more information about customizing your forms and changing visibility of the fields, see the help topic [Creating Custom Entry and Transaction Forms](#).

For more information about custom forms, see the help topic [Custom Forms](#).

Showing the Fulfilled Field on Sales Orders for NetSuite Connector

NetSuite Connector looks at the quantityFulfilled on your items on your Sales Order form when completing certain processes. It is important to make sure the field is visible on the Sales Order form NetSuite Connector is posting orders to.

To change the quantityFulfilled field to visible:

1. Go to Customization > Forms > Transaction Forms.
2. Locate the form NetSuite Connector is using.
NetSuite Connector posts to the preferred sales order form unless you have created an order mapping to post to a different form.
3. Click **Edit**.
4. Select the **Sublist Fields** subtab.
5. Locate the **Fulfilled** field.

6. Ensure that the **Show** box is enabled.
7. Click **Save**.

NetSuite SKU and Storefront SKU in NetSuite Connector

SKUs (Stock Keeping Unit) are used by storefronts to identify items. During product sync, NetSuite Connector uses the SKU as unique identifier to match items in the storefront to the corresponding item in NetSuite. The SKU value on an item record is stored on a field in NetSuite and this field needs to be defined in the Product Mappings. When NetSuite Connector retrieves an item from NetSuite, it uses the value in the mapped SKU field to find a matching item in the storefront. This requires the SKU values to uniquely match, one-to-one, between the storefront and NetSuite.

It is important for the product field SKU to be properly defined in both product mappings and order mappings. The best practice is to ensure that the defined fields for the SKU values match in both the order mappings and the product mappings. To know more, see the following sections:

- [Setting the SKU Fields in NetSuite Connector](#)
- [Verifying NetSuite SKU Matches the Storefront SKU](#)

Setting the SKU Fields in NetSuite Connector

For both order sync and product sync, the SKUs in NetSuite must match the SKUs in the storefront.

For order sync, NetSuite Connector takes the SKUs on the order and finds matching SKUs in NetSuite to determine the items to place on the order.

For product sync, if the SKUs do not match when sending updates to the storefront, NetSuite Connector gives an error.

Setting the SKU Field for Order Sync

Use the following procedure to set the SKU field for order sync.

To set the SKU field for order sync:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Settings > Orders.
4. In NetSuite SKU Field ID, enter the field ID of the field used to match items on orders to items in NetSuite.



Note: By default, NetSuite Connector uses the **Item Name/Number** field in NetSuite for SKU value. This field ID of this field is **itemId**, which is by default configured as the SKU field value in NetSuite Connector.

Setting the SKU Field for Product Sync

Use the following procedure to set the SKU field for product sync.

To set the SKU field for product sync:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Mappings > Products.

On the Product Mappings page, the row that has **sku** in the **Shopify Field** column indicates the SKU field. The corresponding value in the **NetSuite Field ID** column is the NetSuite SKU field value for product sync.

4. Click **Add Mapping**. The new row is added at the bottom of the Product Mappings screen.
5. In the NetSuite Field ID of the created row, enter the field ID of the field used to match items on orders to items in NetSuite.



Note: By default, NetSuite Connector uses the **Item Name/Number** field in NetSuite for SKU value. This field ID of this field is **itemId**, which is by default configured as the SKU field value in NetSuite Connector.

6. Click **Save**.



Important: Do not change the SKU field value after the sync is live.

Verifying NetSuite SKU Matches the Storefront SKU

To verify that your NetSuite SKU matches the storefront SKU, you must first determine which field is designated as the main SKU field in NetSuite Connector. This field is set separately for order sync and product sync.



Note: The SKU field configured for Order Sync must match the SKU field configured for Product Sync.

Finding the SKU Field for Order Sync

Use the following procedure to locate the SKU field for order sync.

To find the SKU field for order sync:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Settings > Orders.

The Order Settings page opens. The value in the **NetSuite SKU Field ID** field determines the SKU field value used to match items on orders to the items in NetSuite.

Finding the SKU Field for Product Sync

Use the following procedure to locate the SKU field for product sync.

To find the SKU field for product sync:

1. Log in to app.farapp.com.

2. Select the connector and then select the relevant account.
3. Go to Mappings > Products.

On the Product Mappings page, the row that has **sku** in the **Storefront Field** column indicates the SKU field. The corresponding value in the **NetSuite Field ID** column is the NetSuite SKU field value for product sync.

After you identify the SKU field value in NetSuite Connector, confirm that the NetSuite SKU matches the storefront SKU. By default, NetSuite Connector uses **Item Name/Number** field in NetSuite for SKU value. This field ID of this field is **itemId**, which is by default configured as the SKU field value in NetSuite Connector. You can set a different SKU field value for each connector and account combination in NetSuite Connector. However, you should choose one field value and keep it consistent across all connector and account combinations.

Managing Fields for NetSuite Connector

If your NetSuite Connector account has been active for some time, when you make changes in NetSuite, marketplace, or cart, ensure those changes reflect in NetSuite Connector. Not doing so may run the risk of breaking NetSuite Connector mappings or configuration.

If you delete a field in NetSuite or your marketplace/cart, that field must not be included in any mapping in NetSuite Connector, else, the mapping will break. To fix the mapping in this case, you either must update the mapping with a new field from the corresponding endpoint, or restore the field that you deleted. For example, if you deleted the field, `custbody_fa_channel_order`, from NetSuite, NetSuite Connector will not be able to post the marketplace/cart Order ID to NetSuite. The mapping will break. To fix this, you must recreate or restore the field, or create a new field to replace it.

Adding NetSuite Custom Fields

NetSuite Connector will require information from the marketplace/cart to be added to item records in NetSuite. Sometimes, you will lack these fields in NetSuite. If you lack these fields in your item records in NetSuite, you can create custom fields for your item record.

For more information, see the help topic [Creating a Custom Field](#).

Making Custom Fields in NetSuite Available for Global Search

You can index custom fields and make them searchable. For more information, read the help topic [Including Custom Fields in Global Search](#).

Enabling Order Email Messages in NetSuite Connector

The **To Be Emailed** box in NetSuite sets NetSuite to email a copy of the order to the addresses in the email field on the sales order.

You can map this feature from NetSuite Connector.

To map the To Be Emailed box from NetSuite Connector:

1. In NetSuite Connector, select the relevant connector and account.

2. Go to Mappings > Orders.

3. Select the **Order** tab.

4. On the Order Mappings page, click **Add Mapping**.

5. Select **To Be Emailed NetSuite** from the dropdown.

6. Click **Add Mapping**.

The mapping will appear at the bottom of the Order Mappings page.

7. Click **Close**.

The mapping will have a Mapping Type of Fixed by default.

8. Under the **Field/Fixed Value** column, select either **Checked** or **Unchecked** from the dropdown.

This mapping will automatically set the To Be Emailed field checkbox with the state you select when the order is synced to NetSuite.

9. Click **Save**.

Handling Inventory Items with Alias SKU

You should create separate inventory records for every item rather than creating alias stock keeping unit (SKU).

However, if you already have alias items for order processing, the NetSuite Connector can support them. Although requiring additional set up, the NetSuite Connector can support, for example, multiple SKUs in your marketplace or cart that should match with a single SKU in NetSuite when posting orders.

The following methods enable you to handle or manage alias SKUs. Choose a method depending on your records and business scenario.

Method for Kit Items

This method will enable you to support product and order synchronization for your kit items.

For every alias of an item (for example, an item has an SKU called base), you should create a kit item that will have one component item, which will be the original item (base). The SKU of the kit item will be the alias SKU. No additional set up in the NetSuite Connector is required.

When orders are imported, they will import against the kit item, which will correctly commit inventory from the single component item.

Advantages

- No additional set up in the NetSuite Connector is required.
- The NetSuite Connector can synchronize these alias kit items to your marketplaces or carts.
- If the items are synchronized to your marketplaces or carts, a kit item can contain additional data other than what is specified for the original item. A kit item can have a title, description and other information. NetSuite Connector still needs all the fields that a marketplace or cart requires populated for the kit or alias item. But the values for those fields are not referenced from the component or base item. This allows you to set different values for those fields on the alias item, rather than on the base item. But values must be specified for those fields.
- This is a better approach than having alias SKUs. If you sell items under more than one listing, this approach enables you to manage them better.

Limitations

- This method cannot be applied to matrix item types. If you have matrix items set up and need to support alias SKUs for these items, you should use the next method.
- Kit items cannot be included on Transfer Orders. But if your alias SKUs need to be on transfer orders, you should use the next method.

Method for Custom Records

This method enables you to support product and order synchronization for your kit items.

To set up custom records to add aliases:

1. Go to Customization > Lists, Records, & Fields > Record Types > New.
 2. In the **Name** field, enter a name for the record type, like Item Alias.
 3. Ensure that the **Include Name Field** box is checked. The record names will be the alias SKUs.
 4. Click **Save**.
 5. Open the created custom record type and click the **Fields** subtab.
 6. Click **New Field**.
 7. In the **Label** field, enter a label, like Original Item.
 8. In the **Type** dropdown field, select **List/Record**.
 9. In the **List/Record** dropdown field, select **Item**.
- For more information, see the help topics [Working with Records](#), [NetSuite Record Types](#) and [Creating a New Custom Record Type](#).
10. You can now create an alias item by adding new records of this type. A record name will be the alias SKU and it should have a single field for selecting the original item.
 11. Open NetSuite Connector and go to Settings > Order. In **Advanced Options** and under the Alias SKU Settings section, enter the IDs and the field of the SKU alias records you created.

Using this method, every item can have several aliases. This means you do not need to add more fields for item aliases. As more aliases are needed, you do not have to remap anything in the NetSuite Connector when you are adding them.

Viewing Inventory History in NetSuite

Viewing the inventory history in NetSuite is useful in tracking things like oversells and inventory inconsistencies. Following are the places where you can check the inventory history easily:

- Item related records – For more information, read the help topic [Related Information for Items](#).
- Inventory activity detail report – For more information, read the help topic [Inventory Activity Detail Report](#)

Enabling Quantity-Based Pricing in NetSuite

If you have quantity-based pricing enabled in NetSuite, you must also enable quantity based price levels in NetSuite Connector. This is required even if you do not set pricing in quantities above the first (0) level.

For more information, read [Mapping Price from a Quantity-Based Price Level](#).

To enable the quantity-based pricing feature in NetSuite:

1. Go to Setup > Company > Enable Features.
2. Click the **Transactions** subtab.
3. Check the **Quantity Pricing** box.
4. Click **Save**.

Using NetSuite Connector for Data Migration

For certain connectors like Magento, NetSuite Connector offers add-ons like Magento-to-NetSuite Product Sync. The services provided by NetSuite Connector are intended to be used as ongoing syncs. Most NetSuite Connector processes run many times a day.

NetSuite Connector order sync processes are not intended to be used to do a one-time sync of months' worth of past orders. You can request NetSuite Connector to temporarily import even closed orders for the past 30 or 60 days. You should not import more than 2000 orders through NetSuite Connector because doing so will significantly slow down your order sync processes.

One-time data migration between systems should not be done using NetSuite Connector. Such migration should be handled through CSV file import and export or other means provided by NetSuite and your marketplace or cart.

Configuring VAT in NetSuite Connector

Before you configure VAT in NetSuite Connector, you must first confirm whether your storefront uses pre-VAT or post-VAT prices. To identify the VAT, you can import a sample order, edit the order in NetSuite Connector, and then compare the item prices. If you are using a marketplace, you can safely assume that prices already include VAT.

Follow this procedure to configure VAT on a specific account in NetSuite Connector.

To configure VAT in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the relevant connector and account.
3. Go to Settings > Orders.
4. Edit the tax settings.
 - a. Go to the **Tax** tab.
 - b. To manage the VAT, do one of the following:
 - If your storefront provides the item prices with VAT, check the **Subtract VAT from Order Total When Posting** box. When you check this box, NetSuite Connector will subtract the VAT from the item price before posting.
 - If your storefront provides item prices without VAT, clear the **Subtract VAT from Order Total When Posting** box. When you clear this box, NetSuite Connector will not subtract the VAT from the item price before posting.
 - Leave the **VAT Rate For This Account** field blank. In most cases, NetSuite Connector detects the actual rate used in the storefront and does not need you to specify the tax rate. However, if you find that VAT rates do not calculate correctly, you can enter the rate in this field.

In some instances, you may find that the storefront tax rate is different from the NetSuite tax rate. For example, in the storefront, you may see 20%, but in NetSuite, you see 19.9987% tax rate. This behavior is due to the different ways that NetSuite and storefronts round taxes. For more information, read the help topic [VAT and Tax Rounding](#).

5. Click **Save**.

Handling Incorrect Tax Rates, Tax Codes, or Tax Items in NetSuite Orders

For each order, NetSuite calculates the tax code and applies the tax rate based on the shipping address. On the other hand, NetSuite Connector, overwrites the NetSuite tax rate with the rate used in the marketplace/cart. This process ensures that the tax total in NetSuite matches the tax total in the marketplace or cart. NetSuite Connector uses this method because NetSuite does not allow any manual changes to the **Tax** field.

Incorrect Tax Rates

Incorrect tax rates on the order cause discrepancies between the tax total on the order and in the marketplace/cart. By default, NetSuite Connector only sends the tax rate from the marketplace/cart when it creates the order. If the rates do not match, you must check the following areas in NetSuite.

- Verify if your customers are set to taxable by default (applicable only for US customers). If NetSuite assigns a non-taxable customer to an order, then the order becomes non-taxable because there is no tax rate for NetSuite Connector to overwrite.
- Verify if your items are set to a taxable tax schedule. If NetSuite includes a non-taxable item to an order, then there is no tax rate for NetSuite Connector to overwrite. For more information, read the help topic [Taxable Items](#).
- Verify if you have a tax plug-in installed in NetSuite. Some tax plug-ins require additional mappings or settings in NetSuite Connector to work properly. For example, Avalara plug-in requires NetSuite Connector to check a box to specify that this plug-in should not overwrite the tax rate that NetSuite Connector sends. Refer to the plug-in documentation for any NetSuite fields or settings that you need to set to prevent overwriting tax rates.
- Verify if you have an automation in NetSuite that adjusts tax rates, addresses, customers, or any other value on the order that may trigger a tax recalculation.
- Verify if you use line item taxes in NetSuite. Ensure that you have NetSuite Connector set to use line item taxes. In NetSuite Connector, go to Settings > General. On the NetSuite Settings page, check the **Line-Item Taxes Are Enabled In Your NetSuite** box.
- Check if there are conflicts between your NetSuite tax settings and storefront tax settings.

Tax Code Mapping

NetSuite Connector strongly discourages mapping tax codes to NetSuite for several reasons. First, these two platforms have different functions. NetSuite is designed to calculate tax using tax codes, while NetSuite Connector transmits information from one platform to another. Second, if you update tax codes or add new ones in NetSuite, then you must also update the mappings in NetSuite Connector. A support personnel often edits these mappings, which requires a support ticket. Finally, mapping tax codes may create tax total discrepancies between NetSuite and marketplace/cart, which may generate posting errors on orders.

Incorrect Tax Code or Tax Item

If you notice an incorrect tax code or tax item in NetSuite, check the tax nexus setup. As a leading practice, ensure that you manage tax codes and tax items in NetSuite, not in NetSuite Connector. You can test any adjustments to the nexus by manually creating an order in NetSuite and verifying if the system assigns the correct tax code.

For information about how to set up tax nexuses, codes or schedules, refer to the NetSuite Help Center or contact NetSuite Support.

Creating or Reopening an Accounting Period in NetSuite

NetSuite associates transactions with a specific accounting period. If a date on an order is not within an open accounting period in NetSuite the order will fail to sync. You can create, edit, or reopen your accounting periods as needed to sync your transactions.

- To setup new accounting periods, see the help topic [Accounting Period Setup](#).
- To edit an accounting period, see the help topic [Editing an Accounting Period](#).
- To reopen a closed period, see the help topic [Reopening a Closed Period](#).

Finding the Cost of Goods Sold (COGS) Account on Items

NetSuite provides you a way to assign accounts to an item for things like COGS (Cost of Goods Sold), Asset, Income, and various other accounts. If an item in NetSuite does not have an COGS or Asset account assigned to it, you will not be able to create an order with that item. This means that you must set a COGS and Asset account on all your items.

To find the COGS account for an item:

1. Go to Lists > Accounting > Items.
2. Select an item.
3. Click **View**.
4. Click the **Accounting** subtab.

For more information about setting a default Cost of Goods Sold (COGS) account for your item, see the help topic [Selecting a Default Cost of Goods Sold \(COGS\) Account](#). For more information about creating item records, see the help topic [Creating Item Records](#).

NetSuite Connector Saved Search Export

The saved search export records in NetSuite can be configured to any File Transfer Protocol (FTP). NetSuite Connector uploads the saved search results to the configured FTP. If a file with the same name is found on the FTP location, it will be overwritten.

Any limitations of the FTP, such as connections, permissions, or protocols are controlled by the FTP. The saved search only connects to the designated FTP using the protocols, user credentials, address, and directory that you specify and sends the result to the FTP server. Records generated by the saved search are stored in NetSuite Connector for three days.

NetSuite Connector supports generic saved searches to send FTP or Secure File Transfer Protocol (SFTP) feeds.

Before using this feature, ensure that the target FTP or SFTP is supported. Refer to the documentation of the target server or feed and ensure that it supports FTP or SFTP uploads. If it does, know the steps for manually uploading a file through FTP or SFTP and test the upload.

If you can manually upload a file through FTP or SFTP, you can proceed with setting up and enabling saved search exports to FTP or SFTP with the following procedures:

1. [Installing the NetSuite Connector Saved Search Export Bundle](#)
2. [Creating an Integration Application for the Saved Search Export](#)
3. [\(Optional\) Setting up Authentication Credentials for Using Saved Search Export](#)



Note: This procedure is for email and password authentication only. If you are using token-based authentication for NetSuite Connector, do not perform this procedure.

4. [Creating a Saved Search Export](#)

Installing the NetSuite Connector Saved Search Export Bundle

After successfully uploading a file through FTP or SFTP, you can install the NetSuite Connector Saved Search Export bundle.

To install NetSuite Connector Saved Search Export:

1. Go to Customization > SuiteBundler > Search & Install Bundles.
2. Use the Keywords field to search for **NetSuite Connector Saved Search Export** or for bundle ID **402066**.
3. Click the name of the bundle and click **Install**.



Note: Ensure that the Saved Search Export bundle is only installed on the account that is using NetSuite Connector. An export or feed can only be sent by one account at a time.

Creating an Integration Application for the Saved Search Export

After installing the Saved Search Export bundle, you must create an integration application.

To create an integration application for saved search export:

1. Go to Setup > Integration > Manage Integrations > New.
2. In the **Name** field, enter **Saved Search Export**.

3. In the **State** field, select **Enabled**.
4. In the **Authentication** subtab, check the **User Credentials** box.
5. Click **Save**.

An application ID is created when the integration application is saved. You will need this ID in the next procedure, when you set up authentication credentials in the script.

Setting up Authentication Credentials for Using Saved Search Export

You can set up authentication credentials for using the Saved Search Export bundle by editing its corresponding script and entering the credentials.

 **Note:** This procedure is only applicable if you are using email and password authentication for NetSuite Connector. If you are using token-based authentication, do not perform this procedure.

To set up credentials for using saved search export:

1. Go to Customization > Scripting > Script Deployments.
2. In the **Scripts** dropdown, select **Saved Search Export Scheduled Script**.
3. Click **Edit** on the deployment displayed in the list.
4. Click the **Parameters** tab and enter your email and password (the credentials you use to log in to NetSuite Connector) in the fields.
5. In **Application ID**, enter the ID you copied when creating the integration application.
6. Click **Save**.
7. Go to Setup > Company > General Preferences, then click **Custom Preferences**.
8. In the NetSuite Connector Saved Search Export section of the Custom Preferences subtab, enter your credentials and application ID.
9. Click **Save**.

 **Note:** Errors in using the Saved Search Export bundle, including details, are displayed in the saved search script deployment history.

Creating a Saved Search Export

After creating the integration application and setting up authentication credentials, you can create saved search exports to send files or records by FTP.

Every saved search record you create that is actively used is called a feed.

To create a saved search export:

1. In global search, enter **Page: New Saved Search Export**.
2. Enter the required information about the file you want to save, the search you want to use, and how you want the file to be formatted.
3. Use the fields on the **Schedule** subtab to set the schedule of the export.

4. On the **FTP** subtab, enter the information of the FTP server where you want to send files.
5. Click **Save**.



Note: FTP only updates the file time stamp if it detects that the file is different. If the content of the sent file has not changed, the time stamp is not changed.

Troubleshooting Saved Search Export Errors in NetSuite Connector

To identify and examine common FTP connection errors, perform the following procedures.

- [Viewing Exception Logs](#)
- [Troubleshooting Saved Search Export Script Deployment](#)
- [Troubleshooting FTP Connection Errors](#)

Viewing Exception Logs

You can view the errors encountered on the NetSuite account in the exception logs to get more details on the error.

To view exception logs:

1. Go to Customization > Scripting > Script Deployments.
2. Expand the **Filters** section.
3. In the **Script** list, select **Saved Search Export Scheduled Script**.
4. In the results section, click **View** to open the script deployment record.
5. In the script deployment record, go to the **Execution Log** tab.

Troubleshooting Saved Search Export Script Deployment

You can view additional diagnostic data added to what happens during script deployment by running the script in debug mode.

To manually troubleshoot script deployment:

1. Go to Customization > Scripting > Script Deployments.
2. Expand the **Filters** section.
3. In the **Script** list, select **Saved Search Export Scheduled Script**.
4. Edit and re-run the script.
 - a. In the results section, click **Edit**.
 - b. Change the debug level from **Error** to **Debug**.
 - c. To enable manual export, change the status from **Scheduled** to **Testing**.
 - d. From the **Save** dropdown list, click **Save and Execute**.
5. **Edit** and **Save and Execute** the script as needed.

Troubleshooting FTP Connection Errors

Incorrect directory path is a common cause of the Unable to Connect to the FTP Server Error. If you can connect to the FTP server using a different FTP client with the same credentials, it is possible that the server directory path is incorrect.

Slashes and backslashes are common causes of incorrect directory paths. Best practices on using slashes in directory paths are as follows:

- If the FTP server is running on Windows, the path should only contain backslashes.
- Leading or trailing slashes should be removed.

Undeploying Scripts Installed by NetSuite Connector

To stop using one or more NetSuite Connector's scripts, you should undeploy the script.

To undeploy the scripts installed by NetSuite Connector:

1. Go to Customization > Scripting > Script Deployments.
2. (Optional) To expand the filters, click **Filters**.
3. From the **Script** list, select the script that you want to undeploy.
4. On the script result line, click **Edit**.
5. Clear the **Deployed** box.
6. Click **Save**.



Tip: To deploy the script again, click the **Deployed** box and save the script again.

Troubleshooting Common NetSuite Issues in NetSuite Connector

This section provides information about different NetSuite issues and their possible resolution. This section covers the following topics:

- Troubleshooting Common NetSuite Configuration Issues
- Troubleshooting Fields Not Being Visible or Editable Errors
- Troubleshooting Last Name is Missing Errors
- Troubleshooting Character Limit Errors in NetSuite
- Troubleshooting Issues When Using Avalara or Scripts on Sales Orders
- Troubleshooting Sync Script Errors

Troubleshooting Common NetSuite Configuration Issues

Following are some common issues that involve configuration changes in NetSuite.

Issue	Resolution
All inventory is suddenly getting blanked out in the storefront.	If you have the Multi-Location Inventory box in NetSuite checked, NetSuite hides the location list. Clear this box so that NetSuite Connector can see the inventory. If you change the status of this box, you should remove the inventory mappings and recreate them in NetSuite Connector. However, if the issue still does not resolve after rebuilding, contact NetSuite Customer Support.
Shipping cost does not post to NetSuite. The Shipping Cost field is not active on the order form.	Enable the Charge for Shipping feature in NetSuite. For more information, read the help topic Setting Up Shipping .

Troubleshooting Unexpected Errors NetSuite Connector

Sometimes, NetSuite may fail on a request and not return a specific error message to NetSuite Connector. In such case, NetSuite Connector shows that NetSuite reported an unexpected error with an error ID. To fix the issue, contact NetSuite Customer Support and provide the error ID mentioned in the error message.

Troubleshooting Fields Not Being Visible or Editable Errors

The **custbody_fa_channel** and **custbody_fa_channel_order** are the default fields NetSuite Connector uses for order processing. These fields are automatically created when you install the NetSuite Connector SuiteApp.

If you get the error even after installing the SuiteApp, it indicates the fields are not set to be visible on the form that NetSuite Connector uses for orders. To resolve this error, follow this procedure:

To set the default order processing fields as visible:

1. Go to Customization > Forms > Transaction Forms.
2. On the Custom Transaction Forms page, locate the form that NetSuite Connector uses to post orders and click **Edit**.
In most cases, you will be posting to a sales order form. In few cases, such as Amazon Fulfillment By Amazon (FBA), you may post to a cash sale form. Unless you map a different form ID, NetSuite Connector uses your preferred form.
3. On the form, click the **Tabs** subtab.
4. In the sublist, in the **Custom** row, check the **Show** box.
5. Click the **Screen Fields** subtab and in it, click **Custom**.
6. In the sublist, in all the rows that contain **NetSuite Connector** in the label, check the **Show** box.
7. Click **Save**.

Troubleshooting Last Name is Missing Errors

This error appears when NetSuite Connector tries to sync a customer for your order but the complete customer information is not provided from the storefront. NetSuite Connector cannot create a customer if the **Last Name** field is empty.

Depending on the connector and the reason for the error, use one of the following options to resolve the error:

- Amazon Fulfillment By Amazon (FBA) orders do not contain complete customer information. Because Amazon ships the order on your behalf, they do not provide the complete customer information using API. When setting up the FBA orders, you should set up a fixed customer to associate with your FBA orders. To resolve this error for FBA orders, create an Amazon fixed customer. For more information, read [Creating an Amazon Fixed Customer](#).
- If NetSuite Connector is configured to import Amazon orders in a pending state, an improper pending order setup can cause this error. Pending orders do not contain complete customer information. Though the pending orders are later updated to provide complete information, the information is not available when the order is pending. Part of the pending order setup is to assign all pending orders to a fixed customer. If pending orders display an error, contact NetSuite Customer Support with the internal ID of the customer that you want to use as a fixed customer for your pending order.



Note: Configuring pending order sync is no longer supported.

- Customer information privileges are revoked for your Amazon credentials. For more information, read [Troubleshooting Incomplete Customer Information From Amazon MFN Orders](#).
- Shopify POS orders do not have the complete customer information. Designate the default values for names. For more information, read [Assigning Default Values to Shopify Point-of-Sale \(POS\) Orders in NetSuite Connector](#).
- For any other connector, where the complete customer information is not passed through API, set NetSuite Connector to always import orders to the same fixed customer. For more information, read [Assigning a Fixed Customer to All Orders in NetSuite Connector](#).

Troubleshooting Character Limit Errors in NetSuite

When syncing some fields in your storefront, you may face a character limit error. You can resolve this error using one of the following methods:

- Adjusting NetSuite field to increase the character limit
- Truncating the value

Adjusting NetSuite Field to Increase the Character Limit

If you are mapping to a custom field, change the field type to field with a higher character limit. Then update the NetSuite Connector mappings to point to the new field. For more information about field character limits for different field types, read the help topic [Field Type Descriptions for Custom Fields](#).

Truncating the Field Value

In this method, create a logic mapping in NetSuite Connector. This method does not require any changes in NetSuite. This method lets you truncate the values coming from the storefront so that they never exceed the field lengths determined by NetSuite.

To truncate the field value in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Mappings > Orders.

4. Click the **Order** tab.
5. Locate the mapping that is running into character limit error. For example, the **Memo** field.
6. From the dropdown list in the **Mapping Type** column for the mapping, select **Logic**.
The field in the <Marketplace or cart> **Field / Fixed Value** column changes to **Logic Mapping – Click to View/Edit**.
7. Click **Logic Mapping – Click to View/Edit**.
8. Make the following settings in the Edit Logic Mapping window:
 - a. From the **Value Type** field, select **Calculate Value**.
 - b. From the first dropdown list in the **Set Value To** field, select **Order Header Value**.
 - c. From the second dropdown list, select the storefront field from which you want to pull data.
 - d. From the third dropdown list, select **Substring**.
Substring is the mapping rule that tells NetSuite Connector what to do with the data that is coming from the storefront. This selection, adds four more fields to the **Set Value To** field.
 - e. From the fourth dropdown list, select **Fixed Value**.
 - f. In the fifth field, enter **0**.
 - g. From the sixth dropdown list, select **Fixed Value**.
 - h. In the seventh field, enter the maximum field length from the field in NetSuite.
If you do not know the length, check the error message for this error.
9. Click **Save Changes**.
10. Click **Save**.

Troubleshooting Issues When Using Avalara or Scripts on Sales Orders

This topic discusses the following error:

USER_ERROR - You cannot edit the end of group line. You must delete the group

This error usually appears when you use Avalara. If you are not using Avalara and still see this error, make sure you do not have any other scripts running against sales order transactions. These scripts could possibly be trying to modify the line items of the order when the order is created.

If you are using Avalara, the error usually indicates that you should update the Avalara bundle. If updating the Avalara bundle does not resolve the issue, create the order in NetSuite manually and check if you encounter the same error. You may receive a more comprehensive error to identify and fix the problem. If that also does not work, contact NetSuite Customer Support or Avalara support for further assistance.

Troubleshooting Connection Time Out Issues

To troubleshoot this connection time out issue, do the following:

- Ensure that you have NetSuite Connector IP addresses whitelisted.
Refer to the list of IP addresses that NetSuite Connector requires to be whitelisted to make a connection, see [NetSuite Connector IP Addresses to Safelist \(Allowlist\)](#).
- Ensure that your NetSuite Account Number is correct in NetSuite Connector.
Your NetSuite Account Number is the number at the beginning of the NetSuite URL.

Ensure that the number in the URL matches what is specified in the NetSuite Account Number field in NetSuite Connector. The exception here is that if you use a sandbox or release preview in NetSuite, you may have to substitute a dash for an underscore. For example, if 12345-RP is indicated in the URL, you should enter 12345_RP in the NetSuite Account Number field in NetSuite Connector.

- Try regenerating the NetSuite Access Tokens.

After generating the new access token, copy it and then log in to your NetSuite Connector account. Go to NetSuite > Settings > Credentials and enter the NetSuite Token ID and NetSuite Token Secret into the corresponding fields. Click Save and Test Connection. You should receive a 'Test Successful' response.

Troubleshooting Sync Script Errors

If your NetSuite configuration setup uses many custom fields, your Map/Reduce (MR) Sync script instance that updates NetSuite Connector data may encounter the following error:

SyntaxError: Invalid String [at JSON.parse (native), at Object.getInputsData]

This error occurs when the JSON file is over one million characters long. This error could result in orders not syncing into NetSuite Connector properly.

To avoid this error, lower the amount of data that is synced at the same time.

To lower the amount of data being synced at the same time:

1. Go to Customization > Scripting > Scripts.
2. In the **FA | MR Sync – Update FA** script row, click **Deployments**.
3. In the **customdeploy_fa_mr_sync_update_fa** row, click **Edit**.
4. Click the **Parameters** tab.
5. In the **Amount of Records to Sync Per MR Run** field, enter **10**.



Note: The default value of this field is 20. If the issue persists after lowering the value to 10, set the value to 5. If lowering to 5 does not resolve the issue, contact NetSuite Customer Support.

6. Click **Save**.

Integrating NetSuite Connector with Supported Storefronts or 3PLs

This chapter discusses authorizing NetSuite Connector on your storefronts or 3PL accounts.

The following topics cover the integration of each account with NetSuite Connector:

- [Integrating NetSuite Connector with Amazon](#)
- [Integrating NetSuite Connector with eBay](#)
- [Integrating NetSuite Connector with Magento 2](#)
- [Integrating NetSuite Connector with Shopify](#)

Integrating NetSuite Connector with Amazon

To integrate NetSuite Connector and Amazon, authorize NetSuite Connector on your Amazon seller account.

To authorize NetSuite Connector on Amazon Vendor Central Account

1. Log in to [app.farapp.com](#).
2. Select Amazon connector and then select the relevant account.
3. Go to Settings > Credentials.
4. Select your country from the **Seller Central Site** list.
5. Click **Login with Amazon**.



Note: You will be navigated to the Amazon Seller Central login page. If you are not currently logged in to Amazon Seller Central, you will need to log in.

6. Follow the instructions on the Amazon Seller Central website.

You will be brought back to the NetSuite Connector page with a success or failure message. If you get a failure message, contact NetSuite Customer Support.

Integrating NetSuite Connector with eBay

To integrate NetSuite Connector and eBay, authorize NetSuite Connector on your eBay seller account. The authorization is valid for one year. When the authorization expires, follow the same steps to reauthorize the account.

Authorizing NetSuite Connector on eBay Seller Account

If you have multiple eBay accounts, log out of all your accounts before connecting your account to NetSuite Connector. Also, if you are authorizing multiple accounts, make sure you log off the current authorized account before authorizing the next account.

To authorize NetSuite Connector on eBay Seller Account

1. Log in to app.farapp.com.
 2. Select eBay connector and then select the relevant account.
 3. Go to Settings > Credentials.
 4. Click **Authorize Account**.
- The eBay website opens.
5. Follow the instructions on the eBay website.

You will be brought back to the NetSuite Connector page with a success or failure message. If you get a failure message, contact NetSuite Customer Support.

Integrating NetSuite Connector with Magento 2

To connect your Magento 2 account to NetSuite Connector, you must fill out the credentials on your Magento 2 connector's Credentials page in NetSuite Connector.



Note: Note: if you are using a firewall for your Magento 2 account, you must disable it so that NetSuite Connector can connect with Magento 2.

To specify Magento 2 Connector Credentials

1. Log in to NetSuite Connector.
 2. Go to Settings > Credentials.
 3. On the Credentials page, in the **Magento 2 Username** field, enter your username.
 4. In the Magento 2 Password field, enter your password.
- The username and password will be credentials of the Magento 2 Admin User account that can use to connect NetSuite Connector to Magento 2.
5. In the **Magento 2 Endpoint** field, enter the URL of your site.
 6. If your Magento 2 account uses two-factor authentication (2FA), in the **Magento 2 Uses 2FA** field, select **On**; otherwise, select **Off**.
 7. (Optional) If you are using 2FA, in the **Magento 2 Secret Key** field, enter the secret key that you used when setting up 2FA for Magento 2 for the first time.
- If you are using an existing Magento 2 administrator user, you can reset your 2FA for the user. The Magento 2 Secret Key is a string of text displayed on the 2FA setup screen. It is only displayed once, and after a new user is created, if 2FA is enabled for the Magento 2 store. The first time the user tries to log in, a message is displayed instructing to set up 2FA. The user will also receive an email with a link that opens a screen showing a QR code to be scanned. The screen also displays the Magento 2 Secret Key. This key must be copied and entered on the NetSuite Connector Magento 2 credentials page. This is the case if you are using Google Authenticator, but this may be different if you are using a different authenticator.
8. Click **Save and Test Connection**.

Integrating NetSuite Connector with Shopify

Use the following procedure to access Shopify account.

Before proceeding, make sure you have added the Shopify connector to your account. To add a new connector, contact your NetSuite account manager.

To access Shopify account:

1. Log in to app.farapp.com.
2. Select the Shopify connector and account from the left menu.
3. Go to Settings > Credentials.
4. In the **Shopify Shop Name** field, enter the name of your shop.



Note: Shop name is the name that goes in the URL `https://[yourshopname].myshopify.com`. Do not enter the full URL in the field, just enter the shop name portion of the URL.

5. Click **Save Settings**.
6. Click **Authorize Account**.

You will be navigated to Shopify website. Follow the instructions on Shopify site and when done you should get a success message in NetSuite Connector. If you get a failure message, contact NetSuite Support.

NetSuite Connector Sync Management

This chapter covers details on managing different sync types in NetSuite Connector.

- [Managing Sync Operations in NetSuite Connector](#)
- [Disabling a Mapping in NetSuite Connector](#)
- [Order and Product Sync Error Messages](#)
- [NetSuite Connector Order Sync Management](#)
- [NetSuite Connector Refund Sync Management](#)
- [NetSuite Connector Fulfillment Sync Management](#)
- [NetSuite Connector Product Sync Management](#)

Managing Sync Operations in NetSuite Connector

This topic covers the following sync operations:

- [Enabling or Disabling Syncs](#)
- [Manually Triggering a Scheduled Sync in NetSuite Connector](#)
- [Stopping and Starting Syncs](#)
- [Viewing the Date, Time, and Frequency of Syncs](#)
- [Order and Product Resyncing After Storefront or NetSuite Credential Updates](#)

Enabling or Disabling Syncs

NetSuite Connector lets you enable or disable different syncs between a storefront and NetSuite.

To enable or disable a sync:

1. Log in to [app.farapp.com](#).
 2. Select the connector and the relevant account.
 3. Go to Data Flows > Manage Data Syncs.
-  **Note:** You can also access the page from the dashboard by clicking the pencil icon in the status column for the desired connector and account.
4. On the Overall Status page, enable or disable the desired syncs.

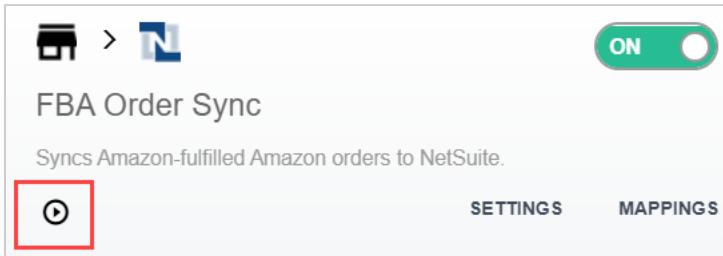
Manually Triggering a Scheduled Sync in NetSuite Connector

You can manually trigger a sync in NetSuite Connector after the sync is enabled. This will run the corresponding sync immediately rather than waiting for the next scheduled run.

To manually trigger a sync:

1. Log in to NetSuite Connector.
2. Select the desired connector and account.

3. Go to Data Flows > Manage Data Syncs.
4. Locate the sync you want to run.
5. Select the **Start Sync** icon. The following screenshot shows the icon on the FBA Order Sync:



Stopping and Starting Syncs

In NetSuite Connector, you can stop and start syncs from the Data Syncs Dashboard.

To stop and start a sync:

1. Log in to app.farapp.com.
2. On the left panel menu, select the connector and account whose sync you want to stop or run.
3. Go to Data Flows > Manage Data Syncs.
4. On the Data Sync Dashboard page, toggle the On/Off switch, to stop or start a sync.

Viewing the Date, Time, and Frequency of Syncs

If you are running automated syncs with NetSuite Connector, you can view the date and time when the last successful sync ran, as well as the frequency of sync runs.

To view the date, time, and frequency of syncs:

1. Log in to app.farapp.com and select the relevant connector and account from the left panel.
2. Go to Data Flows > Manage Data Syncs.
- The Overall Status page opens.
3. Go to the status box of the sync type that you want to view the schedule of runs.
4. Hover over the **Play** button. The date and time of the last sync and its frequency is displayed.
Clicking the Play button runs the sync.

Order and Product Resyncing After Storefront or NetSuite Credential Updates

If the storefront or NetSuite is disconnected from NetSuite Connector due to invalid credentials, none of the NetSuite Connector syncs post to their respective platforms. After updating the credentials in NetSuite Connector and confirming your account is connected to NetSuite Connector, the syncs automatically resume and the data is posted again. You do not need to manually initiate the scheduled syncs or import or post any order, products, or order data. In most cases, NetSuite Connector will pick up from where it left off. However, for long disconnection periods, you may require help from NetSuite Customer Support.

Disabling a Mapping in NetSuite Connector

You can disable an order or product mapping from the mapping interface in NetSuite Connector.

To disable a mapping:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Perform one of the following tasks:
 - To disable an order mapping, go to Mappings > Orders.
 - To disable a product mapping, go to Mappings > Products.
4. Locate the mapping that you want to disable and in the mapping row, click the toggle button on the left side of the mapping.



Note: If your mapping does not have the toggle button, delete the mapping by clicking the **Delete** icon on the right-side of the mapping. Note that you should not delete the **Category** field mapping because that can break your sync.

5. Click **Save**.

Order and Product Sync Error Messages

When orders and products encounter syncing issues, NetSuite Connector notifies you through email. Make sure you have added your email address to the notification section for NetSuite and your connectors in NetSuite Connector. For more information, read [Notification and Email Settings for NetSuite Connector](#).

You can also check the error messages in NetSuite Connector. Hover over the order on the Orders page to view the error message. For example, in the following screenshot, you see the error message that the order is not posted to NetSuite because the last name and subsidiary are missing.

The screenshot shows the NetSuite Orders page. At the top, there are tabs for 'Orders' and 'Products'. Below the tabs, the breadcrumb navigation shows 'Shopify > All > Orders'. Underneath the breadcrumb, there is a 'BULK ACTIONS' dropdown set to 'Select' and a 'Search' input field. On the right, there are buttons for 'Filter: Open' (with a close icon), 'FILTER', and 'RETRIEVE'. The main area displays a table of orders. The columns are: CONNECTOR, ACCOUNT, ORDER ID, STATUS, DATE, CUSTOMER, and TOTAL. The first two rows show successful posts. The third row, for an order from 'Shopify' with 'Account6', has a red status cell containing 'Error Posting to NetSuite'. The fourth row, for another 'Shopify' order, has a tooltip explaining the error: 'NetSuite is rejecting the customer because "Last Name, Subsidiary" is missing. You'll either need to set up a Last Name, Subsidiary mapping to add the Last Name, Subsidiary to the customer or remove the requirement for the Last Name, Subsidiary field in NetSuite. Please see the following article for instructions for both options: https://netsuite.custhelp.com/app/answers/detail/a_id/100881'. The 'TOTAL' column for this row shows '\$109.50'.

The messages provide the reason along with the resolution. You may need to adjust a configuration or setting in NetSuite to resolve the error.

For more information about some common errors, read [Troubleshooting Common Order Sync Errors in NetSuite Connector](#).

The error messages on the product appear in a similar way on the Products page in NetSuite Connector.

NetSuite Connector Order Sync Management

This section covers the following topics about order syncs in NetSuite Connector:

- [Order Mappings in NetSuite Connector](#)
- [Mapping Order Location in NetSuite Connector](#)
- [Setting up Custom Order Mappings in NetSuite Connector](#)
- [Viewing Order Mapping in NetSuite Connector](#)
- [Setting Up Customer Deposits](#)
- [Mapping a Subsidiary on Orders in NetSuite Connector](#)
- [Setting Up Order Sync to Handle Multiple Subsidiaries in NetSuite](#)
- [Appending a Set Value to Order Numbers in NetSuite Connector](#)
- [Reloading List Values for Order Mappings in NetSuite Connector](#)
- [Assigning a Fixed Customer to All Orders in NetSuite Connector](#)
- [Selecting the First Available Lot Number on Items During Order Sync in NetSuite Connector](#)
- [Managing Order Discounts in NetSuite Connector](#)
- [Displaying a Discount Item or Rate Field on the Order Form for NetSuite Connector](#)
- [Handling Incorrect Tax Totals on Discounted Orders](#)
- [Ensuring That NetSuite and Storefront Order Totals Are Matching](#)
- [Posting an Order with Total Variance into NetSuite](#)
- [Configuring Accounting for NetSuite Connector Order Discounts](#)
- [Changing the User That is Assigned the Role for Token Authentication](#)
- [Writing to Custom Segment Fields on Orders in NetSuite Connector](#)
- [Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite](#)
- [Checking if an Order was Imported by NetSuite Connector](#)
- [Finding Order Data that NetSuite Connector Sent To NetSuite](#)
- [Finding an Available Order Data to Map in NetSuite Connector](#)
- [Editing an Order in NetSuite Connector](#)
- [Deleting and Canceling Orders in NetSuite Connector](#)
- [Enabling Order Cancellations](#)
- [Posting a Canceled Order](#)
- [Using Batch Operations in the NetSuite Connector Order Portal](#)
- [Changing Synced Orders](#)
- [Identifying the NetSuite Form that NetSuite Connector Uses for Order Sync](#)
- [Changing the Preferred Order Form for NetSuite Connector](#)
- [Changing the Script Execution Order in NetSuite Connector When TaxJar is Enabled](#)
- [Setting Order Filters in NetSuite Connector](#)
- [Checking for Item Filters in an Order Form Used by NetSuite Connector](#)

- Setting Order Cutoff Dates in NetSuite Connector
- Handling a High Volume of Orders
- Importing Orders from the Storefront in NetSuite Connector
- Setting Order Statuses for Automatic Import in NetSuite Connector
- Delaying Order Imports in NetSuite Connector
- Searching for an Order in NetSuite Connector
- Searching for Orders Posted By NetSuite Connector
- Setting Customers As Taxable By Default
- Handling Line Level Tax Rates in NetSuite Connector
- Configuring Value-Added Tax (VAT) in NetSuite Connector
- Omitting Remitted Tax in NetSuite Connector
- Assigning Items to Tax Groups When Syncing Orders in NetSuite Connector
- Configuring Gift Cards for NetSuite Connector
- Mapping the Shipment or Payment Methods in NetSuite Connector
- Handling Invalid Shipping Tax Code Reference Key in NetSuite Connector
- Fixing Incorrect Shipping Tax on the Order in NetSuite
- Troubleshooting TranID Field Visibility Error in NetSuite Connector
- Troubleshooting Negative Total Error
- Configuring Shipping for NetSuite Connector
- Posting an Order After Resolving the Error on the Order
- Exporting Order Sync Errors from NetSuite Connector
- Troubleshooting Order or Fulfillment Sync Errors in NetSuite Connector
- NetSuite Connector Order Sync FAQ

Order Mappings in NetSuite Connector

NetSuite Connector allows you to customize how your order data is mapped from a connector to NetSuite. This section covers the following topics about order mappings in NetSuite Connector:

- Mapping Types
- Creating or Editing Order Mappings
- Creating or Editing Order Line Mappings
- Creating or Editing Customer Mappings
- Testing Order Mappings

Mapping Types

Mapping types determine how the source data will be mapped to NetSuite. NetSuite Connector allows you to set the following mapping types:

- **Fixed** – This allows you to set a constant value to send to a NetSuite field. For example, if you want the memo field in NetSuite to always contain “NetSuite Connector Order,” then you should use this mapping type.
- **Order Header** – This allows you to map header fields in an order to a NetSuite field. For example, if you want the memo field in NetSuite to contain the order date, then you should use this mapping type.

- **Order Header With Translation** – This allows you to specify the value to send to NetSuite based on a set of conditions in fields in the marketplace or cart. You could think of this mapping as “if (marketplace field) equals (specific value), then send (another value) to NetSuite.” For example, if you want the memo field in NetSuite to map based on the state where the billing was made, you should use this mapping type. This allows you to specify that orders coming from California will say “Order Billed in CA” and orders coming from Texas will say “Order Billed in TX” in the NetSuite memo field. You can set as many conditions as you like to cover every possible value in your chosen marketplace or cart field.
- **Order Line** – This is similar to Order Header Mapping but maps line item fields in an order to NetSuite. For example, if you want to map some data about the item to a transaction line custom field, you should use this mapping.
- **Order Line With Translation** – This is similar to Order Header With Translation but maps line item fields in an order to NetSuite.

Creating or Editing Order Mappings

Order mappings are used for fields that apply to the whole order. These are usually header fields in the marketplace or cart.

To create or edit order mappings:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. On the Orders page, click the **Order** tab.
NetSuite Connector displays all the mappings for your channel or account.
5. Click **Add Mapping**.
The Add NetSuite Mapping window appears.
6. On the NetSuite Field list, do one of the following:
 - To map to a standard NetSuite field, select the field from the list.
 - To map to a custom NetSuite field or a custom segment, select **Custom Field** then enter the field ID you want to use.



Note: The custom field id should start with custbody. Custom segments have IDs that start with cseg. To get the field id of a field in NetSuite, read [NetSuite Connector Communication with Internal IDs and Field IDs](#).

7. Click **Add Mapping**, then **Close**.
The new mapping appears at the bottom of the mapping list.
8. Locate your mapping and from the Mapping Type column, choose one of the following types:



Note: Not all mapping types are available for all fields.

- **Fixed** – Sets a constant value that will always post to NetSuite.
- **Order Header** – Select from a list of header fields NetSuite Connector has pulled from an order.
- **Order Header With Translation** – Allows you to specify the value based on “if this, then that” conditions.
- **Logic** – Used for more complex mappings.

9. From the <marketplace or cart> field/Fixed Value column, select the field or value you want to post to NetSuite. The value here will depend on the mapping type you chose in the previous step:

Mapping Type	Action
Fixed	Enter a constant value.
Order Header	Select from the list of header fields.
Order Header With Translation	<ol style="list-style-type: none"> 1. Select from the list of header values. 2. Click Click to View/Edit. A mapping window appears. 3. Ensure to map the NetSuite value as needed for all existing rows, if any. 4. Click Add Row. 5. Enter the connector and NetSuite value. For example, if you selected the Location field and enter 1 in the connector value and 3 in the NetSuite value, whenever an order has the location 1, a value of 3 is posted in NetSuite. 6. Repeat steps 4–5 to map all possible values for the field. 7. (Optional) Enter a value in the Default Value field for NetSuite Connector to use when it finds an unmapped value. 8. Click Save.

10. Click **Save**.

Creating or Editing Order Line Mappings

Order line mappings are used to map to fields at the line item level in an order.

To create or edit order line mappings:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. On the Orders page, click the **Order Item** tab.
NetSuite Connector displays all the mappings for your channel or account.
5. Click **Add Mapping**.
The Add NetSuite Mapping window appears.
6. On the NetSuite Field list, do one of the following:
 - To map to a standard NetSuite field, select the order line item field from the list.
 - To map to a custom NetSuite field, select **Custom Field** then enter the field ID you want to use.

Note: The custom field id should start with custcol. To get the field id of a field in NetSuite, read [NetSuite Connector Communication with Internal IDs and Field IDs](#).

7. Click **Add Mapping**, then **Close**.
The new mapping appears at the bottom of the mapping list.
8. Locate your mapping and from the Mapping Type column, choose one of the following types:
 - **Fixed** – Sets a constant value that will always post to NetSuite.
 - **Order Header** – Select from a list of header fields NetSuite Connector has pulled from an order.

- **Order Header With Translation** – Allows you to specify the value based on “if this, then that” conditions.
 - **Order Line** – Select from a list of fields NetSuite Connector has pulled in an order line item.
 - **Order Line With Translation** – This is similar to Order Header With Translation but maps line item fields in an order.
 - **Logic** – Used for more complex mappings.
9. From the <marketplace or cart> field/Fixed Value column, select the field or value you want to post to NetSuite. The value here will depend on the mapping type you chose in the previous step:

Mapping Type	Action
Fixed	Enter a constant value.
Order Header or Order Line	Select from the list of header or line item values.
Order Header With Translation or Order Line With Translation	<ol style="list-style-type: none"> 1. Select from the list of header or line item values. 2. Click Click to View/Edit. A mapping window appears. 3. Ensure to map the NetSuite value as needed for all existing rows, if any. 4. Click Add Row. 5. Enter the connector and NetSuite value. For example, if you selected the Location field and enter 1 in the connector value and 3 in the NetSuite value, whenever an order has the location 1, a value of 3 is posted in NetSuite. 6. Repeat steps 4–5 to map all possible values for the field. 7. (Optional) Enter a value in the Default Value field for NetSuite Connector to use when it finds an unmapped value. 8. Click Save.

10. Click **Save**.

Creating or Editing Customer Mappings

Customer mappings are used for fields that apply to the customer of an order.

Note: This only applies to newly created customers. Existing customers will not be affected.

To create or edit customer mappings:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. On the Orders page, click the **Customer** tab.
NetSuite Connector displays all the mappings for your channel or account.
5. Click **Add Mapping**.
The Add NetSuite Mapping window appears.
6. On the NetSuite Field list, do one of the following:
 - To map to a standard NetSuite field, select the field from the list.

- To map to a custom NetSuite field, select **Custom Field** then enter the field ID you want to use.



Note: The custom field id should start with custentity. To get the field id of a field in NetSuite, read [NetSuite Connector Communication with Internal IDs and Field IDs](#).

- Click **Add Mapping**, then **Close**.

The new mapping appears at the bottom of the mapping list.

- Locate your mapping and from the Mapping Type column, choose one of the following types:
 - Fixed** – Sets a constant value that will always post to NetSuite.
 - Order Header** – Select from a list of header fields NetSuite Connector has pulled from an order.
 - Order Header With Translation** – Allows you to specify the value based on “if this, then that” conditions.
 - Logic** – Used for more complex mappings.
- From the <marketplace or cart> field/Fixed Value column, select the field or value you want to post to NetSuite. The value here will depend on the mapping type you chose in the previous step:

Mapping Type	Action
Fixed	Enter a constant value.
Order Header	Select from the list of header values.
Order Header With Translation	<ol style="list-style-type: none"> Select from the list of header values. Click Click to View/Edit. A mapping window appears. Click Add Row. Enter the connector and NetSuite value. For example, if you selected the Location field and enter 1 in the connector value and 3 in the NetSuite value, whenever an order has the location 1, a value of 3 is posted in NetSuite. Repeat steps 3–4 to map all possible values for the field. (Optional) Enter a value in the Default Value field for NetSuite Connector to use when it finds an unmapped value. Click Save.

- Click **Save**.

Apply an Order Mapping to All Storefronts in NetSuite Connector

You can apply a mapping to all storefronts by checking the box in the **Apply to All Channels** column on the Order Mappings list. To open the order mappings list, from the desired connector, go to **Mappings > Orders**, and click the **Order** tab. The same column also displays in the **Order Item** and **Customer** tabs.

Testing Order Mappings

NetSuite Connector allows you to test your mappings and ensure they are working properly. If your account is live and has orders coming in, you can test mappings using the Test Mappings button on the Orders Mapping page. If your account is not live or you do not have active orders, you can retrieve an order then use the test button.

Using the Test Mappings Button

If you have an existing order, you can use the Test Mappings button to ensure your mappings are working correctly.

To test order mappings using the Test Mappings button:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. On the Order Mappings page, click the **Order**, **Order Line** or **Customer** tab.
5. Click **Test Mappings**.
The Test Mappings window appears.
6. Enter an existing order ID already in NetSuite Connector, then click **Show Mapping**.
NetSuite Connector displays a list of NetSuite fields and the values to those fields based on current mappings.



Note: Testing mappings this way does not update the order in NetSuite.

Mapping Order Location in NetSuite Connector

When an order is synced, NetSuite Connector can assign orders to a location using a mapping. This assignment can be at the order header level or at line item level. The process of assigning the location is same in both cases, only change is the page in which the mapping is created in NetSuite Connector.

To map the order location:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Orders.
4. Perform one of the following actions:
 - To map the location at order header level, click the **Order** tab.
The Order Mappings page opens.
 - To map the location at line item level, click the **Order Item** tab.
The Order Item Mappings page opens.
5. Click **Add Mapping**.
The Add NetSuite Mapping window opens.
6. From the **NetSuite Field** list, select **Location**.
7. Click **Add Mapping**.
8. Click **Close**.
A new row for the **Location** field displays at the end of the mappings list.
9. In the newly added row in the mappings list, from the dropdown list in the **<marketplace or cart name> Field / Fixed Value** column (for example **Shopify Field / Fixed Value**), select the location.
Click the **Reload NetSuite Lists** button and then refresh the page.
10. (Optional) For Amazon connector, you will have an additional dropdown list that enables you to select whether the mapping applies to FBA orders, MFN orders, or both. Make appropriate selection in the list.
11. Click **Save**.

Setting up Custom Order Mappings in NetSuite Connector

NetSuite Connector provides you the capability to create custom order mappings with custom logic.

 **Note:** This is an advanced feature.

Overview

Custom mappings are created on the order mappings page for your connectors. To access these pages, log in to NetSuite Connector, select your connector and account in the NetSuite Connector menu on the left, and navigate to **Mappings > Orders**. Afterwards, click **Order** at the top of the page. You must know how to create basic mappings.

Order Header with Translation Mapping Type

Order Header with Translation mappings are used when you have certain values coming from the marketplace/cart that you want to map to a specific value in NetSuite for a field. In this example, there are different Store IDs in Shopify that you want to map to specific Departments in NetSuite.

To map order headers with translation mapping:

1. Click **Add Mapping** at the top right of the page.
 2. Select a NetSuite field from the dropdown. In this example select **Department**.
 3. Click **Add Mapping**.
 4. Click **Close**.
- The mapping should now show up under Order Mappings.
5. The mapping should now show up under Order Mappings.
 6. In the Mapping Type for the new Department mapping, select **Order Header with Translation**. This changes the **Field/Fixed Field Value** column to a drop-down selection.
 7. From the drop-down select **Store ID**.
- A pop-up will appear with mapping options.
8. Under the **Shopify Value** column, enter the **Store ID** value from Shopify and then the corresponding NetSuite value (such as the Internal ID of the Department you want to map the Store ID to).
 9. Click **Add Row**.
- This adds a new Store ID and NetSuite mapping value.
10. Click **Save** and then **Close**.

Order Header with Logic Mapping Type

In this example, orders are shipped within the US, and you want to will use a different order Location for California than you do for the rest of the US.

To map order headers with logic mapping:

1. Click **Add Mapping** at the top right of the page.
2. Select a NetSuite field from the dropdown. In this example select **Location**.
3. Click **Add Mapping**.
4. Click **Close**.

The mapping should now show up under Order Mappings.

5. The mapping should now show up under Order Mappings.

6. Complete the **Edit Logic Mapping** table:

- In the **Check If** row:

1. In the first column select a condition for the filter.

- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
- **Fixed Value** – A constant value (something that does not change from order to order. An example would be a number '1'.
- **Computed Value** – a value that is a result of a condition or an equation or operation.

2. In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:

- a. In the second column, select an order header value from the dropdown.
- b. In the third column, select a condition how to evaluate from the dropdown.
- c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
- d. In the fifth column, enter a value to evaluate against in the **Value 2** field.

- If **Fixed Value** is selected in the first column –

- a. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
- b. In the third column, select a condition how to evaluate from the dropdown.
- c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
- d. In the fifth column, enter a value to evaluate against in the **Value 2** field.

- If **Computed Value** is selected in the first column –

- a. In the second column, click **Click to View/Edit Value**.
- b. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
- c. Complete the form.

- If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
- If **Calculate Value** is selected:

- i. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.

- ii. In the second column:

- If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
- If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
- If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.

- iii. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
- iv. Click **Save Changes**.

■ In the **If True, Set Value To** row:

In the first column select a condition for the filter.

- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
- **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
- **Computed Value** – a value that is a result of a condition or an equation or operation.

In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:
 1. In the second column, select an order header value from the dropdown.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 1. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 1. In the second column, click **Click to View/Edit Value**.
 2. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 3. Complete the form:
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:
 - a. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - b. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - c. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - d. Click **Save Changes**.

- In the **Otherwise, Set Value To** row:

In the first column select a condition for the filter.

- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
- **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number ‘1’.
- **Computed Value** – a value that is a result of a condition or an equation or operation.

In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:
 1. In the second column, select an order header value from the dropdown.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 1. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 1. In the second column, click **Click to View/Edit Value**.
 2. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 3. Complete the form:
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:
 - a. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - b. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - c. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - d. Click **Save Changes**.

Note that in rows two and three, the Location we use is set when the condition is true, and the condition to use when it's false.

7. Click **Save Changes.**

Appended Values

You can string multiple values together into Appended Values, which allows you to create filters with values from NetSuite and are appended with a string of your choice.

To map order headers with appended values:

1. Click **Add Mapping** at the top right of the page.
2. Select a NetSuite field from the dropdown.
3. Click **Add Mapping**.
4. Click **Close**.
The mapping should now show up under Order Mappings.
5. The mapping should now show up under Order Mappings.
6. Complete the **Edit Logic Mapping** table:
 - In the **Check If** row:
 1. In the first column select a condition for the filter.
 - **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order. An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
 2. In the second, third, fourth, and fifth column:
 - If **Order Header Value** is selected in the first column:
 - a. In the second column, select an order header value from the dropdown.
 - b. In the third column, select a condition how to evaluate from the dropdown.
 - c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Fixed Value** is selected in the first column –
 - a. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - b. In the third column, select a condition how to evaluate from the dropdown.
 - c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Computed Value** is selected in the first column –
 - a. In the second column, click **Click to View/Edit Value**.
 - b. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 - c. Complete the form.
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:

- i. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - ii. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - iii. In the third column, select an operation to do to the values, conditions, or filters in steps 1-2.
 - iv. Click **Save Changes**.
- In the **If True, Set Value To** row:
- In the first column select a condition for the filter.
- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
- In the second, third, fourth, and fifth column:
- If **Order Header Value** is selected in the first column:
 1. In the second column, select an order header value from the dropdown.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Fixed Value** is selected in the first column –
 1. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Computed Value** is selected in the first column –
 1. In the second column, click **Click to View/Edit Value**.
 2. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 3. Complete the form.
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:
 - a. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.

- b. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - c. In the third column, select an operation to do to the values, conditions, or filters in steps 1-2.
 - d. Click **Save Changes**.
- In the **Otherwise, Set Value To** row:
- In the first column select a condition for the filter.
- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
- In the second, third, fourth, and fifth column:
- If **Order Header Value** is selected in the first column:
 1. In the second column, select an order header value from the dropdown.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Fixed Value** is selected in the first column –
 1. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Computed Value** is selected in the first column –
 1. In the second column, click **Click to View/Edit Value**.
 2. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 3. Complete the form:
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:
 - a. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - b. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.

- If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
- If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
- c. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
- d. Click **Save Changes**.

Note that in rows two and three, the Location we use is set when the condition is true, and the condition to use when it's false.

7. Click **Append Value**.

An **Appended With** table shows.

8. Complete the Appended With table:

- In the **Value Type** row:
 - **Set Value Conditionally** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Calculate Value** – a value that is a result of a condition or an equation or operation.
 - **Set Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
- In the second, third, and fourth row (the third and fourth row will appear depending on the selected option in **Value Type** in the first row):
 - If **Set Value Conditionally** is selected in the first column:

In the **Check If** row:

1. In the first column select a condition for the filter.
 - **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
2. In the second, third, fourth, and fifth column:
 - If **Order Header Value** is selected in the first column:
 - a. In the second column, select an order header value from the dropdown.
 - b. In the third column, select a condition how to evaluate from the dropdown.
 - c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Fixed Value** is selected in the first column –
 - a. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - b. In the third column, select a condition how to evaluate from the dropdown.

- c. In the fourth column, select either **Fixed Value**, **Order Header Value**, or **Computed Value**.
- d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 - a. In the second column, click **Click to View/Edit Value**.
 - b. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 - c. Complete the form.
 - o If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - o If **Calculate Value** is selected:
 - i. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - ii. In the second column:
 - ♦ If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - ♦ If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - ♦ If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - iii. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - iv. Click **Save Changes**.

In the **If True, Set Value To** row:

In the first column select a condition for the filter.

- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
- **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number ‘1’.
- **Computed Value** – a value that is a result of a condition or an equation or operation.

In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:
 1. In the second column, select an order header value from the dropdown.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, or **Computed Value**.
 4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 1. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 2. In the third column, select a condition how to evaluate from the dropdown.
 3. In the fourth column, select either **Fixed Value**, **Order Header Value**, or **Computed Value**.

4. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - If **Computed Value** is selected in the first column –
 1. In the second column, click **Click to View/Edit Value**.
 2. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 3. Complete the form.
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - If **Calculate Value** is selected:
 - a. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - b. In the second column:
 - ◆ If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - ◆ If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - ◆ If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - c. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - d. Click **Save Changes**.
 - If **Calculate Value** is selected in the first column, complete the **Set Value To** row:–
 1. In the first column, select either **Fixed Value**, **Order Header Value**, **Computed Value**
 2. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 3. In the third column, select a how to transform the value in the **Value 1** field from the dropdown.
 - If **Set Value** is selected in the first column –
 1. In the first column, select either **Fixed Value**, **Order Header Value**, **Computed Value**
 2. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
9. Click **Save Changes**.

Viewing Order Mapping in NetSuite Connector

You can preview the data NetSuite Connector sends to NetSuite for an order, by viewing the order mapping results. There are following two methods in which you can view the order mapping results:

1. From the Orders dashboard
2. From the Order Mappings page

To view data mapping from the Orders dashboard:

1. Go to Data Flows > Orders.

The Orders dashboard opens and displays the orders imported by NetSuite Connector.

2. Search the order for which you want to view the data mapping.

For more information, read [Searching for an Order in NetSuite Connector](#).

3. After locating the order, click the **Action** menu on the right of the order.

4. Select **Show Order Mapping**.

The order mapping test results appear.

To view data mapping from the Order Mappings page:

1. Go to Mappings > Orders.

2. Click the **Order** tab.

3. Click **Test Mappings**.

The Test Order Mappings window opens.

4. In the **Enter the ID of an <ConnectorName>/Default Order in FarApp** field, enter the order number.

5. Click **Show Mapping**.

The order mapping test results appear.

Order Mapping Test Results

The Test Order Mappings window displays the results of the mappings. The results have the following two sections:

1. **Customer Output** – Displays the customer mappings for the selected order. Following is a sample customer output:

Customer Output:

```
record: Record Type = "Customer"  
isPerson: true  
firstName: John  
lastName: Doe  
email: [REDACTED]  
taxable: true  
addressbookList:  
    addressbook:  
        addressbookAddress:  
            addressee: John Doe  
            addr1: [REDACTED]  
            city: [REDACTED]  
            zip: [REDACTED]  
            state: [REDACTED]  
            country: _unitedStates  
        defaultBilling: true  
        isResidential: true  
        defaultShipping: true  
subsidiary: Internal ID = "3"  
parent: Internal ID = "2504"
```

Order Output:

```
record: External ID = "Shopify-default-1034" Record Type = "SalesOrder"  
entity: Internal ID = "1960"  
shippingAddress:  
    addressee: John Doe  
    addr1: [REDACTED]  
    city: [REDACTED]  
    state: [REDACTED]  
    zip: [REDACTED]
```



Note: If the order uses a fixed customer, you will not see the Customer Output section.

2. **Order Output** – Displays the order mappings for the selected order.

```

Order Output:

record: External ID = "Shopify-default-1034" Record Type = "SalesOrder"
entity: Internal ID = "1960"
shippingAddress:
  addressee: John Doe
  addr1: [REDACTED]
  city: [REDACTED]
  state: [REDACTED]
  zip: [REDACTED]
  country: _unitedStates
shipIsResidential: true
billingAddress:
  addressee: John Doe
  addr1: [REDACTED]
  city: [REDACTED]
  state: [REDACTED]
  zip: [REDACTED]
  country: _unitedStates
tranDate: 2019-11-07T17:03:34Z
paymentMethod: Internal ID = "4"
shipMethod: Internal ID = "686"
itemList:
  item:
    item: Internal ID = "1158"
    quantity: 1
    price: Internal ID = "-1"
    amount: 1.0
    rate: 1.00
    isTaxable: true
    taxRate1: 9.0000
    location: Internal ID = "2"
    shippingCost: 0.00
    isTaxable: true
    taxRate: 9.0000
    customFieldList:
      customField: Field Name = "custbody_fa_channel_order"
        value: 1034
      customField: Field Name = "custbody_fa_channel"
        value: Shopify
      customField: Field Name = "custbody_fa_shipping_tax"
        value: 0.0000
      customField: Field Name = "custbody_fa_order_total"
        value: {"shippingTax": 0.0, "orderTotal": 1.09, "shippingCost": 0.0, "itemTotal": 1.0, "discountTotal": 0.0, "taxTotal": 0.09}
      customField: Field Name = "custbody_custbody Domestic_order"
        value: true
      customField: Field Name = "custbody_test"
        value: 5
    otherRefNum: 3

```



Note: When testing order mappings, the mapping results that appear are based on the mappings and order data in NetSuite Connector at the time of viewing. You do not necessarily see the mapping results of the time when the order was actually sent to NetSuite.

Setting Up Customer Deposits

When a customer pays advance for an order or other service, you record the funds as a customer deposit. In NetSuite, these payments are recorded in your general ledger as a liability until the goods or services

are actually delivered. These payments do not affect the customer's accounts receivable balance. When the order is filled, the deposit is applied against the invoice and the liability is canceled out.

NetSuite Connector supports posting customer deposits for your orders to NetSuite. However, the leading practice is to capture payment in your marketplace or cart at the time of order placement. The authorization and capture configuration must be set up with your payment gateway.

Before setting up customer deposits in NetSuite Connector, you must set up customer deposits in NetSuite. For more information, read the help topic [Customer Deposits](#). For help with some common customer deposit errors, read [Troubleshooting Customer Deposit Errors in NetSuite Connector](#).

After setting up customer deposits in NetSuite, follow this procedure in NetSuite Connector.

To set up customer deposit:

1. Log in to [app.farapp.com](#).
2. Select the appropriate connector and account.
3. Go to Settings > Other Transactions.
4. Click the **Deposit, Payment, and Check** tab.
5. In the Deposit, Payment, and Check Settings page, make the following settings:
 - a. To post the customer deposits against a specific account, in the **Account for Customer Deposits** field, enter the internal ID of the account.
 - b. To accept customer deposits against only one payment method, from the **Payment Method for Customer Deposits** box, select the payment method.
If you do not see the required payment method, refresh the list by clicking the **Refresh** icon.
 - c. To post a customer deposit for every order, check the **Post a Customer Deposit for Each <store>/Default Order** box.
Where <store> is the marketplace or cart name.
 - d. To post customer deposits to undeposited funds in NetSuite, check the **Post Customer Deposits to Undeposited Funds** box.



Note: Do not map both payment method and customer deposit to an order. If you do so, NetSuite rejects the customer deposit. To accept the order, either unmap the payment method or turn off customer deposit for NetSuite.

Troubleshooting Customer Deposit Errors in NetSuite Connector

For any customer deposit errors, make sure you have correctly configured the customer deposits. For more information, read [Setting Up Customer Deposits](#).

Following are the common customer deposit errors.

Error	Description
INSUFFICIENT_PERMISSION – You do not have permissions to set a value for element undepfunds due to one of the following reasons: 1) The field is read-only; 2) An associated feature is disabled; 3) The field is available either when a record is created or updated, but not in both cases.	Usually, this error indicates that you do not have the Account field set to show on your sales order form. Therefore, NetSuite Connector does not have the permission to write the customer deposits in the required fields. To resolve this error, check the NetSuite sales order form that NetSuite Connector is using and make sure that the Account field is visible in NetSuite. Then check the Show box for the field.

Error	Description
<p>Order posted successfully but customer deposit is failing to post because the sales order in NetSuite has payment method set but NetSuite only allows deposits for orders without a payment method and without terms. Please unset both from the order in NetSuite and the deposit will then be able to sync. Also, make sure you don't have payment method or terms mapped in FarApp in order for future deposits to sync.</p>	<p> Note: By default, NetSuite Connector uses your preferred sales order form.</p> <p>This error means that the order is posted to NetSuite with a payment method or terms mapped to the sales order. Because NetSuite does not permit to post a customer deposit when an order contains either a payment method or terms, the deposit cannot post. The solution is to delete the values from the payment method and terms fields on the sales order. Also, remove any payment methods and terms mappings in NetSuite Connector to let customer deposits post successfully for future orders.</p>

Mapping a Subsidiary on Orders in NetSuite Connector

All records referenced in an order must be in the same subsidiary as the order. If you have multiple subsidiaries, you must set up order mappings so that they post to the correct subsidiary. The mapping you create should set a subsidiary that is consistent for the customer, the order, and all the items in the order. Any inconsistency causes NetSuite Connector to reference invalid records when the order syncs.

To map a subsidiary on orders:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
- The Payment page appears by default.
4. To create a subsidiary mapping, do one of the following:
 - To add a mapping for orders, click the **Order** tab.
 - To add a mapping for customers, click the **Customer** tab.
5. Click **Add Mapping**.
6. From the Add NetSuite Mapping popup window, select **Subsidiary** in the **NetSuite Field** list.
7. Click **Add Mapping**, and then click **Close**.
8. Locate your new mapping at the end of the list. Under the **<marketplace or cart> Field/Fixed Value** column, select your desired subsidiary.
9. Click **Save**.
10. Repeat steps 4–9 on the other tab to ensure mapping consistency between orders and customers.

Setting Up Order Sync to Handle Multiple Subsidiaries in NetSuite

When creating a translation or logic mapping to sync a nonfixed subsidiary value on your orders and customers, you should use a separate connector account for each subsidiary.

For example, assume that your company has two subsidiaries, one for domestic operations and the other for international operations. In such case, you should process domestic orders in one connector account and international orders in another connector account in NetSuite Connector. There are two

options to achieve this setup. The following list explains the options with Shopify and Amazon as example storefronts and connectors:

- **Two Shopify stores and a Shopify connector account for each store in NetSuite Connector** – One Shopify connector account syncs the domestic subsidiary in NetSuite and the other Shopify connector account syncs the international orders to the international subsidiary in NetSuite.
- **Amazon.com processes the domestic items and Shopify processes the international orders**
 - The Amazon connector syncs to the domestic orders to the domestic subsidiary in NetSuite. The Shopify connector syncs the international orders to the international subsidiary in NetSuite.

Attempting to sync multiple subsidiaries through a single connector account in NetSuite Connector can result in NetSuite errors caused by referencing records in a conflicting or invalid subsidiary.

For instructions on setting a fixed subsidiary mapping, read [Mapping a Subsidiary on Orders in NetSuite Connector](#).

Appending a Set Value to Order Numbers in NetSuite Connector

You may want to add a prefix to the order numbers when sending the orders from NetSuite Connector to NetSuite. This requirement is common for Shopify, because Shopify adds order prefixes, but NetSuite Connector drops the prefixes when sending the orders into NetSuite. If you are mapping the order number to a field like **PO#** in NetSuite, you can add a logic mapping in NetSuite Connector. The logic mapping should be for appending a set value before or after the order number.

To append a set value before the order numbers:

1. Log in to [app.farapp.com](#).
2. Select the connector and then select the relevant account.
3. Go to Mappings > Orders.
4. Click the **Order** tab.
5. Click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
6. From the **NetSuite Field** dropdown list, select **PO #**.
7. Click **Add Mapping**.
A confirmation message appears indicating the field is added to the end of your list of mappings.
8. Click **Close**.
A new row for the **Order** mapping appears at the end of the mappings list.
9. In the newly added row, from the dropdown list in the **Mapping Type** column, select **Logic**.
10. Click the **Logic Mapping – Click to View/Edit** link in the **<Marketplace or Cart Name> Field / Fixed Value** column.
The Edit Logic Mapping window appears.
11. Make the following settings in the Edit Logic Mapping window:
 - a. From the **Value Type** field, select **Set Value**.
 - b. From the **Set Value To** dropdown list, select **Fixed Value**.
 - c. In the text field next to the **Set Value To** dropdown list, enter the prefix that you want to append to the order number.
 - d. Click **Append Value**.

- The window expands with few more settings in the Appended With section.
- e. (Optional) To add the value after the order number, switch the values that you enter before and after clicking **Append Value**.
 - f. In the Appended With section:
 - i. From the **Value Type** dropdown list, select **Set Value**.
 - ii. From the **Set Value To** dropdown list, select **Order Header Value**.
 - iii. From the dropdown list next to the **Set Value To** dropdown list, select **Order ID**.
 - g. Click **Save Changes**.
A confirmation message displays in the popup window.
 - h. Click the **Close** button.
12. Click **Save**.

Reloading List Values for Order Mappings in NetSuite Connector

When you add new values to lists in NetSuite, they are not immediately updated in NetSuite Connector. For example, you might want to add a new department, location or sales representative. In order for these new values to be available in NetSuite Connector, you will need to reload all list values from NetSuite.

To reload NetSuite list values:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. Under the **Order**, **Order Item** or **Customer** pages, click **Reload NetSuite Lists**. This will load the values of NetSuite lists including built-in lists such as classes and locations and any custom lists you've defined.
5. A banner message will display at the top of the screen after the reload is successful. Refresh your browser page to see the newly added list values.

Assigning a Fixed Customer to All Orders in NetSuite Connector

You can configure NetSuite Connector to send all orders to NetSuite using the same customer. This approach is useful when you do not want to have a new customer record created for each new customer on an order.

You must first configure the customer in NetSuite and then in NetSuite Connector.



Note: This topic applies to all connectors, except Amazon connector. For Amazon connector, see [Assigning a Fixed Customer to All Amazon Orders in NetSuite Connector](#).

Configuring Fixed Customer in NetSuite

You can either use an existing customer or set up a new customer as a fixed customer. Make sure the customer is taxable so that both taxable and non-taxable orders can be assigned to the customer.

For more information, read [Setting Customers As Taxable By Default](#). After setting up the customer in NetSuite, note the customer's internal ID.

To locate the internal ID, read [Finding the Internal ID of Records](#).

Configuring Fixed Customer in NetSuite Connector

After you configure the fixed customer in NetSuite, configure the fixed customer in NetSuite Connector.

To configure a fixed customer in NetSuite Connector

1. Log in to [app.farapp.com](#).
 2. Select the appropriate connector and account.
 3. Go to Settings > Orders.
- The Order Settings page opens.
4. In the General tab, click **Advanced Options**.
 5. In the **Post All Orders Against Customer** field, enter the internal ID of the customer that you want to use as your fixed customer.



Note: The Amazon connector has two settings, one for MFN orders and another for FBA orders. Enter the internal ID in the field that you use.

6. Click **Save**.

All future orders for the selected connector and account combination are sent to NetSuite using the internal ID of the customer.

Selecting the First Available Lot Number on Items During Order Sync in NetSuite Connector

Certain items in NetSuite require additional data referred to as **inventory detail**. For such items, NetSuite Connector must provide this data on the sales order or cash sale during order sync for NetSuite to accept the order posting. To handle these items, you should configure NetSuite to automatically provide the inventory detail. Otherwise, NetSuite Connector redirects NetSuite to the first available lot number. If NetSuite provides the requested data, you can use that value for the inventory detail. Ensure that you also have a line item level location field mapping so that NetSuite Connector can determine the location of the first available lot number.

Configuring NetSuite this way can ensure obtaining the detail you expect, otherwise, you can perform troubleshooting to determine what happened in NetSuite. However, this approach is not possible with NetSuite Connector.

To enable NetSuite Connector to select the first available lot number on items that require inventory detail during order sync:

1. Log in to [app.farapp.com](#).
2. Select the relevant connector and account.
3. Go to **Settings > Orders**.
4. On the General Order Settings page, click **Advanced Options**.
5. Go to Inventory Detail (Lot Number) Settings.

6. Check the box of the transaction record type that you want to enable the functionality for.
7. Click **Save**.

A message appears and confirms that the functionality is enabled.

Managing Order Discounts in NetSuite Connector

The leading practice for discounts is to post all discount, promotion, or coupon amounts to a generic discount item on the order header in NetSuite. This approach ensures that the discount total on the order posts to NetSuite, resulting in the NetSuite order total matching the marketplace or cart order total. Posting the discount to the order header displays the discount total in order summary enabling you to easily view the discount on the order.



Note: If you install the NetSuite Connector SuiteApp, the SuiteApp creates and configures the generic discount item automatically. In that case, the following procedures are not required.

Creating a Discount Item

Create a discount item in NetSuite and configure the same in NetSuite Connector.

To create a discount item:

1. Go to Lists > Accounting > Items > New.
2. Select **Discount**.
3. In the **SKU** field, enter the SKU value that you designated for order sync.
For more information on the SKU field, read the help topic [Verifying NetSuite SKU Matches the Storefront SKU](#).
To use NetSuite Connector's auto-correction functionality to compensate for rounding differences in discount calculations between marketplace or cart and NetSuite, in the SKU field, enter **FARAPP_GENERIC_DISCOUNT**.
4. In the **Rate** field, enter **1**.
NetSuite connector will overwrite this rate in each order to match the discount on the order.
5. Check the **Apply Before Sales Tax** box.



Important: If available, set the **Respect Discount Item Tax** preference in the Set Up Taxes page (Setup > Accounting > Set Up Taxes).

6. Click **Save**.

For information on configuring the accounts on discount items read [Configuring Accounting for NetSuite Connector Order Discounts](#).

For gift cards and gift certificates, make sure you create a different discount item.

For more information, read [Gift Cards and Gift Certificates in NetSuite Connector](#).

To designate an account for the discount item, read [Configuring Accounting for NetSuite Connector Order Discounts](#).

Locating the Internal ID of the Discount Item

Note the internal ID of the discount item.

To locate the Internal ID of the Discount Item

1. Go to List > Accounting > Items.
2. In the Filters section, from the **Type** list, select **Discount**.
3. From the search results, locate the discount item.
4. From the **Internal ID** column, note the internal ID of the discount item.

If you do not see the **Internal ID** column, configure NetSuite to display the internal IDs. For more information, read [Viewing Internal IDs in NetSuite](#).

Configuring Discount Settings in NetSuite Connector

Each connector and account combination can have different discount settings. Make sure you are accessing the right connector and account when configuring these settings.

To configure discount settings in NetSuite connector:

1. Log in to app.farapp.com.
2. Select the appropriate connector and account.
3. Go to Settings > Orders.
4. Click the **Discount** tab.
5. In the **Default Discount Item to Post to (Internal ID)** field, enter the internal ID of the discount item.
By default, NetSuite Connector applies discount at the order header level.
6. (Optional) To apply discount at line item level, click **Line Items** for this setting.



Note: If you are configuring a gift certificate or gift card discount item, do not post both the default discount item and gift card or gift certificate discount item to the header level. You can post only one discount item at the header level.

7. Click **Save**.

Displaying a Discount Item or Rate Field on the Order Form for NetSuite Connector

To sync discounts at the header level of an order, the form that NetSuite Connector posts to NetSuite must have fields containing the following field IDs:

- discountitem
- discountrate

These fields are usually displayed on the NetSuite user interface as **Discount Item** and **Discount Rate**.

To display the discount item or rate field on the order form:

1. Go to Customization > Forms > Transaction Forms.
2. Click the **Edit** link of the relevant record and form.
NetSuite Connector most commonly syncs to the sales order form.

3. Go to the **Promotions** subtab and then click the **Items** subtab.
4. If **Discount Item** and **Rate** options are not under the **Items** subtab, check under the **Promotions** subtab. Check the **Show** boxes for **Discount Item** and **Rate** (or **Discount Rate**).
5. Click **Save**.

Handling Incorrect Tax Totals on Discounted Orders

If NetSuite Connector shows correct discount amounts while you find that the Order Summary in NetSuite displays an incorrect tax amount you will need to:

- Verify NetSuite discount Item configuration
- Verify Country Tax setup

Verify NetSuite Discount Item Configuration

Make sure your discount item is set to apply tax correctly in NetSuite. For discount items used for general discounts, it should apply before sales tax, for discount items used for gift certificates it should be set to not apply before sales tax. For more information on the proper configuration of the tax configuration in NetSuite, read the following topics:

- For standard discounts, read [Configuring the Discount Item in NetSuite Connector](#).
- For gift certificate discounts, read [Configuring Gift Cards for NetSuite Connector](#).

Verify Country Tax Setup

Some countries in NetSuite (including The United States) have a setting that controls whether the specific discount item's tax setting is followed when applying the discount. the following describes the steps for you to see if this setting is on:

To verify country tax setup:

1. Go to Set Up > Accounting > Set Up Taxes.
2. Select the subtab having the country's name.
3. Check the **Respect Discount Item Tax Preference** box.
4. Click **Save**.

Ensuring That NetSuite and Storefront Order Totals Are Matching

You must ensure that every order and total in NetSuite is imported correctly by NetSuite Connector to match what is posted on the storefront. To do this, follow these best practices:

- **The prices of the items on the order must match the prices set up in NetSuite.**

By default, the NetSuite Connector will override the prices set up for items in NetSuite and will use the item prices on the order. This ensures that item totals match. If you have setup your Order Item mappings to use a price level other than -1, or Custom, the NetSuite prices will be used instead, which can cause mismatched order totals. If you notice that the item prices on the order in NetSuite do not match those on the storefront order, go to Mappings > Orders to open the Order Item Mappings

page. Then, ensure that your mapping for **Price Level** is set to **Custom (IID: -1)** for this storefront or account. If you do not have a Price Level mapping yet, you can add one by clicking **Add Mapping Row**.

- **Configure discounts in the NetSuite Connector discount settings before applying them to orders.**

If there is a discount on a particular order, you must check if that discount is included in NetSuite. If not, you must set up the discount in the NetSuite Connector so that they import properly into NetSuite. For more information, refer to [Managing Order Discounts in NetSuite Connector](#).

- **The tax totals on the NetSuite order must match those from the channel order.**

For taxes to import correctly into NetSuite, you must configure them in NetSuite Connector. If the tax total in NetSuite is incorrect, refer to [Configure Taxes in Setting Up NetSuite Connector Order Sync](#).

- **Configure discount items in NetSuite.**

If an order has a discount, you must check the **Apply Before Taxes** box on the discount item in NetSuite. For more information, see the help topics [Applying Sales Tax or VAT to Discount Items](#) and [Applying VAT Before or After Discounts](#).

To review your configuration, refer to [Configuring Discount Settings in NetSuite Connector](#).

Posting an Order with Total Variance into NetSuite

If the NetSuite Connector order totals validator displays an error about a totals variance when sending an order to NetSuite, you can bypass that error and post the order. Note that bypassing the error does not correct the variance. The order is posted with the variance intact.

To post an order with variance into NetSuite:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. Locate the order with the total validator error and click the **Action** menu (pencil icon) for that order.
5. Select **Force Post to NetSuite**.

NetSuite Connector ignores the variance and posts the order to NetSuite.

Configuring Accounting for NetSuite Connector Order Discounts

When using discounts, you must configure the discount item's impact on the general ledger using one of the following ways:

- By configuring a discount account
- By configuring the discount item to be non-posting

Configuring a Discount Account

Post the discount against the general ledger by configuring a discount account. For example, if you apply a 10% posting discount to a \$100 item, the revenue amount posted to the associated revenue account for that item is \$100. Also, an offsetting debit amount of \$10 posts to the related discount account, such as sales discount account.

To configure a discount account:

1. Go to Lists > Accounting > Items.
2. Click **Edit** on the discount item row.
3. Click the **Accounting** subtab.
4. In the **Accounting** subtab, choose the **Account** option and select the discount account from the adjacent dropdown list.
5. Click **Save**.

Configuring a Discount Item to be Non-Posting

Post the net item amount after the discount against the general ledger by configuring the discount item to be non-posting. When a discount item is marked as non-posting, the item does not post as an individual transaction line. The item posts the net amount after the discount. When you create a sales order and add the non-posting discount after a line item, the discount is applied to the previous line item only. For example, if you apply a 10% non-posting discount to a \$100 line item, the revenue amount that posts to the associated revenue account for that item is \$90.

To configure a discount item to be non-posting:

1. Go to Lists > Accounting > Items.
2. Click **Edit** on the discount item row.
3. Click the **Accounting** subtab.
4. In the **Accounting** subtab, choose the **Non-posting** option.
5. Click **Save**.

Changing the User That is Assigned the Role for Token Authentication

When NetSuite Connector posts an order to NetSuite, the user indicated on the **System Notes** subtab on the sales order, is the one who generated the authentication tokens. This user is also granted the role associated with NetSuite Connector Web Services.

To change the user who is assigned the role for token authentication, refer to step **2** of the procedure in topic [Setting Up Token-Based Authentication for NetSuite Connector](#) and in step **2b**, you can change the user or employee. After changing the user, any future imports of order data will have the updated user listed on the sales order, under the **Set By** column.

Writing to Custom Segment Fields on Orders in NetSuite Connector

NetSuite Connector supports writing to custom segment fields but you must make sure the correct permissions are set on your custom segment.

To set up writing to Custom Segment fields:

1. Locate your custom segment in NetSuite.
2. Go to the **Permissions** subtab.

3. Set the **Default Record Access Level** and **Default Search/Reporting Access Level** are set to **Edit**.

Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite

If you want to add Promo Codes to your marketplace or cart, NetSuite Connector can map the promo codes from the marketplace or cart to the corresponding coupon code in NetSuite. This ensures that the transactions between the marketplace or cart and NetSuite match. For more information on promo code discrepancies, read [Promo Code Amount Discrepancies](#).



Note: It is recommended that discount, promotion, or coupon amounts should be set up to a generic discount item.

To set up matching Promo Codes from the marketplace or cart to NetSuite:

1. In NetSuite, do the following:
 - a. List Coupon Codes at the Promotion Level from List > Marketing > Promotion> New. This is critical because NetSuite Connector can only detect coupon codes listed at the Promotion Level. If your Coupon Code is not set to the Promotion Level, then NetSuite Connector will be unable to match the coupon codes and this feature will not work.
 - b. Select the type of promotion you want to create and configure the promotions as needed.



Important: Coupon Codes are case sensitive.

- c. Click **Save**. If you see that your Promotion has the Coupon Code listed when you view the list of your promotions, this means the Coupon Code is listed at the Promotion Level. The Promotions without the Coupon Code displayed will not be recognized by NetSuite Connector.
2. In NetSuite Connector, do the following:
 - a. After setting up the Coupon Codes in NetSuite, log in to [app.farapp.com](#) and go to the connector that you want to set up Promo Codes for.
 - b. From Settings > Orders > Discount > Advanced Options, check the box **POST COUPON CODE TO MATCHING PROMO CODE IN NETSUITE (APPLIES TO ALL SHOPIFY ACCOUNTS)**. This instructs NetSuite Connector to look for a matching Coupon Code in NetSuite if a Promo Code is used in the marketplace or cart.
 - c. Enter the internal ID of the default discount item that you configured, as detailed in topic [Managing Order Discounts in NetSuite Connector](#). If no Promo Code is found on the order, then NetSuite Connector will revert to using your default discount item.

Promo Code Amount Discrepancies

NetSuite Connector can handle discounts in different ways. The most common method is to use a default discount item, as detailed in [Managing Order Discounts in NetSuite Connector](#).

You can also enable an option to match to SuitePromotions codes, as discussed in [Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite](#). However, if you enable this setting, you may see errors about nonmatching promo codes.

When you sync orders with promo codes, NetSuite Connector includes only the promo code, but not the amount. If the promo amount in NetSuite does not match the promo amount in your marketplace/cart, then you may encounter issues in the order. Such cases cause discrepancies between the order total in NetSuite and the order total in the marketplace/cart. You may see an error similar to the following message:

NetSuite Connector found a match for "YOURPROMO" promotion code in NetSuite, but the discount amount in NetSuite (\$50.00) does not match the order (\$51.99).

Unlike default discount items, NetSuite Connector cannot overwrite the promo amount on the order to match the total amount on the marketplace/cart. You must ensure that the promo amount in the marketplace/cart exactly matches the promo amount in NetSuite.

Fixing Promo Code Amount Discrepancies

When the marketplace/cart and NetSuite amounts are different, issues with promo code amounts occur. These discrepancies are typically small amounts.

To fix the rounding issue, NetSuite implemented a workaround that will take the variance amount on a default discount item. NetSuite developed this solution for small discrepancies, but you may find that the solution works for other amount variances.

To use the workaround, you must do the following:

- Use a default discount item named FARAPP_GENERIC_DISCOUNT. For more information, read [Managing Order Discounts in NetSuite Connector](#).
- Both FARAPP_GENERIC_DISCOUNT and your promo code must use either the same account or no account (non-posting). On the promotion, you can select the FARAPP_GENERIC_DISCOUNT item as the value for the **Discount Item for Accounting** field. Your individual accounting needs may differ, but you can refer to the following options as examples.
 - On the discount item record, go to the Accounting subtab. If this discount item is configured as non-posting, the **Account** field should be blank.
 - On the promotion record, find the Primary Information field group. If this promotion does not have a discount item selected for accounting, the **Discount Item for Accounting** field should be blank.

Checking if an Order was Imported by NetSuite Connector

When you install the NetSuite Connector SuiteApp, the SuiteApp creates the following custom fields on sales orders and cash sales posted or synced by NetSuite Connector:

- NetSuite Connector Storefront (field ID: custbody_fa_channel)
- NetSuite Connector Storefront Order Number (field ID: custbody_fa_channel_order)

The **NetSuite Connector Storefront** field indicates the storefront from which the order was imported. The **NetSuite Connector Storefront Order Number** field indicates the storefront order number.

You should not change these fields or hide the subtabs that contain these fields. Otherwise, NetSuite Connector will be unable to post the storefront and order number details and show an error when trying to sync the orders. Ensure that the subtab is not hidden. Also, these fields are for internal use only and the content of these fields is subject to change anytime. Therefore, you should not use these fields for automation or reporting. You should rather create custom fields specifically for automation or reporting and sync the same data to those fields.

For more information about unhiding the custom fields, read [Troubleshooting Fields Not Being Visible or Editable Errors](#).

Finding Order Data that NetSuite Connector Sent To NetSuite

You can find the order information that NetSuite Connector sent to NetSuite by using the Web Services Usage Log.

To find order data that NetSuite Connector sent to NetSuite:

1. Find and open the sales order or cash sale that you want information on.
2. Click the **System Information** subtab and take note of the times that NetSuite Connector created or updated the order.
3. Go to Setup > Integration > SOAP Web Services Usage Log.
4. Use Available Selectors to specify the date range and the time that the order was updated or created.
5. Find the jobs indicating **Add** or **Update** in the Action column.
6. In the Request column, click **View** for those jobs.
7. Check the corresponding XML file that NetSuite Connector sent.

Finding an Available Order Data to Map in NetSuite Connector

You can check order data fields that are available to map from the Order Mappings page in NetSuite Connector.

To find the order data that is available to map:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Orders.
4. Click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
5. From the **NetSuite Field** list, select the field that you want to map.
6. Click **Add Mapping**.
The mapping is added to the Order Mappings page.
7. Click **Close**.
8. In the newly added mapping row, perform the following actions:
 - From the list in the **Mapping Type** column, select **Order Header with Transaction**.
 - From the list in the **Storefront Fixed / Field Value** column, select **Custom Field**.
The Map Custom Order Field popup window opens.
9. From the **Account** list, select the account.
10. From the **Order ID** field, enter the order ID.
11. Click **View Field Data**.
12. Click the plus icon next to the order data field that you want to map.

A mapping window opens. Each row in the window represents a field on the order that can be mapped.

Editing an Order in NetSuite Connector

You can edit order information directly in NetSuite Connector by following these steps.

To edit an order in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select your connector and account.
3. Go to **Data Flows > Orders**.
4. Search for the order that you want to edit by either scrolling the page or entering the order ID in the search bar.
5. Click the pencil icon on the right of the order and select **Edit**.
A popup screen is displayed with the order information you can edit.
6. Make the changes and click **Save Changes**.

Deleting and Canceling Orders in NetSuite Connector

The delete and cancel features serve different purposes in NetSuite Connector. This topic provides details and differences between the two functions.

Deleting an Order

When you delete an order in NetSuite Connector, the order data is removed only from NetSuite Connector. The order is not deleted from the storefront or NetSuite. Deleting the order does not prevent NetSuite Connector from syncing the order again, provided:

- the order still contains a status that NetSuite Connector syncs.
- the order falls within the current time period in which NetSuite Connector is looking for orders.

To delete an order from NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the required connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. In the order row that you want to delete, hover over the **Action** menu (pencil icon) and click **Delete from FarApp**.
5. In the confirmation alert, click **OK**.

The next time the order sync runs, depending on the connector, the order may be pulled into NetSuite Connector again.

To pull the deleted order back into NetSuite Connector, you can manually retrieve the order. However, you must first delete the originally synced order from NetSuite.

For more information, read [Importing Orders from the Storefront in NetSuite Connector](#).

Canceling an Order

When you cancel an order, NetSuite Connector stops processing the order and effectively ignores the order and stops syncing order updates from the storefront or NetSuite. NetSuite Connector stores the order data for the canceled order but considers the order closed and does not take any other action on the order. Cancel an order when you make changes to the order outside NetSuite Connector and no longer want NetSuite Connector to sync the order data.

To cancel an order:

1. Log in to app.farapp.com.
2. Select the required connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. In the order row that you want to cancel, hover over the **Action** menu (pencil icon) and click **Cancel**.

The order is canceled and the status of the order shows as **Cancelled**.

Reverting Canceled Orders

If you canceled an order in error, you can revert the canceled order.

To revert a canceled order:

1. Log in to app.farapp.com.
2. Select the required connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. In the order row that you want to revert, hover over the **Action** menu (pencil icon) and click **Uncancel**.

Deleting and Canceling Orders in Bulk

You can delete or cancel multiple orders in bulk using the **Delete Selected** and **Cancel Selected** options respectively from the **Bulk Actions** dropdown list.

To delete or cancel orders in bulk:

1. Log in to app.farapp.com.
2. Select the required connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. From the orders list, check the boxes adjacent to the orders that you want to delete or cancel.
5. Perform one of the following actions in the **Bulk Actions** dropdown list:
 - To delete the selected orders, select **Delete Selected**.
 - To cancel the selected orders, select **Cancel Selected**.

Enabling Order Cancellations

This section discusses order cancellation process from NetSuite to marketplace.

Order Cancellations from NetSuite to Marketplace or Cart

When an order is canceled in NetSuite, the NetSuite Connector is able to sync order cancellation back to the marketplace or cart and automatically close the order. This functionality works for the following connectors:

- BigCommerce
- Magento2
- Shopify
- Walmart
- WooCommerce

To enable order cancellations from NetSuite to marketplace or cart:

1. Log in to app.farapp.com.
 2. Select the appropriate connector and account.
 3. Go to Settings > Orders > General.
The Order Settings page opens.
 4. Click **Advanced Options**.
 5. Check the **If an Order is canceled in NetSuite, Cancel It in [Marketplace/Cart]** box.
-  **Note:** If you do not see this option for any of the above listed connectors, contact NetSuite support.
6. Click **Save**.

Posting a Canceled Order

If an order fails to post several times, NetSuite Connector automatically changes the order status to **Cancelled**. To make NetSuite Connector to try to post the order again, you must first undo the cancellation of the order. For information on undoing canceled orders, see [Reverting Canceled Orders](#).

Using Batch Operations in the NetSuite Connector Order Portal

In the NetSuite Connector order portal, you can cancel, undo cancellation, hold, release, post, or delete multiple orders at once.

To use batch operations in the NetSuite Connector order portal:

1. Log in to app.farapp.com and select the relevant connector and account from the left panel.
2. Go to Data Flows > Orders.
3. Display the orders you want by using the Filter menu at the top right of the table.
You can adjust the page size at the bottom of the table.
4. Check the boxes of the orders that you want to apply an action to.
Check the box at the header to automatically select all the orders on the page. Clear the box of an order that you want to exclude from an operation.
5. At the top left of the table, in the **Bulk Actions** field, click the bulk action that you want to apply for the selected orders.

Changing Synced Orders

The order corrections feature in NetSuite Connector lets you make changes to orders that were already synced to various endpoints. The correction flow is one-way depending on the type of correction described in the following sections.

Changes from the Marketplace or Cart to NetSuite

Direction (marketplace/cart -> NetSuite): From the marketplace/cart to NetSuite. This feature does not update order changes FROM NetSuite to the marketplace/cart.

By default, NetSuite Connector does not sync updates or changes to orders from your marketplace/cart to NetSuite. But your account can be configured so that if you update an order in the marketplace/cart, such as adding another item or changing the quantity on the existing items, NetSuite Connector can sync the updates to the sales order in NetSuite. This is possible as long as the order status is not Completed in either the marketplace/cart or NetSuite Connector. However, orders that are updated in NetSuite cannot be posted in the marketplace/cart.

The order corrections feature also supports order cancellations from the marketplace/cart to NetSuite. When an order is cancelled in the marketplace/cart, NetSuite Connector will detect the order status change and then sync the order updates to NetSuite to close all of the line items.

For more information about Order Cancellations, see [Enabling Order Cancellations](#).

Changes from NetSuite to 3PL

Direction (NetSuite -> 3PL): From NetSuite to the 3PL. This feature does not update order changes FROM the 3PL to NetSuite.

For 3PLs, a setting can be enabled that allows changes to an order in NetSuite to be updated in your 3PL. Similar to changes from the marketplace/cart to NetSuite, this will only work if the order status is not yet completed in NetSuite Connector or the 3PL.

If necessary, your account can be configured to allow an order cancellation in NetSuite to be reflected in the marketplace/cart or 3PL, and then automatically close out the order.

If you want your account to be configured to enable this feature, contact NetSuite customer support. Additional charges may apply if this feature is enabled.

Identifying the NetSuite Form that NetSuite Connector Uses for Order Sync

Although the records and forms NetSuite Connector syncs to can be changed with a mapping, the default behavior is as follows:

- NetSuite Connector syncs majority of the orders as sales orders, except the Point of Sale (POS), Amazon Fulfillment By Amazon (FBA), and Walmart Fulfillment Services (WFS) sync as cash sales.
- NetSuite Connector does not specify which forms to use for the record, so NetSuite uses the preferred form.

You can check the record or form to which NetSuite Connector is syncing using one of the following methods:

- Checking the order mappings by going to Mappings > Orders in your connector account.

- Viewing an order that is successfully synced and checking the record or form to which the order was synced. If a **customForm** field is being sent, then NetSuite Connector syncs to the form whose internal ID shows for that field. If **customForm** field is not sent, then NetSuite uses the preferred form for the record type being synced.

Changing the Preferred Order Form for NetSuite Connector

NetSuite Connector posts orders to your preferred form for the relevant transaction or other record type. For orders, this form will be your preferred sales order form. For Amazon FBA or Shopify POS orders, the form can be a cash sale form. To change the form that NetSuite Connector uses to post orders, either change the preferred form in NetSuite or create a mapping in NetSuite Connector.

For more information about changing the preferred form in NetSuite, read the help topic [Defining Preferred Forms](#).

Creating a Form Mapping in NetSuite Connector

This procedure changes the order form that NetSuite Connector uses to post the orders. This procedure can also be applied to the customer form by selecting **Customer** tab in step 4 below.

If you assign a form that is not relevant for a sync, like cash sale form for syncing sales orders, you get an error during sync.

To create a form mapping in NetSuite Connector:

- Log in to app.farapp.com.
- Select the connector and the relevant account.
- Go to Mappings > Orders.
- Click the **Order** tab.
- Click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
- From the **NetSuite Field** list, select **Custom Form**.
- Click **Add Mapping**.
The mapping is added to the Other Transaction Item Mappings page.
- Click **Close**.
- In the newly added row, in the <Marketplace or Cart Name> **Field / Fixed Value** column, if this field is not a dropdown list, click **Reload NetSuite Lists**.
After the reload process completes, the field changes to a dropdown list.
- Click the list in the **NetSuite Field/Fixed Value** column and select the form that you want to use.
- Click **Save**.

Changing the Script Execution Order in NetSuite Connector When TaxJar is Enabled

If you enable TaxJar in NetSuite, change the script execution order on the form (either sales order or cash sale) to which NetSuite Connector is posting the orders. The **FA | UE Order Total Validation** script should run after the **TaxJar** script.

The following procedure uses sales order as an example, but you can change the form type as needed.

To change the script execution order:

1. Go to Customization > Scripting > Scripted Records.
2. On the Sales Order row, click **Edit**.
3. In the **User Event Script** subtab:
 - a. Click the row that contains the **FA | UE Order Total Validation** script.
The cursor changes to a four-way arrow.
 - b. Drag the script to the end of the list.
4. Click **Save**.

Setting Order Filters in NetSuite Connector

NetSuite Connector enables you to setup order filters for your marketplace and carts.



Note: This is an advanced feature. Make sure to familiarize yourself with the order mappings section of the connector in NetSuite Connector.

Set up Order Filters

The following describes the steps for you to create an order filter in NetSuite Connector:

To setup order filters:

1. In NetSuite Connector, select a connection and go to **Settings > Orders**.
2. Click **Advanced Options**.
3. Click **Edit Order Filters**. This option will appear if the particular marketplace or cart supports order filtering.
4. Complete the **Edit Order Filters** table:
 - In the **Check If** row:
 1. In the first column select a condition for the filter:
 - **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order. An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
 2. In the second, third, fourth, and fifth column:
 - If **Order Header Value** is selected in the first column:
 - a. In the second column, select an order header value from the dropdown.
 - b. In the third column, select a condition how to evaluate from the dropdown.
 - c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.

- d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 - a. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - b. In the third column, select a condition how to evaluate from the dropdown.
 - c. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - d. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 - a. In the second column, click **Click to View/Edit Value**.
 - b. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 - c. Complete the form.
 - If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Order Filters (BETA)** form.
 - If **Calculate Value** is selected:
 - i. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - ii. In the second column:
 - If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - iii. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - iv. Click **Save Changes**.
- 5. In the **If True, Set Value To** row:
 - In the first column, select either **Set Order Filter** or **Computed Value**.
 - In the second column:
 1. If **Set Order Filter** is selected in the first column, select either **Don't Import Order** or **Import Order**.
 2. If **Computed Value** is selected in the first column, follow the steps for completing the computed value form in step 4–b.
- 6. In the **Otherwise, Set Value To** row:
 - In the first column, select either **Set Order Filter** or **Computed Value**.
 - In the second column:
 1. If **Set Order Filter** is selected in the first column, select either **Don't Import Order** or **Import Order**.
 2. If **Computed Value** is selected in the first column, follow the steps for completing the computed value form in step 4–b.
- 7. Click **Save Changes**.

Testing Order Filters

NetSuite Connector enables you to test your Order Filters after you have created them.

To test your filters

1. Log into app.farapp.com.
2. Select the third-party logistics provider in the navigation menu.
3. Select a relevant account.
4. Go to Settings > Orders.
5. Expand **Advanced Options**.
6. Click **Test Order Filters**.
7. In the **Test Order Filters** window, enter an order ID in the **Enter The ID Of Any Shopify/Default Order** field.
8. Click **Check Order**. A result will display.

Checking for Item Filters in an Order Form Used by NetSuite Connector

NetSuite Connector uses order forms to filter items that are displayed. You will be unable to create new orders if you try to create an order with an item that is not included in your filter's criteria. If you are unable to create a new order, you can check your order form if it is using a filter.

To check if an order form is using a filter:

1. Go to Customization > Forms > Transaction Forms.
 2. Locate the form you are syncing. This can either be a **Cash Sale** form or a **Sales Order** form, unless specified in NetSuite Connector. These two are commonly used when syncing with FBA and Shopify POS.
 3. Click **Edit**.
 4. Go to the **Sublist Fields** subtab.
 5. Click the **Item Filter** list.
- If the **Item Filter** list is empty, then no filter is applied. If the list has a selection then a filter is applied to your orders. You can empty the item filter list to test your form. You can also edit your filters instead to ensure your items that are sold in the marketplace or cart can be used on the order form.
6. Click **Save**.

Setting Order Cutoff Dates in NetSuite Connector

The order cutoff date is controlled by the **Import Orders Created After** setting in your NetSuite Connector account. This setting controls how far back NetSuite Connector looks for orders in the marketplace or cart when syncing orders.

To set the order cutoff date

1. Log in to app.farapp.com.
2. Select the appropriate connector and account.
3. Go to Settings > Orders > General.
The Order Settings page opens.
4. Click **Advanced Options**.
5. From the **Import Orders Created After** field, select the order cutoff date and time.
6. Click **Save**.

Initial Order Sync Activation Cutoff Date

After initial activation of your order sync, NetSuite Connector retrieves all live orders starting from a beginning date to the current date (including the current date). The beginning date is determined by the order cutoff setting. If order cutoff is not set, NetSuite Connector uses the internal default for the marketplace or cart.

Setting the Order Cutoff Date

Use the following procedure to set the order cutoff date.

To set the order cutoff date:

1. Log in to app.farapp.com.
2. Select the appropriate connector and account.
3. Go to Settings > Orders.
4. Click **Advanced Options**.
5. Click in the **Import Orders Created After** field and select the date up to which you want to retrieve orders back.



Note: The dates that exceed the maximum limit of cutoff date cannot be selected.

6. In the time selection window that opens, select the time on the previously selected date up to which you want to retrieve the orders back.

The time can be selected in the multiples of five minutes.

After selecting the time, the selected date and time display in the **Import Orders Created After** field.

7. Click **Save**.

After activating order sync, NetSuite Connector imports all the eligible orders up to the cutoff date and time set in the **Import Orders Created After** field.

Handling a High Volume of Orders

If you need to post large number of orders, such as 500 orders per hour or more, consider upgrading your account to NetSuite SuiteCloud Plus.

For more information, read <http://www.netsuite.com/portal/assets/pdf/ds-datacenter-premium-tiers.pdf>.

For more than 1000 orders per hour or more, contact NetSuite support.

Importing Orders from the Storefront in NetSuite Connector

NetSuite Connector enables you to manually import an order from the storefront.

To import an order in NetSuite Connector manually:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Orders.
4. Click **Retrieve**.

The Retrieve Order into FarApp window opens.

5. (Optional) To change the connector or account, change the values in the **Channel** and **Account** fields respectively.
 6. In the **Order ID** field, enter the order ID that you want to import.
 7. Click **Retrieve**.
- If the order is imported, a success message displays on the window. Else, an error message displays on the window.
8. Click **Close**.
 9. To post the order to NetSuite, click the **Action Menu** on the order row and select **Post Order to NetSuite**.



Note: If you change the storefront or account settings, navigate to the Orders page of the respective connector and account to see the imported order.

For more information, read [Specifying Payment Methods](#).

Setting Order Statuses for Automatic Import in NetSuite Connector

You can determine and set which order statuses are automatically imported by NetSuite Connector from your marketplace or cart. Most connectors allow you to do this under their Order Settings page. If a sync has run and an order was not imported, it is possible that the order status was one that NetSuite Connector does not automatically import.

To set order status fields to automatically import:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Settings > Orders.
4. The order status fields are listed in the **Automatically import orders in the following statuses** pane. Highlighted fields on the list are automatically imported. Hold down the Ctrl key to select multiple order status fields.



Note: If the pane is missing, the connector does not support setting order status fields for automatic import. Contact NetSuite Support for assistance.

5. Click **Save**.

Delaying Order Imports in NetSuite Connector

Sometimes, you may want delay importing of orders to NetSuite Connector by a few minutes. This delay may be required to run some process, such as a fraud detector in the storefront. You can achieve this delay using order filter and filter out orders that are not older than the desired minutes.

The following procedure is for filtering orders that are less than 10 minutes old for Shopify connector. You can adjust the timing, as required.

To introduce delay in order imports:

1. Log in to app.farapp.com.
2. Select the appropriate connector and account.
3. Go to Settings > Orders.
The **General** tab of Order Settings page opens.
4. Click **Advanced Options**.
5. Click **Edit Order Filters**.
The Edit Order Filters popup window opens.
6. Update the fields in the popup window as follows:
 - a. **Check if: Order Header Value –Current time – Is Greater Than – Computed Value**
 - b. **If True, Set Value To: Set Order Filter – Import Order**
 - c. **Otherwise, Set Value To: Set Order Filter – Don't Import Order**
7. Click the **Click to View/Edit Value** link that displays in the first row.
8. Update the fields in the popup window as follows:
 - a. **Value Type: Calculate Value**
 - b. **Set Value To: Order Header Value – Order Date – Add Time To Date (Seconds) – Fixed Value – 600.**



Note: 600 indicates 600 seconds, that is 10 minutes.

9. Click **Save Changes**.
10. Click **Close**.
11. In the Order Settings page, click **Save**.

Searching for an Order in NetSuite Connector

All orders imported in NetSuite Connector can be viewed in NetSuite Connector's Orders dashboard.

To search for an order in NetSuite Connector:

1. Log in to app.farapp.com.

2. Select the connector and the relevant account.
 3. Go to Data Flows > Orders.
- The Orders dashboard opens.
4. (Optional) To change the filter, point to **Filter** and from the filter list, select the required filter.
 5. In the **Search Orders** field, enter the order number.
 6. Press **Enter**.
- The results are displayed in the list.

Searching for Orders Posted By NetSuite Connector

An order will have the status **Order Posted, Waiting for Shipment** if it is imported into NetSuite. To find the order ID that is imported into NetSuite or the NetSuite internal order ID, by click **Show NetSuite Transaction ID** in the right-most dropdown in NetSuite Connector and click **OK**. You can also find the information within NetSuite without logging into app.farapp.com.

To find an order in NetSuite using Search Order ID field:

1. Go to Transactions > Sales > Enter Sales Orders > Search.
2. In the **View** dropdown, select the custom field you created for storefront order.
3. Select the order from the list.

Alternatively, you can search for a record's internal ID. See the help topic [How to Find a Record's Internal ID](#) for more information.

If you cannot find your order in NetSuite and NetSuite Connector shows an internal ID for the order, then the order must be deleted from NetSuite. For the order to reimport, you will need to post the order again from NetSuite Connector.

Setting Customers As Taxable By Default

 **Note:** This setup is only applicable for the United States.

Setting your customers as taxable ensures that proper tax calculation is recorded for all of your transactions for customers in the United States in NetSuite.

To set customers as taxable:

1. Go to Setup > Accounting > Set Up Taxes.
2. Select United States.
3. Enable the **Customers Default To Taxable** box.
4. Click **Save**.

Handling Line Level Tax Rates in NetSuite Connector

If you enable line level taxes, the tax rate may not transfer correctly from the sales order to the invoice or cash sale on billing the order. To fix this issue, set the **Match Line-Item Tax Rates Match with Tax Rates Created from Transaction (Beta)** box in NetSuite. You can check this box in the NetSuite Connector Setup page by going to Connector > Configuration > Setup.

Configuring Value-Added Tax (VAT) in NetSuite Connector

Before you configure VAT in NetSuite Connector, you must first confirm whether your storefront uses pre-VAT or post-VAT prices. To identify the VAT, you can import a sample order, edit the order in NetSuite Connector, and then compare the item prices. If you are using a marketplace, you can safely assume that prices already include VAT.

Follow this procedure to configure VAT on a specific account in NetSuite Connector.

To configure VAT in NetSuite Connector:

1. Log in to app.farapp.com.
 2. Select the relevant connector and account.
 3. Go to Settings > Orders.
 4. Edit the tax settings.
 - a. Go to the **Tax** tab.
 - b. To manage the VAT, do one of the following:
 - If your storefront provides the item prices with VAT, check the **Subtract VAT from Order Total When Posting** box. When you check this box, NetSuite Connector will subtract the VAT from the item price before posting.
 - If your storefront provides item prices without VAT, clear the **Subtract VAT from Order Total When Posting** box. When you clear this box, NetSuite Connector will not subtract the VAT from the item price before posting.
 - Leave the **VAT Rate For This Account** field blank. In most cases, NetSuite Connector detects the actual rate used in the storefront and does not need you to specify the tax rate. However, if you find that VAT rates do not calculate correctly, you can enter the rate in this field.
- In some instances, you may find that the storefront tax rate is different from the NetSuite tax rate. For example, in the storefront, you may see 20%, but in NetSuite, you see 19.9987% tax rate. This behavior is due to the different ways that NetSuite and storefronts round taxes. For more information, read the help topic [VAT and Tax Rounding](#).
5. Click **Save**.

Omitting Remitted Tax in NetSuite Connector

In many states, the marketplaces collect and remit tax on your behalf. Therefore, the leading practice is to sync orders from the states where the tax is remitted as non-taxable. If you account for the tax in NetSuite, you would be overstating the actual revenue because the tax revenue is actually never distributed to you.

To automatically omit tax:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Settings > Orders.
- The Order Settings page opens.
4. Click **Advanced Options**.

5. Check the **Omit Remitted Tax When Posting Amazon Orders** box.
6. Click **Save**.

Assigning Items to Tax Groups When Syncing Orders in NetSuite Connector

NetSuite Connector cannot assign items to tax groups. You should configure your taxes in NetSuite, including tax groups and tax codes. NetSuite will build the sales order based on your tax configuration and then NetSuite Connector will automatically overwrite that rate with the rate that was charged in the storefront.

For more information on tax codes, see the help topic [Tax Codes Overview](#).

Configuring Gift Cards for NetSuite Connector

There are two processes associated with gift cards or gift certificates:

- Selling the card or certificate
- Redeeming the card or certificate

Purchasing a Gift Card or Gift Certificate

The leading practice is to handle the sale of gift cards and certificates in the same way as any other item using an SKU. Make sure you set the gift card or certificate as non-taxable.

Redeeming a Gift Card or Gift Certificate

The leading practices for redemption is to map the value of redemption to a default discount item in NetSuite.

Creating a Discount Item

You must first configure the discount item in NetSuite.

To create a discount item:

1. Go to Lists > Accounting > Items > New.
2. Select **Discount**.
3. Enter the SKU in the **Item Name/Number** field.
4. In the **Rate** field, enter 1.

For each order, NetSuite Connector overwrites this value to the relevant amount.



Note: Do not check the **Apply Before Sales Tax** box.

5. Make other configuration settings that suit your accounting needs for gift certificates.
6. Click **Save**.

After saving, note the internal ID of the discount item.

Configuring the Discount Item in NetSuite Connector

Each connector and account configuration has separate discount settings.

To configure the discount item in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the appropriate connector and account.
3. Go to Settings > Orders.
4. Click the **Discount** tab.
5. In the **NetSuite Discount Item to Post Gift Certificates to (Internal ID)** field, enter the internal ID of the discount item you created in [Creating a Discount Item](#).
6. Click **Line Items**.
If you are using a default discount item, make sure the default discount item and gift certificate or gift card discount item both are not posted to the header level. In NetSuite, you can only post one discount item at a time to the header level.
7. Click **Save**.

Mapping the Shipment or Payment Methods in NetSuite Connector

This section details shipment and payment methods in NetSuite Connector and contains the following topics:

- [Mapping Order Shipment Methods in NetSuite Connector](#)
- [Mapping Order Payment Methods in NetSuite Connector](#)

Mapping Order Shipment Methods in NetSuite Connector

Create mappings for the shipment methods coming from the marketplace or cart to a corresponding shipment method in NetSuite. This enables you to track the shipment methods and costs on the associated order in NetSuite.

Mapping Options

NetSuite Connector displays shipment methods detected in NetSuite, such as Ground, Next Day, Priority, and so on. In NetSuite, these shipment methods are known as shipping items. These shipment methods appear in a dropdown list when mapping shipment methods from your marketplace, cart, or 3PL. In addition to the shipment methods detected in NetSuite, the following options are also available:

- **Do Not Post** –Default setting for an account. When a new shipment method is added from a marketplace, cart, or 3PL, NetSuite Connector shows an error message instructing you to map the shipment method. After mapping, NetSuite Connector uses that mapping for the marketplace, cart, or 3PL.
- **Not Mapped** – Selecting this option omits the shipment method from the order in NetSuite.



Note: If you do not map a shipment method, the shipping cost will be dropped when NetSuite imports the orders.

You can map shipment methods in the following ways:

- From the Shipment Mappings page
- Directly from the order

Mapping Shipment Method from the Shipment Mappings Page

The Shipment Mappings page enables you to view and edit all the shipment method mappings. On the Shipment Mappings page:

- The left column displays the shipment methods from the marketplace, cart, or 3PL. If an order with a shipment method is not imported, the left column will appear blank.
- The right column displays the NetSuite shipment method to which the marketplace, cart, or 3PL shipment method is mapped.
- The **Default Shipment Method to Post When No Match Found** field displays the default shipment method. When a new shipment method is added to the left column, the method selected in this field is selected in the right column by default.

To map a shipment method from the Shipment Mappings page:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Mappings > Orders.
 4. Click the **Shipping** tab.
- The Shipment Methods page opens.
5. When a new shipment method is added to the left column, in the right column, select the NetSuite shipment method that you want to map.



Note: If you do not see the required NetSuite shipment method in the list, refresh the NetSuite shipment methods. For more information, read [Refreshing Shipping Methods for NetSuite Connector](#).

6. Click **Save**.

The current and all future orders with the same shipment method will use the mapping you created.

Mapping Shipment Method from an Order

If you have an order with a shipment method not recognized by NetSuite Connector, you can directly set up a new mapping for the order.



Note: This method does not work for Amazon connector.

To map a shipment method from an order:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Orders.
4. Search the order that you want to map by entering the order ID in the **Search** field and pressing **Enter**.
5. On the order, click the **Action** menu (pencil icon) and select **Edit**.

The Order Details window opens.

6. Click the **Mapping** tab.

The shipment method for the order appears with the default mapping. Usually, the default mapping is **Do Not Post**.

7. From the **NetSuite Shipment Method** list, select the NetSuite shipment method to which you want to map the marketplace, cart, or 3PL shipment method.



Note: If you do not see the required NetSuite shipment method in the list, refresh the NetSuite shipment methods. For more information, read [Refreshing Shipping Methods for NetSuite Connector](#).

8. Click **Save Changes**.

The current and all future orders with the same shipment method will use the mapping you created.

Refreshing Shipping Methods for NetSuite Connector

If you add a new shipping method to NetSuite, NetSuite Connector automatically makes the new shipping method available for mapping within 24 hours. To map the shipping method immediately, you should refresh the Shipping Mappings page.



Note: To refresh the shipping methods, you must have the shipping account on the shipping items in NetSuite.

To refresh the shipping methods in the Shipping Mappings page:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Orders.
4. Click the **Shipping** tab.

The Shipping Methods page opens.

5. Click the **Refresh** icon next to any mapping on the page.
- Wait for some time for the update to complete.
6. Refresh the page in the browser.

The dropdown list should now include the updated list of shipping methods in NetSuite.

If you do not see the shipping method, read the topic [Showing Missing Shipment Methods in NetSuite Connector](#).

Showing Missing Shipment Methods in NetSuite Connector

The Shipment Methods page lists two different sets of shipping methods:

- The left column lists the shipments methods used in the storefronts that appeared on the synced orders.
- The right column contains a dropdown list containing shipping methods in NetSuite.

If you do not see your storefront shipment methods in the left column, it means that an order has not yet been imported with that shipment method. After an order imports with that shipping method, the shipment method gets added to the left column and you will be able to map it.

If you do not see your NetSuite shipping method in the dropdown list in the right column, check the following:

1. Refresh your shipment methods. For more information, read [Refreshing Shipping Methods for NetSuite Connector](#). After refreshing, check the **Shipment Method** dropdown list to see if the missing shipment method now appears.
2. If you do not see the shipment method after refreshing, check if your shipment method is set up correctly in NetSuite. For more information, read [Configuring Shipping for NetSuite Connector](#).
3. If you have multiple subsidiaries, make sure the subsidiaries used by NetSuite Connector are set in the **NetSuite Subsidiaries Used For FarApp Connectors** field. For more information, read [Setting Up NetSuite Connector for Syncs](#). If the missing shipment method is in a subsidiary that is not added in the field, enter the internal ID of that subsidiary and save the NetSuite Settings page.
4. After performing steps **2** and **3**, repeat step **1**.

For information about setting up shipping in NetSuite, read the help topic [Shipping](#).

Troubleshooting NetSuite Connector Shipping Method Errors

If you get an error mentioning that NetSuite is reporting invalid shipment method with internal ID #, ensure the following fields are set up correctly:

- **Display Name/Code** – Populate a value in this field.
- **Account (Shipping)** – Select an account from this field.
- **Subsidiary** – Select the subsidiary to which NetSuite Connector is syncing.
- **Charge tax on this shipping portion of item** – You must select a tax schedule. The tax schedule should appear irrespective of whether the shipping item should taxed or should not be taxed.

Mapping Order Payment Methods in NetSuite Connector

By default, in NetSuite Connector, you must create mappings for the payment methods coming from your marketplace or cart to a corresponding method in NetSuite. This mapping enables you to track the payment methods on the associated order in NetSuite.

Mapping Options

NetSuite Connector displays payment methods detected in NetSuite, such as Cash, Visa, Check, and so on. These payment methods display in a dropdown list when mapping payment methods from your marketplace or cart. In addition to the detected methods, there are following additional options:

- **Do Not Post** – Default setting for your account. When a new payment method comes from the marketplace or cart, NetSuite Connector shows an error and asks you to map the payment method. After mapping the payment method, NetSuite Connector uses the mapping for the selected marketplace or cart payment method in future.
- **Not Mapped** – NetSuite Connector omits the payment method from the order in NetSuite and payment method is not mapped.

Mapping Process

There are two ways in which you can map the payment methods.

- [Mapping from the Payment Mappings Page](#)
- [Mapping Payment Methods Directly from an Order](#)

Mapping from the Payment Mappings Page

You can view and edit all the payment methods from the Payment Mappings page. This method is easier when you have multiple payment methods to map or when you need to change existing mappings.

The Payment Mappings page displays the following columns:

- The left column displays the marketplace or cart payment methods. If an order with a payment method is not imported, the column appears blank.
- The right column displays the NetSuite payment method to which the marketplace or cart payment method is mapped. If an order with a payment method is not imported, the column appears blank.

To map payments from the Payments Mapping page:

1. Log in to app.farapp.com.
 2. Select the appropriate connector and account.
 3. Go to Mappings > Orders.
- The Payment Methods page opens.
4. Add the new payment method in the left column.
 5. Click the dropdown in the right column to select the NetSuite payment method to which you want to map the new payment method.



Note: If you do not see the required payment method, refresh the NetSuite payment method.

For more information, read [Refreshing NetSuite Payment Methods for NetSuite Connector](#).

6. Click **Save**.

The current order and all future orders with the same payment method will use the mapping that you created using this procedure.

Mapping Payment Methods Directly from an Order

If you have an order without a payment method not mapped, you can map the payment method directly in the order.



Note: This method does not work for Amazon orders.

To map payment methods directly from an order:

1. Log in to app.farapp.com.
 2. Select the appropriate connector and account.
 3. Go to Data Flow > Orders.
- The Payment Methods page opens.
4. On the right of the order, click the **Action Menu** (pencil icon).
 5. Click **Edit**.
- The Order Details window appears.
6. Click the **Mapping** tab.
 7. From the **NetSuite Payment Method** list, select the NetSuite payment method that you want to map.



Note: If you do not see the required payment method, refresh the NetSuite payment method. For more information, read [Refreshing NetSuite Payment Methods for NetSuite Connector..](#)

Click **Save**.

The current order and all future orders with the same payment method will use the mapping that you created using this procedure.

Specifying Payment Methods

The orders that NetSuite Connector imports to post to NetSuite should use a specific payment method. This payment method determines the account that NetSuite uses for the transaction. Also, most connectors indicate the payment method that a customer used for an order.

Every payment method used in a marketplace or cart (for example, Visa, MasterCard, or PayPal) should have a corresponding payment method in NetSuite. There are some marketplaces (like Amazon) where the payment method is not available, but you still need to create a payment method in NetSuite for that marketplace (for example, a payment method called 'Amazon').

You can add new payment methods from Setup > Accounting > Setup Tasks > Accounting Lists > New, and then selecting **Payment Method**. For more information, see the help topic [Creating a Payment Method](#). After adding payment methods, they will be available on your transactions. For example, on a sales order the **Payment Method** field is on the **Billing** subtab, and on a cash sale it is on the **Payment** subtab.

Refreshing NetSuite Payment Methods for NetSuite Connector

If you add a new payment method to NetSuite, NetSuite Connector automatically makes the payment method available for mapping within 24 hours. To map the payment method immediately, you should refresh the Payment Mappings page.

To refresh the payment methods in the Payment Mappings page:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Mappings > Orders.
- The Payment Methods page opens.
4. Click the **Refresh** icon next to any mapping on the page.
- Wait for some time for the update to complete.
5. Refresh the page in the browser.

The dropdown list should now include the updated list of payment methods in NetSuite.

Posting a Gift Card or a Gift Certificate as a Payment Method for an Order

An order contains a gift card or gift certificate in one of the following cases:

- **When a gift card or gift certificate is used to pay for the entire order** – In this case, NetSuite Connector shows the gift card or gift certificate as the payment method on the order. You can then map the gift card or gift certificate payment method to a payment method in NetSuite, like any other payment method. For more information about mapping payment methods, read [Mapping Order Payment Methods in NetSuite Connector](#).

- **When a gift card or gift certificate is used as partial payment on an order** – In NetSuite, you cannot have more than one payment methods on an order. But in this case, there is an additional payment method on the order. Therefore, when properly configured, NetSuite Connector sends the gift card or gift certificate to NetSuite as a discount item. For more information about setting up a gift card or a gift certificate discount item, read [Gift Cards and Gift Certificates in NetSuite Connector](#).

Troubleshooting NetSuite Connector Invalid Payment Method Errors

When you use an invalid payment method to sync an order, NetSuite Connector displays the error message, ERROR: INVALID_KEY_OR_REF - Invalid paymentmethod <reference_key>. The reference key in the error message is the internal ID of the invalid payment method in NetSuite.

To fix this error, you need to search the NetSuite record for the payment method, then investigate what causes the issue.

The following procedure applies only for the **Payment Method** field. These steps may not be applicable for the **Payment Option** field, which replaces the **Payment Method** field when you enable the Payment Instruments feature in NetSuite.

To troubleshoot the invalid payment method error:

1. Go to Setup > Accounting > Accounting Lists. You can also use the NetSuite global search and enter **Page: Payment Methods** to go to the same page.
2. Find the invalid payment method based on the reference key in the error message. This number should match the NetSuite internal ID.
If the internal ID column is missing on the page, you might need to set the **Show Internal IDs** preference first. To learn more, read [Viewing Internal IDs in NetSuite](#).
3. Edit and save the record without any changes. If NetSuite requires you to provide additional details, enter the required information before you save the record. Then, try to sync the order again.
4. If the sync still fails, try to recreate the order in NetSuite manually. Make sure you have the same information in the following fields:
 - Customer
 - Subsidiary
 - Shipping Address
 - Items
 - Payment Method

When you recreate the order, check if the payment method is missing from the dropdown list. If the payment method is available as an option, NetSuite will display an error if you try to save the order with the invalid payment method.

NetSuite Connector cannot always identify why NetSuite rejects other payment methods. You may need to investigate and make the necessary adjustments in NetSuite. If you can enter the order manually using the payment method in question, NetSuite Connector should be able to send orders to NetSuite using the same payment method.

Troubleshooting Payment or Shipping Methods Not Being Available to Map

You can map only the shipment methods and payment methods for orders that you have imported into NetSuite Connector. If NetSuite Connector does not list a shipment or payment method from your storefront, then no order with that method has got into NetSuite Connector.



Note: For multiple subsidiaries set up in NetSuite, make sure you set up those subsidiaries in the **NetSuite Subsidiaries Used For FarApp Connectors** in NetSuite Connector. For more information, read [Setting Up NetSuite Connector for Syncs](#).

Mapping the Shipment or Payment Methods in NetSuite Connector

The following procedure provides steps to locate and map the shipment or payment methods for mapping in NetSuite Connector.

To map shipment or payment methods in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the connector and relevant account.
3. Go to Mappings > Orders.
4. Perform one of the following actions:
 - For mapping a shipping method, click the **Shipping** tab
 - For mapping payment method, click the **Payment** tab.

Depending on your action, a table displays the shipping methods or payment methods on orders that were imported into NetSuite Connector from the storefront. For any orders that come with a new shipping method or payment method method, NetSuite Connector adds an entry to the left column of the table.

5. For the new entry, in the right column of the table, enter the corresponding NetSuite method.

Check the following articles for information about refreshing shipping and payment methods:

- [Refreshing Shipping Methods for NetSuite Connector](#)
- [Refreshing NetSuite Payment Methods for NetSuite Connector](#)

Handling Invalid Shipping Tax Code Reference Key in NetSuite Connector

NetSuite Connector does not map a shippingtaxcode by default. This means NetSuite will always assign shippingtaxcode automatically. Do not use the taxCode or taxItem mapping, if you have it set. After disabling taxCode or taxItem mapping, try syncing the order again.

If NetSuite is automatically assigning an invalid tax code, the tax setup in NetSuite could be set up wrong. To resolve the error, follow these steps.

To fix automatic assigning of invalid tax code:

1. Go to Setup > Accounting > Set Up Taxes.
The Set Up Taxes page opens.
2. In the **Tax Code Lists Include** field, select **Tax Groups and Tax Codes**.
3. Click **Save**.

You can now manually resync the order or wait for the scheduled sync to automatically retry. If the error is not resolved, contact NetSuite Connector Support.

Fixing Incorrect Shipping Tax on the Order in NetSuite

If the orders that NetSuite Connector sends to NetSuite show incorrect shipping tax amount, the issue is usually that the NetSuite Connector SuiteApp is not installed.

If the issue persists even after installing the NetSuite Connector SuiteApp, contact NetSuite Customer Support.

Configuring Shipping for NetSuite Connector

Set up the shipping configuration and shipping items (referred to as shipping method in NetSuite Connector).

To set up the shipping configuration:

1. Go to Setup > Accounting > Shipping.
2. In the Set Up Shipping page, ensure that the **Charge for Shipping** box is checked.
3. Click **Submit**.

Setting Up Shipping Item Configuration

NetSuite by default provides few shipping items, but you must complete the set up of the shipping items.

To set up the shipping item configuration:

1. Go to Lists > Accounting > Shipping Items.
 2. From the shipping items list, click the shipping item that you want to set up.
 3. Make sure the **Display in Web Site** box is checked.
 4. In the **Shipping Rate** subtab:
 - a. From the **Account (Shipping)**, select an account.
 - b. Choose **Flat Rate** and enter **1.00** in the adjacent field.

NetSuite Connector overwrites this amount with the actual shipping amount.
 5. Click the **Shipping Rules** subtab.
- Depending on your NetSuite setup, the subtab name can also be **Shipping and Handling Rules**.
6. From the **Charge Tax on This Shipping Portion of Item** field, select a taxable tax schedule.
- You must select a taxable tax schedule even if you do not tax shipping, else you will get a mismatched totals error on order import. Selecting a taxable tax schedule enables NetSuite Connector to use the item as both taxable and non-taxable.
7. Click **Save**.

Setting Up Shipping Address as Non-Residential in NetSuite Connector

By default, when NetSuite Connector syncs with NetSuite, the shipping address on an order is marked as residential. Use the following procedure to override this behavior.

To set up the shipping address as non-residential:

1. Log in to [app.farapp.com](#).
2. Select the connector and then select the relevant account.
3. Go to Mappings > Orders.
4. Click the **Order** tab.
5. Click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
6. From the **NetSuite Field** dropdown list, select **Is Residential**.
7. Click **Add Mapping**.
A confirmation message appears indicating the new mapping is added to the end of the mappings list.
8. Click **Close**.
A new row for the **Is Residential** mapping displays at the end of the mappings list.
9. In the newly added row, from the dropdown list in the **<Marketplace or Cart Name> Field / Fixed Value** column, select **Unchecked**.
10. Click **Save**.

Posting an Order After Resolving the Error on the Order

If an error occurs when you post orders to NetSuite, you can hover over the order to view the error. NetSuite Connector continuously tries to repost orders with errors as part of order sync. If you resolve the error, the order successfully posts to NetSuite when the scheduled sync runs again after 20 minutes. Alternatively, to manually sync the order, click the **Action** icon on the order and select **Retry Posting to NetSuite**.

You can also select multiple orders and post them all by selecting **Post Selected** from the **Bulk Actions** dropdown list.

Exporting Order Sync Errors from NetSuite Connector

You can export the errors that NetSuite Connector encountered when syncing orders to NetSuite.

To export order sync errors:

1. Log in to [app.farapp.com](#).
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. Click **Export Errors**.
The Export Order Errors popup window opens. The **Channel** and **Account** fields are populated with the current connector and account.
5. Verify the selection in **Channel** and **Account** fields.
6. Click **Download CSV File**.
Depending on your browser settings, the file either automatically downloads or a prompt for location to download file appears and then you can download the file.
7. Click **Close**.

Troubleshooting Order or Fulfillment Sync Errors in NetSuite Connector

If you get an error with order or fulfillment sync in NetSuite Connector, hover over the order row to see the error details.

If you are not clear how to resolve the error, either search SuiteAnswers or the help center documentation with relevant keywords to find the solution. The following table lists some common error messages that users get and related help topic for finding the resolution. If you require further assistance, contact NetSuite Connector Support.

Error Message	Related Topic
Shipment method hasn't been mapped. Please map this order's shipment method or view/edit all of your shipment methods.	Mapping Order Shipment Methods in NetSuite Connector
Payment method hasn't been mapped. Please map this order's payment method or view/edit all of your payment methods.	Mapping Order Payment Methods in NetSuite Connector
NetSuite is reporting that shipment method with internal ID # isn't valid. Please double-check that this shipment method is valid and that it's allowed for this customer.	Troubleshooting NetSuite Connector Shipping Method Errors
ERROR: INVALID_KEY_OR_REF - Invalid paymentmethod reference key.	Troubleshooting NetSuite Connector Invalid Payment Method Errors

Troubleshooting Common Order Sync Errors in NetSuite Connector

When importing orders, you may encounter error messages from NetSuite. This topic discusses the common errors and their resolution.

Error	Description
ERROR: INVALID_KEY_OR_REF – Invalid item reference key	<p>This error means that NetSuite Connector matched the SKU on the order to a particular item in NetSuite. However, when NetSuite Connector sent that item in the order, NetSuite rejected the item. To resolve the error, check the following in NetSuite:</p> <ul style="list-style-type: none"> ■ Verify that the item with the internal ID shown in the error message belongs to the item you expected to be in the order. Note that all SKU values on items in NetSuite must be unique. If there are duplicate SKU values, NetSuite Connector may match to an unexpected item that is not configured correctly. ■ If you are using subsidiaries, make sure the item whose internal ID is shown in the error message is in the subsidiary your ecommerce orders use. ■ If the item is inactive, NetSuite does not permit posting orders against inactive items. If you want to keep the item inactive, first activate the item in the order to let NetSuite accept the order. Then deactivate the item again.
ERROR: INVALID_KEY_OR_REF – Invalid subsidiary reference key.	<p>This error means that NetSuite Connector is trying to post a subsidiary value that NetSuite is not permitting. This error may be on the customer record or on the order. Follow these steps to narrow down the problem:</p>

Error	Description
	<ol style="list-style-type: none"> 1. Check the order mappings and search for subsidiary. You should see a mapping for the ID referenced in the error message. 2. With the data listed in the order mapping, create a customer record or order manually in NetSuite.
Error: User_Error - Please configure the inventory detail for this line.	<p>Inventory details are a set of fields in NetSuite. These fields must be populated on sales orders or cash sales when an item on an order is lot controlled, bin controlled, or serialized.</p> <p>This error usually indicates that you do not have inventory detail configured for the item in NetSuite. You should be able to import the order after configuring the inventory detail for the item in NetSuite. The leading practice is that NetSuite provides the functionality to automatically populate the inventory details on orders. You can populate the sales order or cash sale form with the inventory details, or use a script to populate the inventory details.</p> <p>You may have the inventory details configured, but the lot or bin that you are using can be out of inventory for the ordered item. In such cases, replenish the stock for the order to post.</p> <p>If none of the preceding solutions work, NetSuite Connector can select the first available lot at a location. This selection will apply to all orders.</p>
ERROR: USER_ERROR - You must enter at least one line item for this transaction.	<p>To resolve this error, check if all items on an order correctly populate in NetSuite Connector and match the storefront order data. Then check the ordered items configuration in NetSuite. For example, for assembly items, ensure that you have a Bill of Materials (BOM) account selected for the line items.</p>
ERROR: USER_ERROR - Inventory items must have a positive amount.	<p>This error appears when one of the items is not priced above \$0.00. If there is an item group in the order, also check the items within the group.</p>
ERROR: TRANS_UNBALNCD — Transaction was not in balance.	<p>This error appears when setting up Avalara to handle taxes in NetSuite. For more information, read the documentation on Avalara website.</p>
ERROR: Invalid record type []	<p>This error appears due to an error in the script in NetSuite. Check the scripts that are running in your account.</p>
ERROR: UNEXPECTED_ERROR - An unexpected error occurred. Error ID:....	<p>This error is mostly caused by one of the following reasons:</p> <ul style="list-style-type: none"> ■ A NetSuite outage or bug. ■ An error in a NetSuite script. For script issues, contact NetSuite Customer Support. <p>If you are not able to resolve the issue, contact NetSuite Customer Support with the error ID.</p>
An order wants you to enter a payment method, but the payment method has already been mapped.	<p>This issue can occur if an order is posting to an existing customer in NetSuite. Check if the existing customer record has a default credit card. If yes, remove the default credit card so that the payment method NetSuite Connector is trying to send to NetSuite can take effect.</p>
Order not found in NetSuite, can't fix item group prices	<p>This error appears when the order was deleted in NetSuite without deleting or canceling the order in NetSuite Connector. To resolve the error, delete the order from NetSuite Connector and import the order again, if required.</p>

Troubleshooting Order Import Issues in NetSuite Connector

This topic covers troubleshooting for the following order import issues:

- [Order Not in NetSuite Connector](#)

- Order Not Imported into NetSuite
- Order Not Imported into NetSuite but Its Status is: Order Posted, Waiting for Shipment

Order Not in NetSuite Connector

If your order is not in NetSuite Connector, check the following:

- **Order sync is enabled** – Check if order sync is enabled. Orders will not import if order sync is disabled.
- **Marketplace or cart is connected to NetSuite Connector** – Test your credentials to ensure your account is connected. For instructions on specific connectors, read the help topic [Integrating NetSuite Connector with Supported Storefronts or 3PLs](#).
- **Order status for NetSuite Connector is set to import** – Check the order status. If the missing order's status is not highlighted in the **Automatically import orders in the following statuses** field, the order will not be imported. For more information, read [Setting Order Statuses for Automatic Import in NetSuite Connector](#).
- **Order date is older than the order cutoff date** – If the order date is before the order cutoff date set in the **Import Orders Created After** field, the order will not import. For more information, read [Setting Order Cutoff Dates in NetSuite Connector](#).
- **Order sync has run** – Order sync runs every 20 minutes. So, NetSuite Connector will take up to 20 minutes from when the order status updated to import the order.



Note: For Amazon FBA, the default order sync interval is 5 hours.

- **Manually import the order** – Check if you can manually import the order from your marketplace or cart. For instructions on manually importing an order, read [Importing Orders from the Storefront in NetSuite Connector](#).

Order Not Imported into NetSuite

If your order did not import into NetSuite, search for the order by ID in NetSuite Connector. If you do not find the order, read [Searching for Orders Posted By NetSuite Connector](#).

Sometimes, the order is in NetSuite Connector, but indicates an error that must be resolved for importing into NetSuite. If the order has an error for several days and you do not resolve the issue, NetSuite Connector may automatically cancel the order. Make sure you address order import errors so that the orders import properly. You should get order import errors on your email address.

If your order status is **Cancelled**, you can revert a canceled order. For more information, read [Reverting Canceled Orders](#).

Order Not Imported into NetSuite but Its Status is: Order Posted, Waiting for Shipment

Search for the order in NetSuite. For more information, read [Searching for Orders Posted By NetSuite Connector](#).

Troubleshooting Refund Import Issues in NetSuite Connector

NetSuite Connector can post refunds from a cart to NetSuite only if an invoice or cash sale is associated with the order. Without a matching invoice or cash sale, NetSuite Connector cannot tell what the refund transaction type should be. If you do not intend to issue invoices or cash sales for your orders in NetSuite, then there are following options to post refunds:

- Cancel the unbilled orders in NetSuite by enabling a setting in your NetSuite Connector account.
- Post the refund only as a return authorization in NetSuite. In this case, NetSuite Connector does not search for a matching invoice or cash sale.

In NetSuite Connector, you can enable these options from the Refund Settings page, by going to Settings > Other Transactions of your connector account.

Troubleshooting End of Group Line Order Errors

When an item group is included in an order, you may get the following error message: ERROR: USER_ERROR

- You cannot edit the end of group line. You must delete the group. You can configure item groups so that NetSuite recognizes the start or end line item of the group. The item group is then considered as a single item for sorting and other purposes. To fix this error, you must first determine which SKUs in the order are set up as item groups in NetSuite. You then remove the start/end line picking ticket references for each item group in the order.

To remove the start or end line picking ticket references on an item group:

1. Go to Lists > Accounting > Items.
2. On the Items page, click **Edit** on the item group.
3. On the **Purchasing/Inventory** subtab, clear the **Reference Start/End Lines on Picking Tickets** box.
4. Click **Save**.
5. Repeat steps 2–4 for other item groups in the order.

You can wait for the order sync to run or manually sync the order. If you get the same error message after syncing, you can delete the order and retrieve it again in NetSuite Connector.

For more information, read [Deleting and Canceling Orders in NetSuite Connector](#).

Troubleshooting Duplicate SKUs Error in NetSuite on Order Sync

You should always have unique SKUs in NetSuite across all item types. For example, with NetSuite Connector, you should not use the same SKU on both an inventory item and kit item.

Fixing Duplicate SKUs Error

If you have inadvertently used the same SKU twice, follow these steps to fix the problem.

To fix the duplicate SKUs error:

1. Determine which of the duplicate SKUs you are going to use in future.
2. For the SKU that you do not want to use,
 - a. Fulfill all the orders containing that SKU.
 - b. Change the **SKU** field to some unique value. For example, if the SKU value was **MYSKU**, change it to **MYSKU-OLD**.
3. Wait until the order sync process runs again. This time, NetSuite Connector should find only one matching SKU and sync the order.



Important: Do not set the SKU that you do not want to inactive. NetSuite Connector will still continue to search that SKU and may show a different error about the SKU being inactive.

You should not delete the SKU that you do not want to use. If there are open fulfillments for the old SKU, you can face issues after deleting the SKU.

If you change the SKU value, you can always temporarily change it to the original value, if needed.

Troubleshooting Missing Field Errors on Orders

When an order form in NetSuite has a required field that is not mapped in NetSuite Connector, you may get the error: NetSuite is rejecting the order/customer because Field Name is missing. You either need to set up a Field Name mapping to add the field to the order/customer or remove the requirement for the Field Name in NetSuite..

To fix the error, you can either map the missing field in NetSuite Connector or change the order form so the field is not required.

To map a missing field in NetSuite Connector

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Orders > Order.
4. On the Order Mappings page, click **Add Mapping**.
5. From the **NetSuite Field** list, select the missing field.
6. Click **Add Mapping** and close the popup.
7. On the **Field/Fixed Value Column**, select the value that you want NetSuite Connector to post on the orders.
8. Click **Save**.

To change an order form so a field is not required

1. Go to Customization > Forms > Transaction Forms.
2. On the Transaction Forms page, locate the order form in the list and click **Customize** or **Edit**.
3. Find the missing field under the **Screen Fields** subtab,
4. Check the box in the corresponding **Mandatory** column.
5. Click **Save**.

Troubleshooting Order Sync Failure Due to Backordered Items

NetSuite will allow an order to contain backordered items most of the time and NetSuite Connector can sync the order without a problem. The exception is if cash sales are being synced (typically done for Amazon FBA and Point of Sale orders) and the item is a lot-numbered item. In this case, an error is displayed informing you that to sync lot-numbered items on cash sales, you must have available inventory numbers in the specified location in NetSuite. Then, the SKUs that do not have inventory numbers are displayed.

The items listed in this error cannot sync because NetSuite is requiring a lot number on the inventory detail for those items. NetSuite Connector does not have lot number information. NetSuite Connector

provides settings to automatically assign the first available lot to each item posted to sales orders and cash and cash sales. When you enable these settings, NetSuite Connector queries NetSuite for the first available lot number from the location being synced on the order item. But if NetSuite does not have inventory for that item in that location, then there is no lot information available. Therefore, NetSuite Connector cannot obtain the lot number from NetSuite to sync the items on the order.

To fix this error, you must add the inventory numbers of the corresponding items in NetSuite in the location that is being used for the sync.

Troubleshooting Negative Total Error

If you get the error **The total cannot be negative**, it means that when NetSuite calculated the total of the order data sent by NetSuite Connector, the total was negative. This topic includes the common causes for this error.

NetSuite Automation

You may have some automations running in NetSuite that automatically calculate and subtract things like commissions for the sales team. If an order comes in with a \$0 total (a completely discounted order), the script may be subtracting further to send the order into negative total.

Make sure you either exclude the records that NetSuite Connector syncs or change the automation to not subtract an amount that would make the order negative.

Discount Items

If the order has a discount, make sure the discount item is set as nontaxable in NetSuite. This is crucial if the discount is posting as a line item. Typically, charging tax will cause the order total in NetSuite to not match your storefront total. In some instances, charging tax may turn the total into negative and cause the total to not post. If the discount is a line item, the order will have negative amount. If you assign tax to that negative amount, you will get a negative tax amount. For example, a line item discount of -\$1.00 with a 10% tax rate would yield -\$0.1. If you have completely discounted the order (\$0 total), then the negative tax amount will subtract from 0 to make the order negative. The same thing happens with low order totals, if the total is low enough such that the negative tax amount would reduce the total past 0.

To fix this issue, change your discount item to not use a tax schedule or set to a nontaxable schedule. You can make this setting in the **Accounting** subtab of the item record.



Note: It is possible that an item is assigned to a tax schedule named **non-taxable** to still have tax charged to the item. Only having the name as non-taxable is not sufficient. The configuration of the tax schedule and the designated area like state, county, city, or province in the tax schedule must be set to not charge tax.

To verify that the discount item is not taxed, manually create the order that is shipping to the same address as the order with the error. Enter the discount line item and check if NetSuite assigns tax to the item. If yes, then either of the following is true:

- The item is not configured to the correct tax schedule
- The tax schedule selected in the item is not set to correctly assign the address in the order as non-taxable.

Troubleshooting Tax Related Errors in NetSuite Connector

This topic discusses troubleshooting of the following tax related errors:

- Taxable Field Not Visible Errors
- Tax Schedule Errors
- Tax Order Errors
- Incorrect Item Amount with VAT Errors

Taxable Field Not Visible Errors

When NetSuite Connector sends an order to NetSuite, it must be associated with a NetSuite customer. This sometimes requires NetSuite Connector to create a new customer in NetSuite. New customers created by NetSuite Connector are always set as taxable. For more information, read the help topic [Tax Setting for Customers](#).

An error will occur when NetSuite Connector attempts to create a new customer and NetSuite is not set to tag new customers as taxable by default. The error message is NetSuite indicates the field taxable is not visible and/or not editable on your Customer form. This usually happens when the new customer is in a subsidiary based in the United States.

To configure NetSuite to default new customers as taxable, read [Setting Customers As Taxable By Default](#).

 **Note:** If the error appears when creating new customers in subsidiaries outside the United States, contact NetSuite Support. Request support to omit the taxable field from the data used in creating new customers.

Tax Schedule Errors

To apply a tax code to an order, NetSuite looks up the corresponding tax code based on the nexus in the order. Depending on the tax code set in the order, NetSuite either looks at the taxable schedule or non-taxable schedule. If one of the schedules is missing, NetSuite uses the other schedule, if available. For more information, read the help topic [Limitations for Using Preferred Tax Codes](#).

Tax Order Errors

For order errors related to tax configuration issues, you may get the following error message in NetSuite Connector:

NetSuite is reporting that the transaction amount would not be in balance so it's rejecting the order. This error normally indicates that NetSuite Connector is trying to push a tax rate on the order but NetSuite is determining that the order should not be taxable and not allowing it to import.

Perform the following actions to troubleshoot these issues:

- Verify if the order has tax. If the order has tax but it is not supposed have it, correct the issue in the storefront. This fix will stop future orders from being taxed incorrectly.
- Create the order manually in NetSuite.
- Check NetSuite settings.

Creating the Order Manually in NetSuite

To check the problem, create the order manually in NetSuite using the same order information that NetSuite Connector is trying to post to NetSuite. This should help duplicate the problem.

To get the order information, click the **Show Order Mapping** option from the **Action** menu (pencil icon) on the order in NetSuite Connector. Make sure you use the same customer, items, and tax rate from the

NetSuite Connector order to create the order manually in NetSuite. NetSuite Connector does not post the total tax amount. NetSuite Connector posts the tax rates and relies on NetSuite to calculate the correct tax amount. If you cannot post the tax rate on the order in NetSuite, NetSuite Connector will also not be able to post the tax rate.

Checking NetSuite Settings

If you are unable to post tax to the order you manually created in NetSuite, then check for the following:

- Make sure the customer on the order is taxable.
- Make sure the items on the order are taxable.
- Check your tax schedules to make sure the state the order is shipping to is configured. If the order is taxable, but taxable nexus is not set up under the taxable tax schedule, NetSuite defaults to the nexus under the non-taxable schedule.
- Check the shipping address is not entered incorrect.
- If you are using subsidiaries, make sure your tax nexuses are added to the subsidiary used on the order form. The tax nexuses can be added in the **Nexuses** subtab of the subsidiary record.

Incorrect Item Amount with VAT Errors

If the item amount with VAT is incorrect in NetSuite, set NetSuite Connector to either of the following:

- Pass the item totals into NetSuite as received from the storefront.
- Subtract the VAT amount from the item when the order is sent to NetSuite so that NetSuite can add the VAT back.

For more information, read [Configuring VAT in NetSuite Connector](#).

Troubleshooting Invalid Account Reference Key Errors

When NetSuite Connector attempts to send or sync an invalid value in the account field for a transaction the INVALID_KEY_OR_REF - Invalid account reference error will show. This means that the value in the account field is not valid in the context it is used in the related transaction. This error can happen when syncing or when updating records or creating related records, such as refunds, where the account value is already present or inherited from the original transaction record. To resolve these errors, you will need to map a valid account to replace the existing or inherited value in order to sync your data.

To map an account to a transaction, see [Mapping an Account for a Transaction](#).



Important: The account that is used in the transactions that are causing this error might not be present. Changing the account in your transaction records should fix the problem. In the case the account is present but you still get an error, you need to check your mappings or that the account used in this transactions is the same as the data NetSuite Connector is syncing. If these are the same and correct you can try manually creating a transaction record with the same details. If the record is successfully saved, take note of the original transaction ID, the error generated, and the transaction ID of the manually-created transaction record and contact support.

Mapping an Account for a Transaction

You will need to map an account for a transaction when syncing transactions with NetSuite Connector. If an invalid account is mapped to a transaction that is synced, you will receive the following error message: INVALID_KEY_OR_REF - Invalid Account Reference Key Error.

To change the account in your transaction record:

1. Open the transaction record.
 - For invoices, go to Transactions > Sales > Create Invoice > List.
 - For credit memos, go to Transactions > Customers > Issue Credit Memos > List.
 - For customer refunds, go to Transactions > Customers > Issue Customer Refund > List.
 - For refunding cash sales, go to Transactions > Customers > Refund Cash Sales > List.
 - For cash sales, go to Transactions > Sales > Enter Cash Sales > List.
 - For customer payment, go to Transactions > Customers > Accept Customer Payments > List.
2. Select a transaction and click **Edit**.
3. Change the account of the transaction record.
 - For invoices:
 - Go to the **Accounting** subtab.
 - Select an account in the **Account** dropdown.
 - For credit memos:
 - Go to the **Accounting** subtab.
 - Select an account in the **Account** dropdown.
 - For customer refunds, select an account in the **A/R Account** dropdown.
 - For refunding cash sales, select an account in the **Account** dropdown.
 - For cash sales:
 - Go to the **Accounting** subtab.
 - Under **Account Information**, select **Account**.
 - Select an account in the **Account** dropdown.
 - For customer payment:
 - Select **Account** under **Primary Information**.
 - Select an account in the **Account** dropdown.
4. Click **Save**.



Important: The account that is used in the transactions that are causing this error might not be present. Changing the account in your transaction records should fix the problem. In the case the account is present but you still get an error, you need to check your mappings or that the account used in this transactions is the same as the data NetSuite Connector is syncing. If these are the same and correct you can try manually creating a transaction record with the same details. If the record is successfully saved, take note of the original transaction ID, the error generated, and the transaction ID of the manually-created transaction record and contact support.

Troubleshooting hidden or not editable ccApproved Field on Order Form Error

This error indicates NetSuite Connector tried to set the **ccApproved** field, but NetSuite is reporting that the write failed. There can be two potential causes for this failure:

- **Payment instruments is enabled** – If you have the Payment Instruments feature enabled in NetSuite, then the **ccApproved** field can be posted only for specifically mapped payment methods

in NetSuite. NetSuite Connector should automatically detect when payment instruments is enabled and not automatically post the **ccApproved** field. However, if you still see this error, then follow these steps:

- Remove order mappings that have a value set for the **ccApproved** field.
- If you do not find any order mappings with the **ccApproved** field value, contact NetSuite Customer Support. The support team will manually switch the backend setting to let NetSuite Connector know that the Payment Instruments feature is enabled.
- **Field not set to be visible on the form** – The **ccApproved** field is not set to be visible on the form that NetSuite Connector uses for your orders. By default, this form is the preferred form for sales orders or cash sales, but you can configure NetSuite Connector to use a different form. To resolve this error, set the **ccApproved** field to be visible.

Setting the ccApproved Field to be Visible

Set the **ccApproved** field to visible from the transaction form that you use for syncing.

To set the ccApproved field to visible:

1. Go to Customization > Forms > Transaction Forms.
2. Locate the form NetSuite Connector uses to post orders and on that form row, click **Edit**.
The transaction form page opens.
3. Click the **Sublists** subtab.
4. In the **Sublists** subtab, click the **Billing** subtab.
5. On the **Payment** row, check the **Show** box.
6. Click the **Screen Fields** subtab.
7. In the **Screen Fields** subtab, click the **Billing:Payment** subtab.
8. On the **Credit Card Approved** row, check the **Show** box.
9. Click **Save**.



Note: The **ccApproved** field is used only for orders that are paid using credit card. Orders paid using other payment methods such as cash will not display this field in the payment information.

Troubleshooting Cleared ccApproved Box on a Sales Order

NetSuite Connector checks the **ccApproved** box on the sales order in NetSuite for every credit card charge that is approved in the marketplace or cart.

You can confirm or view the entire process that NetSuite Connector uses in completing these steps. For more information, read [Viewing Order Mapping in NetSuite Connector](#).

In some cases you may find that despite NetSuite Connector posting a "true" value for the field, the **ccApproved** box is not checked on the sales order in NetSuite.

NetSuite, however, may not check the sales order even if NetSuite Connector posts a **True** value to the **ccApproved** box.

This happens in either one of the following:

- **A payment method other than credit card is mapped for the order in NetSuite Connector.** If NetSuite Connector is sending the payment on the order as cash then the payment method will not have a Credit Card Approved box to check in NetSuite.

For instructions on how to change your order payment methods mappings, read [Mapping Order Payment Methods in NetSuite Connector](#).

- **A payment gateway integrated in your NetSuite account is configured to always leave the ccApproved box unchecked.** This setting can override the data that NetSuite Connector posts with the order. In this case, NetSuite is configured so that checking the **ccApproved** box requires that the other credit card fields are completed with data that NetSuite Connector cannot sync, such as the last four digits of the credit card number, cardholder name, etc. When the data is not provided, NetSuite will clear the **ccApproved** box.

You can manually test if your NetSuite configuration allows the **ccApproved** box to be checked. navigating to an order in NetSuite, checking the **Credit Card Approved** box (usually found under the **Billing** subtab), and saving the order. If the box does not stay checked after saving, then NetSuite Connector will not be able to check the box either.

To test if your NetSuite Configuration allows the ccApproved box to be checked:

1. Open an order in NetSuite.
2. Go to the **Billing** subtab.
3. Check the **Credit Card Approved** box.
4. Click **Save**.
5. Open the order again.

If the **Credit Card Approved** box is still clear, then NetSuite Connector is unable to modify this box for your orders.

Troubleshooting Orders Outside the Current Account Period Error in NetSuite Connector

This error means that NetSuite Connector tried to sync an order or some other transaction into NetSuite for a data that includes a closed accounting period. However, NetSuite does not accept a date in a closed accounting period and you get this error. You can try the following options to fix the issue:

- **Reopen the accounting period** – If you reopen the accounting period, then NetSuite will let the sync to run. For more information, read [Creating or Reopening an Accounting Period in NetSuite](#).
- **Force the transaction in NetSuite with the current date** – Enable the setting NetSuite Connector provides to force the transaction using the current date. For example, even if the order is two months old, in NetSuite, the order's transaction date will be the date on which the order was synced.



Note: This setting does not work for order related records like customer deposits or customer payments. In such cases, either reopen the accounting period or use the current date and then edit the record in NetSuite to adjust the date as needed.

Enabling Automatic Use of Current Date for Closed Accounting Period Errors

Before proceeding with enabling the setting, note the following:

- The setting does not disable or remove mappings for the transaction date. If you have a mapping to override NetSuite Connector's default handling of the transaction date, then you should override that mapping for this setting to work.

- If you keep the setting enabled, NetSuite Connector will automatically sync orders that fail with the closed accounting period error with the current date. You can also temporarily enable the setting to sync some older orders and then disable the setting.
For information about manually running a scheduled sync, read [Manually Triggering a Scheduled Sync in NetSuite Connector](#).
- This setting applies to all connectors. To temporarily enable the sync for a single connector account, disable the syncs on the accounts you do not want the setting to run.

To enable automatic use of current date for closed accounting period errors:

1. Log in to app.farapp.com.
2. Select NetSuite > Settings > General.
3. Click **Advanced Options**.
4. Check the **Post Current Date for Accounting Period Order Errors** box.
5. Click **Save**.

A banner message confirming the update appears.

Troubleshooting Shipping Related Order Errors in NetSuite Connector

This topic covers troubleshooting of the following shipping related order errors:

- [Shipping Not Getting Calculated](#)
- [Missing Shipping Amounts for Orders](#)
- [Unapplied Shipping Discounts for Orders Posted Using NetSuite Connector](#)

Shipping Not Getting Calculated

Following is an example error:

ERROR: USER_ERROR - NetSuite Connector Error - Web order total is 1214.11 but NetSuite computed the total as 1104.15

To resolve this error, enable the **Charge for Shipping** feature from the Set Up Shipping page, by going to Setup > Accounting > Shipping. After enabling the feature, NetSuite Connector should populate the shipping and sync to NetSuite. Enabling the **Charge for Shipping** changes shipping items such that they require additional data and the items remain invalid until you provide that data. NetSuite provides error about the missing data.

Missing Shipping Amounts for Orders

By default, NetSuite Connector sends the shipping amount to NetSuite. However, if a shipping method is not provided to NetSuite or if the method is not properly configured, then NetSuite discards the shipping cost. To resolve this issue, make sure you map the shipping methods from the storefront to NetSuite. If the default shipping method or the method on the order is set to **Not Mapped**, then no shipment method is sent to NetSuite. Therefore, NetSuite ignores the sent shipping cost.

For instructions on verifying shipping method configuration, read [Configuring Shipping for NetSuite Connector](#).

For instructions on mapping shipping methods, read [Mapping Order Shipment Methods in NetSuite Connector](#).

To make the native **shippingCost** field available on the form, NetSuite must be configured to enable **Charge for Shipping** setting. For more information, read [Configuring Shipping for NetSuite Connector](#).

Unapplied Shipping Discounts for Orders Posted Using NetSuite Connector

For a discount applying to the entire order including the shipping cost in the storefront, there can be an issue when the order posts to NetSuite. The issue is that the discount does not apply to the shipping cost when the order posts to NetSuite. The issue occurs if the SuitePromotions feature is enabled in NetSuite. When this feature is enabled, promotions applied at the transaction level apply only to the transaction subtotal. For example, discounts are not applied to tax and shipping costs.

There are following two ways to solve the issue:

- Create two separate promotions in the storefront. One promotion for the order amount and other promotion for the shipping amount. If you have a promotion specifically for the shipping amount, SuitePromotions can apply the promotion to the shipping cost. These promo codes must have matching promo codes in NetSuite. For more information, read [Setting Up Matching Promo Codes from the Marketplace or Cart to NetSuite](#).

Also, make sure the **SuitePromotions is Enabled in Your NetSuite** box is checked in NetSuite Connector. For more information, read [Setting Up NetSuite Connector for Syncs](#).

- Disable SuitePromotions in NetSuite, which will allow NetSuite to apply discounts to tax and shipping amounts. For more information about enabling and disabling SuitePromotions, read the help topic [Configuring Promotions](#).

Troubleshooting TranID Field Visibility Error in NetSuite Connector

This error shows when the TranID is not editable in NetSuite. NetSuite Connector is trying to write data to the TranID field but is not allowed since the Allow Override in the Auto-Generated Numbers Setup is not checked.

To setup your Auto-Generated Numbers and allow override for your specific TranID, see the help topic [Setting Up Auto-Generated Numbering](#).

For more information on auto-generated numbers and the allow override option, see the help topic [Auto-Generated Document Numbers](#).

NetSuite Connector Order Sync FAQ

The following are some frequently asked questions for NetSuite Connector order sync.

[I resolved my order errors. What do I need to do to get the order into NetSuite?](#)

No action is needed to sync the orders. The orders automatically sync to NetSuite on the next automatic sync run.

[I want to change the order number format my cart outputs. Will that cause problems with NetSuite Connector?](#)

Changing order number format should not cause issues with NetSuite Connector provided you adhere to the following:

- The order numbers for orders that are already retrieved into NetSuite Connector should not change. Changing order numbers on orders that NetSuite Connector has already retrieved can cause issues with subsequent transactions that utilize the order number, like fulfillments or refunds.
- Old order numbers should not be reused on new orders.
- If you have a special mapping for your order numbers, make sure the new order number format does not break the mapping.

If I choose the Post Coupon Code to Matching Coupon Code in NetSuite setting and there is no match, will it fall back to the default discount item?

Configure your default discount item as described in the topic [Managing Order Discounts in NetSuite Connector](#). Then, NetSuite Connector will automatically fall back to using the default discount item when no matching promo code is found.

If I import unfulfilled orders from the storefront to NetSuite, will NetSuite Connector sync the fulfillments for the imported orders?

NetSuite Connector only looks for fulfillments for orders it syncs into NetSuite. If you import orders using CSV files, NetSuite Connector is not aware of those orders and will not attempt to sync them to the storefront.

I accidentally deleted an order from NetSuite Connector. How can I import the order back?

Deleting an order from NetSuite Connector removes the order from NetSuite Connector. In most storefronts, if the order was created or updated in the last two days, NetSuite Connector pulls the order back in the next order sync. In the following cases, you should manually pull the order into NetSuite Connector by clicking the **Retrieve** button on the Orders page:

- The order was from a storefront, such as Amazon, where NetSuite Connector only pulls orders after the last pick
- The order is not from a recent period
- You do not want to wait until NetSuite Connector automatically pulls the order

In all other cases, the orders should be automatically retrieved again by the sync.

For information about retrieving an order, read [Importing Orders from the Storefront in NetSuite Connector](#).

I deleted an order from NetSuite that was imported through NetSuite Connector. Will NetSuite Connector automatically import the order again?

No, because the order is already in NetSuite, NetSuite Connector marks the order as posted and does not resync the order. If you want to resync the order, delete the order from NetSuite Connector and retrieve the order manually. For information about retrieving an order, read [Importing Orders from the Storefront in NetSuite Connector](#).

Can I repost an order from NetSuite Connector that has been deleted from NetSuite?

When NetSuite Connector posts an order to NetSuite, it stores the internal ID that NetSuite generates and assigns the ID to the order. NetSuite Connector also marks the order as posted and does not attempt to sync the order to NetSuite. Instead, NetSuite Connector starts looking for fulfillments associated with that order. Therefore, it is not possible to repost an order that was deleted from NetSuite. To resync the order, delete the order from NetSuite Connector and retrieve the order manually. For information about retrieving an order, read [Importing Orders from the Storefront in NetSuite Connector](#).

Is there a way to export or save my mappings?

NetSuite Connector currently does not have a feature to export mappings.

Can NetSuite Connector create a mapping to add an SKU of my choice to an order?

No, NetSuite Connector can only use items in the order data received from the storefront. It does not support creating mappings to add more SKUs.

Why does my Amazon order not have customer information in it?

Your order may not have customer information because of the following reasons:

- Amazon does not release customer information about Fulfilment By Amazon (FBA) orders when transmitting the orders through API. You must set up a fixed customer in NetSuite Connector for those orders.
- You have the option to sync pending orders. Orders with a pending status do not provide customer information until those orders are moved out of pending status. Those orders are also assigned a fixed customer until NetSuite Connector detects the orders are no longer pending. When the orders are out of pending status, NetSuite Connector updates the orders with the correct details and changes the customer to the correct customer.

Why don't my orders import to NetSuite after enabling TaxJar in NetSuite?

The orders does not import because the script execution order is incorrect. To resolve this issue, change the script execution order on the form (either sales order or cash sale) to which NetSuite Connector is posting the orders. The **FA | UE Order Total Validation** script should run after the **TaxJar** script.

For more information, read [Changing the Script Execution Order in NetSuite Connector When TaxJar is Enabled](#)

Should I use quantityAvailable field to sync quantity updates?

Yes, you should use quantityAvailable field. Quantity available is the total number of inventory for a SKU that is currently available for use in filling new orders. The other common metric of quantity in NetSuite is quantity on hand. Quantity on hand is the total number of inventory that physically located in a warehouse at the current time. However, some of the quantity on hand may already be allocated for fulfilling current sales orders. Therefore, using quantity on hand risks overselling.

Can different connectors sync quantity from different warehouses?

Yes, you can customize quantity mappings for individual connectors. This customization is useful if you are selling in multiple countries and want to avoid shipping internationally for items that you have in stock locally.

Can I have a negative quantity in a storefront?

Most storefronts do not support negative quantity. But when using NetSuite Connector you should post 0 quantity or mark an item out of stock, as appropriate for the storefront.

Some of my orders show as archived in NetSuite Connector. What does that mean? When does that happen and can I still access the orders?

NetSuite Connector archives orders that are 120 days old. Archiving compresses the data that composes the order and removes the order from the orders list in the order dashboard. Archiving is done to conserve space and focus on more recent and likely more relevant orders.

Archived orders are still in NetSuite Connector but do not show in the NetSuite Connector's Order dashboard unless you search for the specific order number.

Removing an Order from Archive in NetSuite Connector

To remove an order from archive, follow this procedure.

To remove an order from archive:

1. Log in to app.farapp.com.
2. Select the desired connector and relevant account.
3. Go to Data Flows > Orders.
4. In the **Search** field, enter the order number that you want to remove from archive and press **Enter**.
The order displays in the list.
5. Click the pencil icon in the **Action** column and select **Unarchive**.

Sometimes, NetSuite Connector automatically removes an order from archive for some processes. Therefore, you may see some orders older than 120 days removed from archive in your account. For example, if you are using refund sync and an order older than 120 days was refunded, NetSuite Connector removes the order from archive to sync the refund.

[Why are my shipping and payment method values blank when orders import into NetSuite Connector?](#)

This issue occurs if the order does not have a shipping method or payment method in the storefront. Verify by checking the order in the storefront. To show these values on NetSuite sales orders, review and update your storefront configuration to include the shipping and payment methods when placing orders. Also, make sure you set up the payment and shipping method mappings. NetSuite Connector then imports this information with the order and posts the information to NetSuite.

For more information about setting up payment and shipping method mappings, read the following topics:

- [Mapping Order Payment Methods in NetSuite Connector](#)
- [Mapping Order Shipment Methods in NetSuite Connector](#)

[Why are old orders appearing in NetSuite Connector?](#)

Old orders may appear in NetSuite Connector if you have the setting to sync order updates originating in your marketplace or cart enabled.

To resolve this issue, try changing the **Import Orders Created After** setting in your NetSuite Connector account to a more recent date. This date will limit how far back NetSuite Connector looks for orders. For more information, read [Setting Order Cutoff Dates in NetSuite Connector](#).

If the orders have previously posted to NetSuite and you want to sync an order update, delete the orders from NetSuite Connector. For more information read [Deleting and Canceling Orders in NetSuite Connector](#).

[Why are all marketplace or cart fields not available when creating an order mapping?](#)

NetSuite Connector can only map fields if they are available in the marketplace or cart's API. For more information about viewing and mapping all available fields, read [Finding an Available Order Data to Map in NetSuite Connector](#). If a field is not visible by following the method in the topic, then that field is not available in the API and cannot be mapped.

Why does the shipment method on the fulfillment in the storefront not match the shipment method in the order?

When syncing fulfillments from NetSuite, NetSuite Connector syncs the shipping method used on the item fulfillment and not the shipping method on the sales order. The item fulfillment represents the actual method used to ship the order, which may differ from the original shipment method on the sales order.

Why does the tax rate on the order in NetSuite not match the tax rate on the order in storefront?

By default, NetSuite Connector sets tax rates on orders in NetSuite such that the tax total in NetSuite matches the tax total in the storefront. There can be rounding method differences when calculating tax between some storefronts and NetSuite. Due to this rounding method difference, you may see slight differences in the tax rate shown on the order in NetSuite. The important number is the tax amount and that should always match between NetSuite and the storefront. If the tax rate and tax total do not match to your storefront, then check the following:

- The **Match Line-Item Tax Rates Match with Tax Rates Created from Transaction (Beta)** box is checked in NetSuite. For more information, read [Handling Line Level Tax Rates in NetSuite Connector](#).
- If you have a NetSuite tax plug-in or solution, like Avalara, you may overwrite the tax rate NetSuite Connector sent with the order. Different tax solutions may or may not let NetSuite Connector set the rates on the orders NetSuite Connector sends. Check your tax solution documentation for answers. If the solution needs a field to be populated on the order, then NetSuite Connector can map the required value to that field. Avalara uses this method. If you are unable to accomplish the mapping through the user interface, contact NetSuite Customer Support.

Why are tax rates on cash sales not matching the sales order tax rates?

When a cash sale is created from sales order in NetSuite, NetSuite automatically recalculates the tax on the cash sale. To correct this, follow these steps.

To correct tax rates on cash sales to match the sales order tax rates:

1. Go to Connector > Configuration > Setup.
2. On the NetSuite Connector Setup page, check **Match Line-Item Tax Rates With Tax Rates Created From Transaction (Beta)**.
3. Click **Submit**.

This setting will ensure that the line-item tax rates on the Cash Sale will match the tax rates on the Sales Order it was created from.

Why do I not receive an email notification when NetSuite Connector syncs have errors?

To get notified for errors, add your email address to the notification settings for the respective error type on each connector. For more information, read [Notification and Email Settings for NetSuite Connector](#).

Why is NetSuite Connector not setting tax on my order in NetSuite when the order was charged tax in the storefront?

If the orders have tax charged in the storefront but that tax is not present on the order in NetSuite, then check the following:

- Verify that the storefront actually charged tax on the order by checking the order directly in the storefront.
- Check the order and order item mappings to see if you have mappings that are setting the **isTaxable** field to **False**.

- Make sure the state of the **Line-Item Taxes are Enabled in Your NetSuite** box in NetSuite Connector accurately reflects the status of line item tax settings in NetSuite.
- If the storefront is remitting taxes for orders to the taxing authority on your behalf, by default NetSuite Connector omits taxes from such orders. To change this behavior, read [Omitting Remitted Tax in NetSuite Connector](#).

Why is NetSuite Connector setting a price level of -1 on my order mapping?

NetSuite Connector sets a price level of -1 on the items in your order because -1 represents the custom price level in NetSuite. When a custom price level is selected, any price can be entered for the item.

NetSuite Connector sets this price level so that the items in your synced order in NetSuite always has the same price as the corresponding item in the storefront order. This price is regardless of what price might be set in various price levels on the item record in NetSuite.

Will NetSuite Connector automatically repost an order that is deleted from NetSuite?

No. Because the order is already in NetSuite, NetSuite Connector marks the order as **Posted** and does not resync the order. To resync such an order, delete the order from NetSuite Connector and retrieve the order manually. For instructions on manually retrieving an order, read [Importing Orders from the Storefront in NetSuite Connector](#).

Can NetSuite Connector post to fields on other records created from original sales orders in NetSuite?

The transformation of sales orders into other records such as cash sale and invoices is done in NetSuite. NetSuite Connector cannot post any additional data to those records or set any mappings to post data to those records. However, the NetSuite Connector SuiteApp provides a functionality for cash sale records that NetSuite creates from the sales orders sent by NetSuite Connector.

Using a script included in the SuiteApp, NetSuite Connector validates that the cash sale order total matches the total on the sales order. This validation ensures that the correct amount is billed.

Can NetSuite Connector look up the lowest cost shipping method from NetSuite and post it on the order during order sync?

No, any shipping method mapping you can create is based on incoming order data. Which shipping method you use to ship the order is determined after NetSuite Connector has already synced the order into NetSuite.

NetSuite Connector Refund Sync Management

This section covers the following topics on managing refund syncs in NetSuite Connector.

- [Refunds and Exchanges in NetSuite Connector](#)
- [Syncing Refund for Orders Deleted from NetSuite](#)
- [Mapping the Restock Field on Item Receipts Created on Refunds](#)
- [Stopping Cash Refunds from Restocking Refunded Items](#)
- [Deleting or Canceling Refunds in NetSuite Connector](#)
- [NetSuite Connector Refund Sync FAQ](#)

Refunds and Exchanges in NetSuite Connector

NetSuite Connector can sync a refund from the storefront to NetSuite, or from NetSuite to the storefront.



Note: You cannot sync refunds in both directions for a single account under one connector. For example, if you have one Shopify account, you must choose which way you want to sync your refunds - either to NetSuite or from NetSuite, but you cannot sync both ways. If you have multiple Shopify accounts though, you can choose the direction of the sync for each account. Also, you can set refunds to sync either way for different connectors; for example, to NetSuite for Shopify, or from NetSuite for WooCommerce.

Consider the following:

- If the storefront is a marketplace, such as Amazon or eBay, the best practice is to perform the refund in the storefront and have NetSuite Connector sync the refund to NetSuite. The reason for this is that if a customer wants a refund, they can request or perform the refund themselves in the marketplace site.



Note: For eBay, the refund must be requested from the buyer's eBay account before you issue the refund from your eBay seller account. If the user issues a refund on the order without the buyer first requesting it from their account, it will appear as an external transaction in the API and NetSuite Connector will not be able to retrieve the refund.

- If the storefront is a cart, such as Shopify or Magento, the best practice is to perform the refund in NetSuite and have NetSuite Connector sync the refund to the storefront. The reason for this is that if a customer wants a refund, they have to message the team that runs the site to request the refund directly.
- NetSuite Connector automatically detects whether to sync a credit memo or a cash refund and syncs against the appropriate transaction in NetSuite. Therefore, the following settings in the Refund Settings page should not be checked:
 - When a Refund is detected in the Storefront, Only Sync a Cash Refund Against a Matching Record in NetSuite. Applies to All Channels/Accounts.
 - When a Refund is detected in the Storefront, Only Sync a Return Authorization Against a Matching Record in NetSuite. Applies to All Channels/Accounts.

If you want to setup refund sync, contact the NetSuite Connector Professional Services team.

Exchanges on Orders in NetSuite Connector

An exchange is essentially a return and a new sale of the returned item. NetSuite Connector supports both transaction types independently, same as NetSuite.

NetSuite Connector provides refund sync. If an exchange generates a refund and the customer is using refund sync, then that refund syncs as expected. However, the return records do not sync.

Syncing Refund for Orders Deleted from NetSuite

If an order is deleted from NetSuite and refunded in the marketplace or cart, NetSuite Connector is not able to post a refund. The reason is that the order needs to exist for syncing a refund but the order is already deleted from NetSuite.

Cancelling a Refund that is Trying to Sync

The following procedure can be used to cancel the posting of a specific refund to NetSuite or the storefront. This procedure is needed when a refund is generating an error on syncing and you are unable to resolve the error. After resolving the error, make sure you take the necessary steps in the storefront and NetSuite to account for the refund.

To cancel a refund that is trying to sync:

1. Log in to app.farapp.com.
2. Select the required connector and then select the relevant account.
3. Go to Data Flows > Other Transactions.
4. Perform one of the following actions:
 - If syncing refund from the marketplace or cart, then from the **Source Channel** dropdown list, select the marketplace or cart.
 - If syncing refund to the marketplace or cart, then from the **Source Channel** dropdown list, select **NetSuite**.
5. From the **Transaction Type** dropdown list, select the transaction type.
6. Locate the transaction from the list that you want to refund, hover over the **Action** icon for that transaction and select **Cancel**.

Stopping Cash Refunds from Restocking Refunded Items

NetSuite Connector refund sync supports refund syncing either to or from NetSuite. The sync works only in one direction at a time per connector and the supported directions can vary by connector. When syncing refunds from NetSuite, you can decide whether an item will be restocked before NetSuite Connector refund sync is involved. However, when syncing refunds to NetSuite, you can make NetSuite automatically restock the refunded items that display in the cash refund records.

If you do not want NetSuite to automatically restock items when they appear on a cash refund, use one of the following options:

- [Creating a Return Location](#)
- [Setting the Item Quantity to 0](#)

Creating a Return Location

You can create a return location in NetSuite for returned or refunded items and set that location on the line items of your cash refunds. You can create a mapping in NetSuite Connector to set that location on the items in the cash refund. This does not affect your salable inventory.

The following are the advantages and disadvantages of using this option:

- Advantages:
 - Easy to setup and configure.
 - Lets you view the item record and see the quantity of item returned by looking at the quantity available in your return location.
 - Lets you dynamically decide whether to set the location using the data provided by the marketplace or cart.
- Disadvantages:
 - Requires Multi-Location Inventory feature enabled in NetSuite.
 - Reports or other processes that tally inventory across locations must be adjusted to exclude the return location.

Before creating the return location in NetSuite Connector, identify or create a location that you want to use for returns in NetSuite.

To map a location for cash refund records:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Other Transactions.
4. Click the **Transaction Item** tab.
5. Click **Add Mapping**.
6. From the **Transaction Type** list, select **Cash Refund**.
7. From the **NetSuite Field** list, select **Location**.
8. Click **Add Mapping**.
The mapping is added to the Other Transaction Item Mappings page.
9. Click **Close**.
10. From the newly added row, click the dropdown list in the **NetSuite Field/Fixed Value** column and select the location that you want to use for returns.
If you created a new location in NetSuite and do not see it in the **NetSuite Field/Fixed Value** dropdown list, click **Reload NetSuite Lists**. After the reload process completes, check the list again.
11. Click **Save**.

Setting the Item Quantity to 0

Following are the advantages and disadvantages of using this option:

- Advantage – Easy to setup and configure.
- Disadvantage – You will not be able to tell how much quantity of the item was returned. This limitation may have reporting impact.

To set the item quantity to 0:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Other Transactions.
4. Click the **Transaction Item** tab.
5. Click **Add Mapping**.
6. From the **Transaction Type** list, select **Cash Refund**.
7. From the **NetSuite Field** list, select **Quantity**.
8. Click **Add Mapping**.
The mapping is added to the Other Transaction Item Mappings page.
9. Click **Close**.
10. From the newly added row, in the **NetSuite Field/Fixed Value** column enter **0**.
11. Click **Save**.

Deleting or Canceling Refunds in NetSuite Connector

The delete refund and cancel refund operations serve two different purposes in NetSuite Connector.

Deleting a Refund from NetSuite Connector

When you delete a refund in NetSuite Connector, you remove all the refund data only from NetSuite Connector. The refund is not deleted from the storefront or NetSuite. NetSuite Connector can sync the refund again if the refund has a status in the storefront or NetSuite, depending on the direction of refund.

To delete a refund from NetSuite Connector:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. On the refund row that you want to delete, hover over the **Action** menu (pencil icon) and click **Delete from FarApp**.
A popup window appears to confirm the action of removing the refund from NetSuite Connector.
5. Click **OK**.

Cancelling a Refund in NetSuite Connector

When you cancel a refund, NetSuite Connector ignores the refund and stops syncing refund information from the storefront or NetSuite, depending on the direction of sync. NetSuite Connector keeps the canceled refund's data but considers the refund as closed and does not take any action on the refund. Cancelling refunds is commonly used in the following scenarios:

- When you make changes to the refund outside NetSuite Connector.
- When you process a refund manually and do not want NetSuite Connector to sync the refund data to the storefront or NetSuite.

To cancel a refund in NetSuite Connector:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. On the refund row that you want to cancel, hover over the **Action** menu (pencil icon) and click **Cancel**.

The refund is canceled and the status of the order shows as **Canceled**.

Reverting a Canceled Refund

If you canceled a refund in error, you can always revert the canceled refund.

To revert a canceled refund:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. On the refund row that you want to revert, hover over the **Action** menu (pencil icon) and click **Uncancel**.

Deleting and Canceling Refunds in Bulk

You can delete or cancel multiple refunds in bulk.

To delete or cancel orders in bulk:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Data Flows > Other Transactions.
4. From the refunds list, check the boxes next to the refunds that you want to delete or cancel.
5. Perform one of the following actions:
 - To delete the selected refunds, from the **Bulk Actions** dropdown list, select **Delete Selected**.
 - To cancel the selected refunds, from the **Bulk Actions** dropdown list, select **Cancel Selected**.

NetSuite Connector Refund Sync FAQ

The following are some frequently asked questions for NetSuite Connector refund sync.

What effect will refund sync actually achieve in NetSuite?

The effect is same as clicking the **Refund** button in NetSuite. If an order has a cash sale but no refund, and you click **Refund**, NetSuite creates a cash refund. Similarly, NetSuite Connector creates a cash refund with refund sync. If an order has an invoice, but no refund, and you click **Refund**, NetSuite creates a credit memo. Similarly, NetSuite Connector will create a credit memo with refund sync.

How do RMAs (Return Merchandise Authorization) fit in refund sync?

If you are doing the RMA (Return Merchandise Authorization) and receiving in NetSuite, you should issue the refund in NetSuite. NetSuite Connector then syncs the refund to the storefront.

If we do a refund in NetSuite, will the refund sync refund the customer's payment method?

Yes, the refund sync to your cart will issue the refund to the customer. Note that NetSuite Connector does not actually trigger the payment refund. The storefront does the payment refund when it gets the refund data for the order.

If a customer pays with a gift card, will the refund be returned to the gift card?

NetSuite Connector cannot issue a refund and then create an order in the storefront for an equal amount purchasing a gift card. However, NetSuite Connector can refund to the store credit on some connectors. Check if the setting for store credit is provided for your connector on the Refund Settings page by going to Settings > Other Transactions. You will have to indicate on the refund in NetSuite that it should be a store credit refund. Then NetSuite Connector can be configured to override the refund payment method.

Why do I have two refund syncs on my NetSuite Connector dashboard?

You see two refund syncs on NetSuite Connector dashboard because NetSuite Connector provides the following two types of refund syncs:

- Incoming sync, bringing refund information to NetSuite from a storefront.
- Outgoing sync, taking refund from NetSuite and syncing the refund to a storefront.

You should sync to NetSuite from marketplaces and from NetSuite for carts. NetSuite Connector does not support multidirectional sync for a single connector account.

NetSuite Connector Fulfillment Sync Management

NetSuite Connector pulls fulfillment data from NetSuite when it detects the sales order is fulfilled or partially fulfilled, depending on the configuration. NetSuite sets these statuses on the sales orders when one or more item fulfillment records with a **Shipped** status are created for the item.

The fulfillment sync only looks for the status and is not aware of how the sales order achieves the status. If you have a 3PL connector with NetSuite Connector, then the 3PL provider may be responsible for fulfilling the sales order. However, the marketplace or cart connector runs independent from the 3PL connector.

This section covers the following topics on managing fulfillment syncs in NetSuite Connector.

- [NetSuite Connector Fulfillment Sync Triggers](#)
- [Creating Mappings For Fulfillment Sync in NetSuite Connector](#)
- [Enabling Multiple Item Fulfillments from a Single Order in NetSuite Connector](#)
- [Pushing Fulfillments Manually from NetSuite in NetSuite Connector](#)
- [Configuring Custom Fields for Fulfillments](#)
- [Troubleshooting Common Fulfillment Sync Errors in NetSuite Connector](#)

NetSuite Connector Fulfillment Sync Triggers

NetSuite Connector pulls fulfillment data from NetSuite when it detects the sales order is fulfilled or partially fulfilled, depending on the configuration. NetSuite sets these statuses on the sales orders when one or more item fulfillment records with a **Shipped** status are created for the item.

The fulfillment sync only looks for the status and is not aware of how the sales order achieves the status. If you have a 3PL connector with NetSuite Connector, then the 3PL may be responsible for fulfilling the sales order. However, the marketplace or cart connector runs independent from the 3PL connector.

Creating Mappings For Fulfillment Sync in NetSuite Connector

NetSuite Connector syncs the shipping details to the marketplace/cart after you fulfill an order in NetSuite (either partial or complete) and create an item fulfillment record. By default, NetSuite Connector syncs the standard information required in the marketplace/cart on the fulfillment. You do not have to set up fulfillment mappings for these fields. These include:

- Carrier
- Ship Date
- Ship Method
- Tracking Number



Important: NetSuite Connector can only map to standard fields in the storefront's Application Programming Interface (API). NetSuite Connector does not support or allow mapping fulfillment information to custom fields created in the storefront.

Mapping additional fields on the fulfillment is optional. Complete the following steps if you need additional fulfillment fields to be mapped:



Note: The following example creates a logic mapping that sets the Tracking URL field to the FedEx homepage whenever a fulfillment uses a shipping method that contains FedEx in the name. Note that this mapping is for demonstration purposes only and should not be used as Shopify will automatically construct the proper tracking URL for FedEx shipments using the tracking number.

To create new mappings for fulfillment sync:

1. In NetSuite Connector, select the connector and relevant account.
2. Go to Mappings > Fulfillments.
3. Click **Add Mapping**.
4. In the **Add Mapping window**, click **Add Shopify Mapping**.
This example uses Shopify connector, so Add Shopify Mapping is used.
5. A message window will appear that confirms that you have added a new mapping. Click **Close**.
Your new mapping will default to a fixed type mapping.
6. Click the dropdown list and select **Logic** mapping.
7. Click **Logic Mapping - Click to View/Edit**.
8. On the Edit Logic Mapping window:
 - Leave the Value Type dropdown as-is.
 - On the first **Check If** field, select **Fulfillment Header Value**.
 - On the second **Check If** field, select **Ship Method**.
 - On the third **Check If** field, select **Contains**.
 - On the fourth field dropdown leave it at **Fixed**, and enter **FedEx** in the final field in the **Check If** line.
 Completing this step tells NetSuite Connector to check in the Ship Method on the fulfillment contain the word FedEx. The following steps will now tell NetSuite Connector what to do when the fulfillment order contains or does not contain the indicated value.
9. In the **If True, The Set Value To** line, leave the first field dropdown set to **Fixed** and enter www.fedex.com into the second field.
10. Leave **Otherwise, Set Value To** line **Otherwise, Set Value To** line as-is.
Leaving this as-is will not do anything if the shipping method does not contain the word FedEx.
11. Click **Save**.

Enabling Multiple Item Fulfillments from a Single Order in NetSuite Connector

The Pick, Pack, and Ship feature in NetSuite is used to generate multiple item fulfillments for a single order. To enable this functionality in NetSuite Connector, you must enable a setting in NetSuite Connector using the following procedure.

To enable the pick, pack, and ship feature in NetSuite Connector:

1. Log in to app.farapp.com.
2. Go to NetSuite > Settings > General.

3. On the NetSuite Settings page, check the **Pick/Pack/Ship Setting is Enabled in Your NetSuite** box.
4. Click **Save**.

Pushing Fulfillments Manually from NetSuite in NetSuite Connector

NetSuite Connector lets you manually push fulfillments from NetSuite to the storefront if you have an order on which you want to get a fulfillment update.

To push fulfillments manually from NetSuite in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. On the order that you want to fulfill, hover over the **Action** menu (pencil icon), and click **Upload Shipment from NetSuite**.

Configuring Custom Fields for Fulfillments

By default, NetSuite Connector uses the built-in tracking number and carrier on the fulfillment record when posting data back to the storefront. Use this feature to define alternative custom fields that NetSuite Connector will use when posting carrier information and tracking number on fulfillments.

If you use custom fields for the tracking number and carrier on the fulfillment record, you need to define those custom fields in NetSuite Connector. When not properly defined in NetSuite Connector, fulfillments can fail to sync or get merged. This results in fulfillments not matching one to one between NetSuite and the storefront.

To configure custom fields for fulfillment:

1. In NetSuite Connector, select the connector and relevant account.
2. Go to Settings > Fulfillments.
3. Expand the Advanced Options section.
4. If you use a custom field for the tracking number, enter it in **Custom Tracking Number Field**.
5. If you use a custom field for the carrier, enter it in **Custom Carrier Field**.
6. To set the location of the custom field, click the Field Location buttons.
Click **Order** for order form or click **Fulfillment** for fulfillment form. The selected option is highlighted in gray.
7. Click **Save**.

Troubleshooting Common Fulfillment Sync Errors in NetSuite Connector

Following are some common fulfillment sync errors:

Error	Description
Fulfillment error: SKU XXX not found in original order so can't fulfill.	<p>When NetSuite Connector syncs a fulfillment, it verifies that the SKUs on the order match with SKUs on the fulfillment. If you edit the order in NetSuite manually, NetSuite Connector will not be able to sync the fulfillment back to the original order. In this scenario, you can perform either of the following actions:</p> <ul style="list-style-type: none"> ■ Edit the order in NetSuite to place the original SKUs back on the order. ■ Cancel or delete the order in NetSuite Connector and handle the fulfillment sync manually. <p>To delete or cancel an order in NetSuite Connector, read Deleting and Canceling Orders in NetSuite Connector.</p>
Fulfillment error: {u'base': [u"This order has been canceled and can't be fulfilled"]}	<p>If the order is canceled, NetSuite Connector cannot sync a fulfillment. In this scenario, you should also cancel the order in NetSuite Connector. To cancel an order in NetSuite Connector, read Deleting and Canceling Orders in NetSuite Connector.</p>

NetSuite Connector Product Sync Management

This section covers the following topics about product syncs in NetSuite Connector:

- [Setting Up Full Product Sync Mapping](#)
- [Setting the Real-Time Price and Quantity Sync Preference](#)
- [Disabling Real-Time Sync](#)
- [Creating a Flag Field for Product Sync](#)
- [Flagging Existing NetSuite Items to Sync](#)
- [Creating a List-Based Storefront Flag Field Manually](#)
- [Creating a Single-Character Storefront Flag Field Manually](#)
- [Mapping Product or Inventory Sync in NetSuite Connector](#)
- [Viewing Product Mapping in NetSuite Connector](#)
- [Mapping Required and Optional Fields for Product Sync](#)
- [Mapping Cart Categories in NetSuite Connector](#)
- [Mapping Inventories in NetSuite Connector](#)
- [Mapping Custom Field Inventories in NetSuite Connector](#)
- [Blank Value Field for Category Product Mapping in NetSuite Connector](#)
- [Removing or Resetting Data for a Field in the Storefront Using Product Sync](#)
- [Updating the SKU of Items Posted with NetSuite Connector](#)
- [Fixing SKUs that Do Not Match the Storefront SKUs](#)
- [Setting Maximum Quantity for Inventory During Product Sync](#)
- [Setting Minimum Quantity for Inventory During Product Sync](#)
- [Adding Product Images for NetSuite Connector](#)
- [Configuring Assembly Item Quantities in NetSuite Connector](#)
- [Batch Updating Items in NetSuite](#)
- [Searching for a Product in NetSuite Connector](#)
- [Reloading Products in NetSuite Connector](#)

- Removing an Item from a Marketplace or Cart
- Deleting a Variation Option in NetSuite Connector
- Mapping Prices in NetSuite Connector
- Configuring Price Calculations Before Sync in NetSuite Connector
- Accommodating Repricer in NetSuite Connector Product Sync
- Configuring Quantity Calculations Before Sync in NetSuite Connector
- Subtracting Backordered Quantities for Multiple Locations in NetSuite Connector
- NetSuite Connector Product Sync FAQ
- Information for Troubleshooting Inventory Sync Issues in NetSuite Connector
- Troubleshooting Product Sync Issues in NetSuite Connector
- NetSuite Connector Product Sync FAQ

Setting Up Full Product Sync Mapping

Full product sync mapping setup varies depending on the storefront to which you are posting items.

Setting Up Price and Quantity Sync Mapping

Use the following procedure to set up price and quantity sync mappings.

To set up price and quantity sync mapping:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Product > Mappings.
4. From the **Category** list, select **Add Category**.
5. Select the category that ends with **PriceQty**. For example, if you are setting price and quantity sync for Walmart, select **WalmartPriceQty**.
6. Click **Add Category**.
7. Add the following fields:
 - a. **Flag Field** – If the NetSuite Connector SuiteApp is not installed in your account, create a custom item field for each storefront. If the flag field is a manually created, it can be a single character free-form text field or a list field. Otherwise, the field must be a single character free-form text field. For more information about flag fields, read the following topics:
 - [NetSuite Connector Storefront Flag Fields](#)
 - [Creating a Single-Character Storefront Flag Field Manually](#)
 - [Creating a List-Based Storefront Flag Field Manually](#)
 - b. **Category** – For carts, NetSuite Connector uses category mapping to determine whether an item should be listed using that category.
For more information, read the following topics
 - [Mapping Amazon Categories in NetSuite Connector](#)
 - [Mapping Cart Categories in NetSuite Connector](#)
 - c. **Price** – The price of your items. NetSuite Connector can use NetSuite price levels, both quantity and non-quantity based, or a custom field.

For more information, read [Mapping Prices in NetSuite Connector](#).

- d. **Quantity/Qty/Inventory** – Different storefronts name quantity differently. NetSuite Connector can pull quantity available from NetSuite locations or warehouses (either single location or NetSuite Connector can pull from multiple locations), or a custom field.
For more information, read [Mapping Inventories in NetSuite Connector](#).
- e. **sku** – This field is required to give NetSuite Connector the ability to match the updates sent to the correct items in the storefront. The **SKU** field that you map here should match the **SKU** field you identified in your order settings.

Following are the specific changes that you need to make for Amazon, eBay, and Walmart storefronts:

- **Amazon** – Add **AmazonInventoryPrice** category and set up the above mappings in that category.
- **eBay** – Set the **eBay Flag** field for price and quantity sync.
Set the above mappings in the **eBayFixedPrice** category. If you plan to update variation listings, set the above mappings in the **eBayVariations** category.
For more information, read [NetSuite Connector Storefront Flag Fields](#).
- **Walmart** – Add the **WalmartPriceQty** category. and set the above mappings in that category.

Setting the Real-Time Price and Quantity Sync Preference

If you want to enable or disable the real-time Price and Quantity sync in NetSuite Connector, you must set this preference in NetSuite.

To set the real-time Price and Quantity sync preference:

1. Go to Setup > Company > General Preferences.
2. Click the **Custom Preferences** subtab.
3. On the FarApp field group, set the **Disable Real-Time Sync** preference.
If you clear this box, NetSuite Connector will enable real-time sync.
If you check this box, NetSuite Connector will still sync updates for items based on the last modified timestamp on the item fields. This process typically occurs every 45 minutes. This update only affects NetSuite Connector, and not the storefront. Item updates to the storefront is a separate process that runs every 60 minutes by default.
4. Click **Save**.

Disabling Real-Time Sync

When real-time sync is disabled, NetSuite Connector updates items based on the last modified timestamp on the item fields and the sync occurs every 45 minutes.

To disable real-time sync from NetSuite:

1. Go to Connector > Configuration > Setup.
2. Check the **Disable Real-Time Sync** box.
3. Click **Submit**.

You can also disable real-time sync from the Product Sync tile in the Manage Data Syncs page in NetSuite Connector.

Creating a Flag Field for Product Sync

You can create a flag field using any of the following methods:

1. **Using the NetSuite Connector SuiteApp** – When you install the NetSuite Connector SuiteApp, the SuiteApp automatically creates the flag field and the associated list.
2. **Creating a single character flag field** – NetSuite Connector enables you to create a single character field with specific requirements that it can read instead of a list-based flag field. The drawback with this method is that there will be an error if you enter an invalid character.
For more information, read [Creating a Single-Character Storefront Flag Field Manually](#)
3. **Creating a list-based flag field** – For more information, read [Creating a List-Based Storefront Flag Field Manually](#).

Flagging Existing NetSuite Items to Sync

In majority of the integrations with NetSuite Connector, NetSuite is treated as a primary record for your items. NetSuite Connector syncs that primary record to your marketplaces and carts. With this approach, the product and inventory updates made in your NetSuite account are propagated to the marketplaces and carts automatically.

The action taken by NetSuite Connector for product sync when you make changes to the item or inventory in NetSuite is determined by the flag field. The flag field is a custom field on the NetSuite item record. The value set in the flag field determines how NetSuite Connector handles the item for product sync.

You can locate the flag field for your marketplace or cart from the Product Mappings page in NetSuite Connector. The value mapped to the **Flag Field (default)** is the field ID for your flag field in NetSuite.

For more information on the flag fields, read [NetSuite Connector Storefront Flag Fields](#).

To flag an item to sync:

1. Go to Lists > Accounting > Items.
2. Click **Edit** next to the item that you want to flag.
3. Click one of the following subtabs:
 - If you have NetSuite Connector SuiteApp installed, click the subtab that the SuiteApp has created. For example, if the SuiteApp has created Shopify fields, you will see the subtab called **Shopify**.
 - If you have manually created the flag field, click the subtab where you have created the field. By default, the field is created in the **Custom** subtab.
4. Hover over the name of a field to view the field ID.
The ID that matches the value for **Flag Field (default)** field ID in NetSuite Connector is the flag field for the item in NetSuite.
5. Perform one of the following actions:
 - For a flag field created by the NetSuite Connector SuiteApp, select a value from the list.
 - For a manually created flag field, enter a value from the single character storefront flag fields.
For more information about single character storefront flag fields, read [Creating a Single-Character Storefront Flag Field Manually](#).

For more information on storefront flag field leading practices, read the help topic [Storefront Flag Field Leading Practices](#).

Creating a List-Based Storefront Flag Field Manually

Before proceeding, make sure you have reviewed all other options mentioned in [Creating a Flag Field for Product Sync](#).

Creating a Custom List

If you have an existing working flag field, you can reuse the list used by that field, except for eBay connector.

Skip this section when you are using an existing list.

To create a custom list:

1. Go to Customization > Lists, Records, & Fields > New.
The Custom List page opens.
2. In the **Name** field, enter a name for the custom list.
3. In the **ID** field, enter the ID for the custom list.
4. From the **Show Options In** field, choose **Order Entered**.
5. Make sure **Matrix Option List** and **Inactive** check boxes are cleared.
6. In the **Values** sublist, add the following values in the given order:
 1. Add/Update Item
 2. Remove Item
 3. Ignore Item
 4. Post Children as Stand-Alone
7. Click **Save**.

Creating a Custom Field

After creating the custom list, create a custom field.

To create a custom field:

1. Go to Customization > Lists, Records, & Fields > Item Fields > New.
The Custom Item Field page opens.
2. In the **Label** field, enter the display name for the field on the item record.
3. In the **ID** field, enter the ID
This ID is used for the flag field in product mappings.
4. From the **Type** list, select **List/Record**.
5. From the **List/Record** list, select the custom list.
6. From the **Applies To** subtab, check the boxes for the item types on which you want the field to appear.
7. (Optional) To determine where the field appears on the item record, click the **Display Type** subtab, from the **Subtab** field, select the subtab.
If the subtab is not selected, the field appears in the **Custom** subtab.

8. Click **Save**.

- NetSuite Connector Product Sync Management
- Creating a Flag Field for Product Sync
- Flagging Existing NetSuite Items to Sync
- Creating a Single-Character Storefront Flag Field Manually
- Troubleshooting Product Sync Issues in NetSuite Connector
- NetSuite Connector Product Sync FAQ

Creating a Single-Character Storefront Flag Field Manually

For using product sync in the following connectors, you should create a flag field and item fields using the procedure provided in the topic [Creating a Single-Character Storefront Flag Field Manually](#):

- Amazon
- BigCommerce
- ChannelAdvisor
- eBay
- Jet
- Magento
- Magento2
- Mivo
- Shopify
- Walmart
- WooCommerce

If you are not using any of the above marketplaces or carts, manually create the flag field.

Creating a Flag Field Manually

If you do not install the NetSuite Connector SuiteApp in NetSuite, create a custom item fields for each storefront flag. The fields must be single character, free form text fields.

For more information, read [Adding NetSuite Custom Fields](#).

Items with the flag set to **Y** are added or updated to the storefront. Items with the flag set to **N** are deleted from the storefront.

NetSuite Connector loads all items from NetSuite that have a value in the **Storefront Flag** field. The items that do not have a value in the **Storefront Flag** field are ignored. By flagging only the items that need to be posted removes clutter, reduces errors, and improves the processing time for your products.

You can use the following values for a flag field:

Value	Definition	Notes
(blank)	Ignored by NetSuite Connector.	NetSuite Connector does not load data from NetSuite if the flag field is blank.

Value	Definition	Notes
Y	Add or update product data to the storefront.	.
N	Delete product or listing from storefront.	If an item is flagged as Y , and later you decide to delist the item, you must flag it to N . i Note: If you are using Price/Qty sync, setting the item to N does not delete the item from the marketplace or cart. The N setting only stops syncing the item.
C	Parent of stand-alone items. C means post only children.	Used for NetSuite matrix item parents. Child items are posted as stand-alone (non-variation) items. NetSuite Connector needs the parent data to populate child items. This setting enables NetSuite Connector to load, but not post the parent data. To utilize this setting, set the parent item to C and child item to Y .
Q	Inventory-only flag. Supported only for eBay connector.	This value is used for posting only inventory updates to eBay when you do not want full product updates.
P	Price-only flag. Supported only for eBay connector.	This value is used for posting only price updates to eBay when you do not want full product updates.
I	Price and inventory flag. Supported only for eBay connector.	This value is used for posting only price and inventory updates to eBay when you do not want full product updates.

When using the storefront flag field, follow the leading practices outlined in the topic [Storefront Flag Field Leading Practices](#).

- [NetSuite Connector Product Sync Management](#)
- [Creating a Flag Field for Product Sync](#)
- [Flagging Existing NetSuite Items to Sync](#)
- [Creating a List-Based Storefront Flag Field Manually](#)
- [Troubleshooting Product Sync Issues in NetSuite Connector](#)
- [NetSuite Connector Product Sync FAQ](#)

Mapping Product or Inventory Sync in NetSuite Connector

The Product Mapping page enables you to setup item mappings from NetSuite to your storefronts and marketplace channels. NetSuite Connector will use these mappings to create and update product listings on those channels.

To map product or inventory sync data:

1. Login to NetSuite Connector.
2. Select a connector.
3. Go to Mappings > Products.
4. Click **Add Mapping**.
5. Complete the **Add Field Mapping** window:
 - On the **Required** row, select either **Category** or **SKU**.

- ON the **Standard** row, select a NetSuite field.
6. Click **Add Mapping**.
 7. Click the new mapping that is added at the bottom of the table.
 8. Complete the Mapping window:
 - If the mapping type is Category mapping:
 1. On the **NetSuite Field ID** row, enter a NetSuite Field ID.
 2. On the **Value** row, enter a value.



Note: NetSuite Connector uses the Category mapping for a category to determine whether a particular item should be listed using that category. There are two typical ways to set up this mapping. First, we can check whether a certain field is populated; all items that have a value in that field will be categorized as 'ShopifyPriceQty' items, in this case. Place a value in the NetSuite Field ID box (and leave Value blank) to use this approach. Alternatively, we can check whether your chosen field has a certain value for items, and only items having that value will be categorized as 'ShopifyPriceQty'. Populate the NetSuite Field ID and Value boxes to use this approach.

- If the mapping type is SKU mapping:
 1. On the **Mapping Type** column, select either **Item Field**, **Item Field Translation**, or **Logic**.
 2. On the second column:
 - If **Item Field** is selected in the **Mapping Type** column, enter a value in the **NetSuite Field ID** column.
 - If **Item Field Translation** is selected in the **Mapping Type** column:
 - a. Click **Translation Mapping – Click to Edit**.
 - b. In the first row:
 - Enter a value in the **NetSuite Value** column.
 - Enter a value in the second column.
 - c. Enable the **On/Off** switch.
 - d. Enter a value in the **NetSuite Field** field.
 - e. Click **Save Changes**.
 - If **Logic** is selected in the **Mapping Type** column:
 - a. Click **Logic Mapping – Click to Edit**.
 - b. Complete the **Edit Logic Mapping for 'sku' (BETA)** window.
 - In the **Check If** row:
 - i. In the first column select a condition for the filter.
 - **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - **Fixed Value** – A constant value (something that does not change from order to order. An example would be a number '1'.
 - **Computed Value** – a value that is a result of a condition or an equation or operation.
 - ii. In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:
 - A. In the second column, select an order header value from the dropdown.
 - B. In the third column, select a condition how to evaluate from the dropdown.
 - C. In the fourth column, select either **Fixed Value, Order Header Value, Computed Value**.
 - D. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 - A. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - B. In the third column, select a condition how to evaluate from the dropdown.
 - C. In the fourth column, select either **Fixed Value, Order Header Value, Computed Value**.
 - D. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 - A. In the second column, click **Click to View/Edit Value**.
 - B. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 - C. Complete the form.
 - ◆ If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - ◆ If **Calculate Value** is selected:
 - I. In the Set Value To row, select either **Order Header Value, Fixed Value**, or **Computed Value** in the first column.
 - II. In the second column:
 - ◊ If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - ◊ If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - ◊ If **Computed Value** is selected in the last column, follow the steps listed in

editing the filters for the **Computed Value** of the form.

- III. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.

IV. Click **Save Changes.**

- In the **If True, Set Value To** row:

In the first column select a condition for the filter.

- **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
- **Fixed Value** – A constant value (something that does not change from order to order). An example would be a number '1'.
- **Computed Value** – a value that is a result of a condition or an equation or operation.

In the second, third, fourth, and fifth column:

- If **Order Header Value** is selected in the first column:
 - i. In the second column, select an order header value from the dropdown.
 - ii. In the third column, select a condition how to evaluate from the dropdown.
 - iii. In the fourth column, select either **Fixed Value, Order Header Value, Computed Value**.
 - iv. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Fixed Value** is selected in the first column –
 - i. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - ii. In the third column, select a condition how to evaluate from the dropdown.
 - iii. In the fourth column, select either **Fixed Value, Order Header Value, Computed Value**.
 - iv. In the fifth column, enter a value to evaluate against in the **Value 2** field.
- If **Computed Value** is selected in the first column –
 - i. In the second column, click **Click to View/Edit Value**.
 - ii. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
 - iii. Complete the form.
 - ◆ If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - ◆ If **Calculate Value** is selected:

- A. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - B. In the second column:
 - ◊ If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - ◊ If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - ◊ If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - C. In the third column, select an operation to do to the values, conditions, or filters in steps 1-2.
 - D. Click **Save Changes**.
- In the **Otherwise, Set Value To** row:
- In the first column select a condition for the filter.
- ◊ **Order Header Value** – These are values that give general information about an order, such as billing and shipping information, order totals, the order date, and more.
 - ◊ **Fixed Value** – A constant value (something that does not change from order to order. An example would be a number '1'.
 - ◊ **Computed Value** – a value that is a result of a condition or an equation or operation.
- In the second, third, fourth, and fifth column:
- ◊ If **Order Header Value** is selected in the first column:
 - i. In the second column, select an order header value from the dropdown.
 - ii. In the third column, select a condition how to evaluate from the dropdown.
 - iii. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - iv. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - ◊ If **Fixed Value** is selected in the first column –
 - i. In the second column, enter any value to evaluate the condition against in the **Value 1** field.
 - ii. In the third column, select a condition how to evaluate from the dropdown.
 - iii. In the fourth column, select either **Fixed Value**, **Order Header Value**, **Computed Value**.
 - iv. In the fifth column, enter a value to evaluate against in the **Value 2** field.
 - ◊ If **Computed Value** is selected in the first column –
 - i. In the second column, click **Click to View/Edit Value**.

- ii. In the Value type row, select either **Select Value Conditionally** or **Calculate Value**.
- iii. Complete the form.
 - ◊ If **Select Value Conditionally** is selected, the form follows the same rules and features as the entire **Edit Logic Mapping** form.
 - ◊ If **Calculate Value** is selected:
 - A. In the Set Value To row, select either **Order Header Value**, **Fixed Value**, or **Computed Value** in the first column.
 - B. In the second column:
 - ◊ If **Order Header Value** is selected in the first column, select an order header value from the dropdown.
 - ◊ If **Fixed Value** is selected in the first column, enter a value to evaluate against in the **Value 1** field.
 - ◊ If **Computed Value** is selected in the last column, follow the steps listed in editing the filters for the **Computed Value** of the form.
 - C. In the third column, select an operation to do to the values, conditions, or filters in steps 1–2.
 - D. Click **Save Changes**.

Note that in rows two and three, the Location we use is set when the condition is true, and the condition to use when it's false.

Viewing Product Mapping in NetSuite Connector

You can preview the data NetSuite Connector sends to the marketplace or cart for a product. There are the following two places from where you can view the data:

1. From the Products dashboard
2. From the Products Mappings page

To view data mapping from the Products dashboard:

1. Go to Data Flows > Products.
The Products dashboard opens and displays the products imported by NetSuite Connector.
2. Search the product for which you want to view the data mapping.
For more information, read [Searching for a Product in NetSuite Connector](#).
3. After locating the product, click the **Action** menu on the right of the product.
4. Select **Show Product Mapping**.
The product mapping test results appear.

To view data mapping from the Product Mappings page:

1. Go to Mappings > Products.

2. Click **Test Mappings**.

The Test Product Mappings window opens.

3. In the **Product Internal ID** field, enter the internal ID of product.



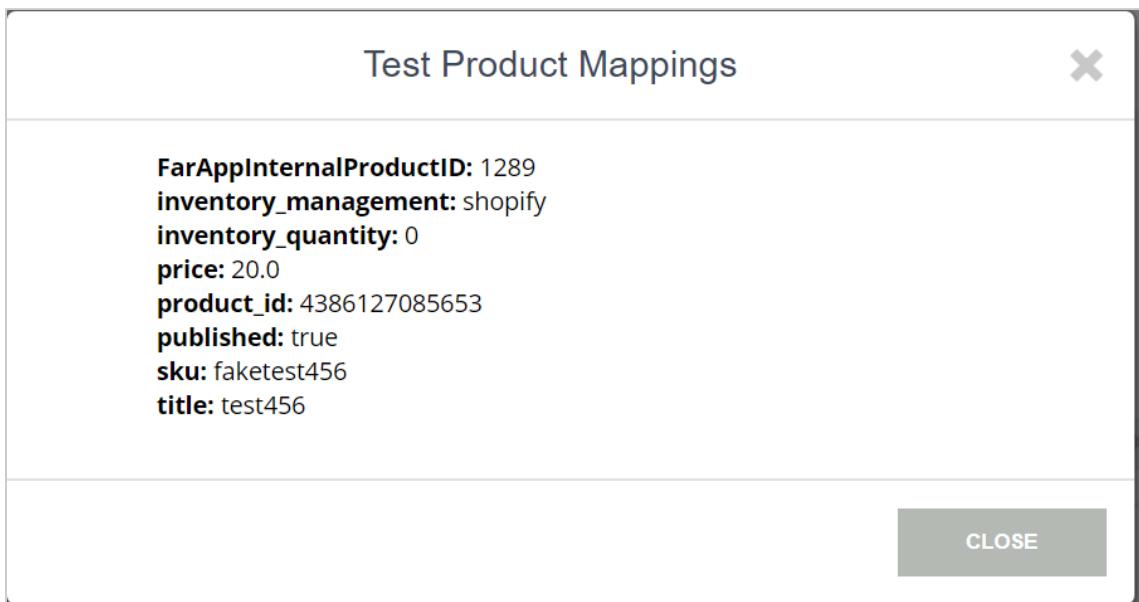
Note: Do not enter the SKU.

4. Click **Test Mappings**.

The product mapping test results appear.

Product Mapping Test Results

The Test Product Mappings window displays the results of the mappings. In addition to the current mappings results for the test product, NetSuite Connector also displays certain values that are not mapped. These values are provided for information purpose. For example, the **FarAppInternalProductID** is both the NetSuite internal ID and the ID NetSuite Connector assigns to the product for internal processes.



Note: When you test mappings for a product that has been sent to the marketplace or cart, the mappings that appear are of the time of viewing. You do not see the mappings of the time when the product was actually sent to marketplace or cart.

Mapping Required and Optional Fields for Product Sync

Product mapping can be either required or optional. The required or optional fields vary across different marketplaces and carts. Often, they require many of the same sets of fields (such as product name or SKU) but use different field IDs for the same data. Each marketplace or cart may also have unique fields that require mapping.

The following procedure details how to determine required fields and map them.

To map required product fields:

1. Log in to app.farapp.com
 2. Select the connector and relevant account.
 3. Go to Mappings > Products.
- NetSuite Connector displays a list of existing product mappings.
4. Click **Add Mapping**.
- The Add Field Mapping popup window appears and has three dropdown lists: Required, Standard and Uncommon.
5. From the **Required** list, identify any field that does not have an existing product mapping.
 6. Select the field you want to map then click **Add Mapping**.
- The field is added to the end of the product mapping list.
7. Click **Close**.
 8. Locate the new field at the bottom of the product mappings list. Under the NetSuite Field ID column, enter the field ID you want to map to this field. As you type, NetSuite Connector will automatically suggest and complete options based on fields IDs it detects in NetSuite.
- i Note:** Some required fields do not allow you to enter the mapping directly but rather shows **Special Mapping – Click to Edit**. You need to click on this link to enter the mapping.
9. Click **Save**.
 10. Repeat steps 5–9 to ensure all required fields are mapped.

You can follow the same procedure to add optional fields from the Standard and Uncommon lists. It is best practice to complete mapping all required fields first then add optional fields one at a time.

Mapping Cart Categories in NetSuite Connector

This topic provides details to create a category mapping for the following:

- All carts
- Marketplaces that do not support full product sync, such as Wayfair and Overstock

To map a category for a cart:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Mappings > Products.
 4. Click the correct sync type button that you are setting up.
- Some connectors do not provide full product sync, so you may not see the **Full Product Sync** button for those connectors.
5. Locate the flag field mapping in the list of product mappings.
- The field should contain the NetSuite field ID. Note the field ID.
6. Click **Add Mapping**.
- The Add <marketplace/cart> Field Mapping window opens.
7. From the **Required** list, select **Category**.
 8. Click **Add Mapping**.
 9. Click **Close**.

The **CATEGORY** field now appears at the bottom of the product mappings list.

10. Locate the **CATEGORY** mapping, and click **Special Mapping – Click to Edit**.
The CATEGORY Mapping window opens.
11. In the **NetSuite Field ID** field, enter the flag field ID that you noted in step 5.
12. Click **Save Changes**.

For more information, read [NetSuite Connector Product Categories](#).

Mapping Inventories in NetSuite Connector

You can sync inventory data during product sync. The field name for inventory can vary in marketplaces or carts. The field name usually contains words like **quantity**, **inventory**, acronyms like **qoh**, or abbreviation like **qty**. If you are not sure of the field used for inventory, after adding a field to the mappings, hover over the field name. The tooltip may provide further clue on whether the field is used for inventory.

You can use the standard inventory mapping procedure in all cases, except the following:

- Shopify connector with multi-location setup – For more information, read [Mapping Shopify Multilocation Inventory in NetSuite Connector](#).
- Magento2 with multi-warehouse (applicable to Magento 2.3X and later using multiple warehouses in Magento2).

To create a standard inventory mapping:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The inventory_level Mapping window opens.



Note: If you do not see the inventory field or the field that storefront uses for inventory , click **Add Mapping** and add the field to your mapping.

6. To select one or more locations to sync inventory, from the **Sum Quantity from the Following Locations** list box, press Ctrl (or Command for Mac) and select multiple locations.
7. Click **Save Changes**.
8. Click **Save**.

Mapping Custom Field Inventories in NetSuite Connector

During product sync, NetSuite Connector by default syncs inventory number on items from one or more designated locations in NetSuite. However, to pull the inventory number from a custom field in NetSuite, use the following procedure.

To map inventory from a custom field:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The Inventory Mapping window opens.



Note: If you do not see the inventory field or the field that storefront uses for price, click **Add Mapping** and add the field to your mapping.

6. Check the **Draw Quantity From Custom Field Rather Than Inventory Locations** box.
7. In the **Custom Quantity Field**, enter the custom field ID.
8. Click **Save Changes**.
9. In the Reload Items window, click **Yes, Reload My Items**.
Reloading may take some time.
10. Click **Save**.

Blank Value Field for Category Product Mapping in NetSuite Connector

NetSuite Connector uses the **Category** field to determine which set of mappings to use for a product. Most of the storefronts use only a single mapping category. In such cases, a special mapping rule is used with the flag field that checks if the flag field is not empty. If the flag field is not empty, then NetSuite Connector knows which mapping category to use.

The logic is that NetSuite Connector checks if an item should sync to a storefront. If yes, then use the selected mapping category, because there is only a single mapping category in use. When this logic is used, NetSuite Connector displays the field ID of the flag field in the **NetSuite Field ID** field and a blank **Value** field.

A blank **Value** field in your Category mapping is not necessarily an indication of an issue if the **NetSuite Field ID** field is populated.

Removing or Resetting Data for a Field in the Storefront Using Product Sync

When you keep a field blank in NetSuite, NetSuite Connector does not automatically keep the corresponding field in the storefront blank.

When no data is provided for a mapped NetSuite field and if a default value is configured for that field's mapping, NetSuite Connector uses the default value. When you keep a field blank in NetSuite, no data

is sent to NetSuite Connector for that field. Therefore, NetSuite Connector falls back to the designated default value, if provided. You can set NetSuite Connector to send a single character (or 0 for numeric fields) for the default value in your product sync mappings. Following are some key points to consider when sending default values:

- Sending a space character or 0 value for some fields can cause issues. Make sure you are aware of the consequences of sending blank data for the fields in question.
- You cannot send a space character or 0 value for required fields. Sometimes, you can set them in NetSuite Connector, but the storefront will mostly reject the product posting.

To set a space or 0 value as the default value for a field in NetSuite Connector:

1. Log in to app.farapp.com.
 2. Select the connector and then select the relevant account.
 3. Go to Mappings > Products.
 4. Locate the mapping of the field for which you want to set a default value and in the **Default Value** field, press Spacebar (or 0 for numeric fields).
-  **Note:** Not all fields permit a default value to be set. The availability depends on the field and mapping type.
5. Click **Save**.

Updating the SKU of Items Posted with NetSuite Connector

For items posted on a storefront using NetSuite Connector, you must not update the item SKU (stock keeping unit) directly on the item record. This can cause problems with order and product synchronization with the NetSuite Connector. If you must update the item SKU, do the following steps to avoid any issues.

To update SKU of items posted with NetSuite Connector:

1. Set the item whose SKU you want to update, to inactive.
For more information, see the help topic [Working with Inventory Items](#), Making an Item Inactive.
2. On the inactive item record, change the SKU to indicate that it is removed or obsolete. For example, you can append the word 'removed' or 'obsolete' to the SKU, like ItemCode-obsolete.
3. Create a new item record with a new or updated SKU.
For more information, see the help topic [Creating Item Records](#).

Fixing SKUs that Do Not Match the Storefront SKUs

NetSuite Connector requires that your Stock Keeping Units (SKU) match in both NetSuite and in the storefront. Without this association, orders and products cannot sync between the two platforms. If the SKUs do not match, create a custom field on the item record in NetSuite and update that field with the corresponding SKU from storefront. With this approach, NetSuite Connector can match items based on the custom field in order and product syncs. Use the following procedure to change the NetSuite SKU field in NetSuite Connector. After you change the SKU field using the procedure, all items should have the corresponding SKU value in that field.

To change the NetSuite SKU field in NetSuite Connector:

1. Log in to [app.farapp.com](#).
2. Select the connector and then select the relevant account.
3. Go to Setting > Order > General.
4. In the **NetSuite SKU Field ID** box, enter the custom SKU field name.
5. Click **Save**.

Setting Maximum Quantity for Inventory During Product Sync

In some cases, you may want to set a maximum limit, or ceiling on the quantity that NetSuite Connector syncs for products. When set, if the actual quantity is more than the maximum limit, NetSuite Connector syncs only the maximum limit quantity. For example, if you set the maximum limit of 25 and the quantity on your item is 30, NetSuite Connector syncs only 25 quantity.

To set maximum quantity for inventory during product sync:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The inventory_level Mapping window opens.



Note: If you do not see the inventory field or the field that storefront uses for inventory , click **Add Mapping** and add the field to your mapping.

6. In the **Maximum Value (Optional)** field, enter the maximum limit value.
7. Click **Save Changes**.
8. Click **Save**.

Setting Minimum Quantity for Inventory During Product Sync

In some cases, you may want to set a minimum limit, or floor on the quantity that NetSuite Connector syncs for products. When set, if the actual quantity is less than the minimum limit, NetSuite Connector syncs only the minimum limit quantity. For example, if you set the minimum limit of 5 and the quantity on your item is 4, NetSuite Connector syncs 5 quantity.

To set minimum quantity for inventory during product sync:

1. Log in to [app.farapp.com](#).

2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The inventory mapping window opens.



Note: If you do not see the inventory field or the field that storefront uses for inventory , click **Add Mapping** and add the field to your mapping.

6. In the **Minimum Value (Optional)** field, enter the minimum limit value.
7. Click **Save Changes**.
8. Click **Save**.

Adding Product Images for NetSuite Connector

Most marketplaces or carts require at least one product image to create a new product. To create a new product listing using NetSuite Connector, include at least one image on each item record in NetSuite that you sync with NetSuite Connector.

You can add images by either hosting your images directly on NetSuite or by hosting using a third-party hosting solution. You can also create a custom field in NetSuite so you can sync more images. After adding the images, you must map them so NetSuite Connector can post them in your marketplace or cart.

- [Hosting Images in NetSuite](#)
- [Hosting Images with a Third-Party](#)
- [Creating a Custom Image Field and Uploading Images to that Field](#)
- [Mapping Image Fields](#)

Hosting Images in NetSuite

Hosting images in NetSuite uses the File Cabinet feature. Before proceeding, make sure you have the image file ready to upload.

For information on uploading images, read the help topic [Upload Images to the File Cabinet](#).

Adding an Image to an Item

After uploading the image to the file cabinet, add the image to the item. You can upload the image, either by selecting the image in one of the native NetSuite image fields or a custom item field of type **Image**.

To add an image to an item:

1. Go to Lists > Accounting > Items.

2. From the list page, click **Edit** next to the item to which you want to add the image.
3. Click the **Web Store** subtab.



Note: The subtab can be different if you are using a custom field.

4. Click the double arrow in the **Item Display Image** field and click **List**.
5. Search for and select the desired image.
6. Click **Save**.

Hosting Images with a Third-Party

If you host your images with a third-party, you will need the image URL. The image URL must be publicly accessible and must end with image file extension such as jpg or gif. For example, <http://www.yourhoster.com/yourimage.jpg>.

You will then have to create a custom item field of type **Free-Form Text**.

Then, add the image URL to the custom item field in the item record.

To add the image URL to the item:

1. Go to Lists > Accounting > Items.
2. From the list page, click **Edit** next to the to which you want to add the image.
3. Locate the field you are using to store the URL and enter the URL.
4. Click **Save**.

Creating a Custom Image Field and Uploading Images to that Field

The following procedure provides steps to create a custom image field in NetSuite uploading the images to that field.

To create a custom image field and upload images to that field:

1. Go to Customization > Lists, Records, & Fields > Item Fields > New
2. On the Custom Items Fields page:
 - a. In the **Label** field, enter the label for the image field.
 - b. In the **ID** field, enter the ID for the field.
 - c. From the **Type** list, select **Image**.
 - d. Click **Applies To** subtab select all the item types to which you want this field to apply.
 - e. Click **Save**.
3. Open an item of a type that you selected when creating the field in **Edit** mode.
4. Locate the field in the item record.
5. Hover over the field and click the Plus icon that appears next to the field..
The File popup window opens.
6. In the popup window:
 - a. Click **Choose File**.

- b. From the Open window select the file and click **Open**.
- c. Check the **Available Without Login** box.
- d. Click **Save**.
- e. On the Items page, click **Save**.

The custom image should display under the custom image field in the item record.

Mapping Image Fields

With the images set up in the NetSuite, you must update the mappings to pull the image fields into NetSuite Connector to post them to the marketplace or cart.

To map image fields:

1. Log in to NetSuite Connector.
 2. On the left menu panel, select the relevant connector and account combination.
 3. Go to Mappings > Products.
 4. Click **Add Mapping**.
 5. On the Add Mapping window, click the **Standard** dropdown field to display a list of fields for mapping. Other fields are listed under the Required and Uncommon dropdown fields.
 6. From the list of fields, locate the name of the images field for the relevant connector. The field varies depending on the connector.
 7. Click **Add Mapping**.
- The mapping will be added at the end of your list of mappings.
8. Click **Close**.
 9. Locate your new mapping in the Product Mappings list and click the **Special Mapping - Click to Edit link**.
 10. On the images Mapping window, in the **Image Field**, enter the field ID of the NetSuite field that contains the image data to send to the marketplace or cart.
- Some marketplaces or carts allow you specify a set of images, while others provide multiple image fields.
- For example, Shopify allows multiple images to be specified using a single field. If you have multiple image fields to map in Shopify, click **Add Row** to have more mapping fields.
11. After completing the mappings, click **Save Changes**.
 12. To test your new mappings, make a change to a product in NetSuite that NetSuite Connector is syncing with. Ensure that the field you are using for images has data populated in NetSuite. The change will trigger an automatic sync by NetSuite Connector.
- For more information on manually syncing a test product, see [NetSuite Connector Manual Sync Tests](#).

Configuring Assembly Item Quantities in NetSuite Connector

NetSuite Connector can sync both the quantity of assembled items and the quantity of buildable items from designated locations for assembly items. If you want to sync only the quantity of assembled items, you must set your preference on the product mappings.

To change the sync quantity for assembly items:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Mappings > Products.
- The Product Mappings page appears.
4. Depending on the connector, select the sync type or the marketplace category:
 - For BigCommerce, Shopify, and other similar connectors, on the **Select Sync Type** section, click either the **Price/Quantity Only** or **Full Product** button. The option you select depends on the sync type of the product mapping.
 - For Amazon, Walmart, and other similar connectors, select the generic marketplace category from a dropdown list. If you are mapping the inventory for a marketplace connector, the sync type buttons will not show.
 5. Locate the inventory field. For example, under the BigCommerce Field column, find the inventory field, which is **inventory_level**.
If you do not see the inventory field, use the **Add Mapping** button to add this field to your product mappings.
 6. Edit the product mapping.
 - a. Click the **Special Mapping - Click to Edit** link under the **NetSuite Field ID** column.
The inventory_level Mapping page appears.
 - b. Select the appropriate locations.
 - c. Clear the **For Assembly Items, Include Buildable Quantity When Syncing Total** box.
When you clear this box, NetSuite Connector will sync only the assembled quantity of assembly items from your designated location. Buildable quantity or the number of items that can be assembled is excluded.
This preference is available in both location-based and custom field type inventory mapping.
 - d. Click **Save Changes**.
 7. On the Product Mappings page, click **Save**.

Batch Updating Items in NetSuite

NetSuite enables you to start batch updates on item records (and other records) based on your set criteria. You can update your item records for just about any field including flag fields, tags, descriptions and more. For more information, read the help topic [Defining a Mass Update](#)

Searching for a Product in NetSuite Connector

All products imported in NetSuite Connector can be viewed in NetSuite Connector's Products dashboard.

To search for a product in NetSuite Connector:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Data Flows > Products.
- The Products dashboard opens. By default, the **Show Children** filter is set to show parent items as well as child items.

4. (Optional) To change the filter, point to **Filter** and from the filter list, select the required filter.
 5. In the **Search Products** field, enter the product SKU.
 6. Press **Enter**.
- The results are displayed in the list.

Reloading Products in NetSuite Connector

Reload the products in NetSuite Connector using the one of the following options:

- [Updating Items in NetSuite](#)
- [Updating Products in NetSuite Connector in Bulk](#)

Updating Items in NetSuite

When you make changes to items in NetSuite, NetSuite Connector automatically marks them to be reloaded when the product sync runs.



Note: Note that the item's flag field must be set to **Add/Update Item** or **Y** if using single character flag field.

Updating a Product in NetSuite Connector

Use the following procedure to manually reload a product from NetSuite Connector.

To update a product in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Products.
4. From the list of products, locate the product that you want to update.
5. Hover over the **Action** icon for the product and from the list, select **Reload from NetSuite**.

Updating Products in NetSuite Connector in Bulk

Use the following procedure to update multiple products in bulk from NetSuite Connector.

To update products in bulk in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Products.
4. (Optional) To change the number of products displayed on a page, select the desired number at the bottom of the page.
5. Perform one of the following operations:
 - To select all items on the page, check the left-side box in the column header.
 - To manually select items to reload, check the left-side box against the desired item rows.
6. From the **Bulk Actions** dropdown list, select **Reload Selected Products from NetSuite**.

7. (Optional) For more than 500 products, click **Next page** and repeat steps **5** and **6**.

Removing an Item from a Marketplace or Cart

Remove an item you no longer need to show on the marketplace or cart.



Note: To remove items through a sync, you must enable full product sync.

Deleting a Variation Option in NetSuite Connector

In this context, variation option means the way in which children SKUs (Stock Keeping Units) vary from each other. These options can be the options that you select from a list on the product page in the storefront. For example, if the children vary by color, then your variation option is color. You select a color from a list on the item when you purchase the item on the storefront. Similarly, if the children vary by color and size, then you have the variation options of color and size.

You can remove or update the variation options in the following ways:

- Manually remove or update the variation options in both NetSuite and storefront directly.
- Delete the item listings in your storefront and NetSuite Connector, then update the variation options in NetSuite on your family. Then let NetSuite Connector sync the updated family back to the storefront as a new family.

If you delete a variation from NetSuite but not from the storefront, you will get an error when posting the products in that family. The reason for the error is that the storefront expects existing variations but NetSuite sends something different.

For example, for children SKUs that vary on color and size, let's assume that you remove the size option from the family in NetSuite. If you do not remove the option from the storefront, you will get an error when syncing. The reason for the error is that the storefront still expects a value for size on all children.

Mapping Prices in NetSuite Connector

You can sync price data during product sync. The field name for price can vary in marketplaces or carts. The field name usually contains the word **price**, like **price**, **regular_price**, or **item-price**. If you are not sure of the field used for price, after adding a field to the mappings, hover over the field name. The tooltip may provide further clue on whether the field is used for price.

To create a mapping for price:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the price mapping, click **Special Mapping – Click to Edit**. The price Mapping window opens.

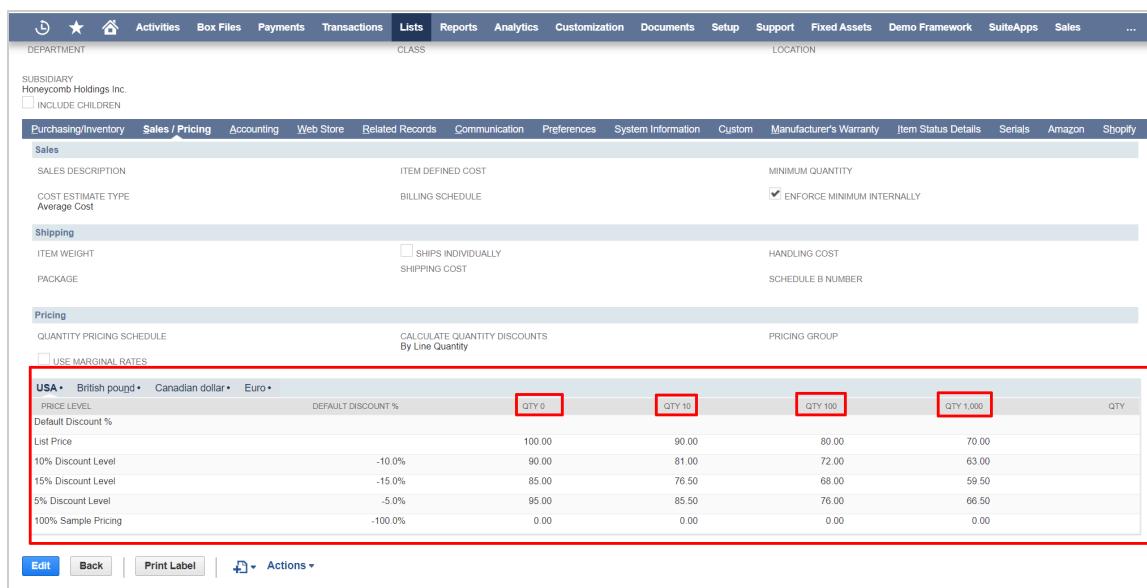
Note: If you do not see the price field or the field that storefront uses for price, click **Add Mapping** and add the field to your mapping.

6. From the **Map Price from the Following Price Level** list, select the price level.
7. Click **Save Changes**.
8. Click **Save**.

Mapping Price from a Quantity-Based Price Level

In NetSuite, you can set item pricing based on quantity of items sold.

In the following screenshot, you can see that the item has different pricing based on quantities 0, 10, 100, and 1000. NetSuite Connector can sync the price from any of these quantity-based prices from any level.



The screenshot shows the NetSuite Sales / Pricing page. At the top, there are tabs for Purchasing/Inventory, Sales / Pricing, Accounting, Web Store, Related Records, Communication, Preferences, System Information, Custom, Manufacturer's Warranty, Item Status Details, Serials, Amazon, and Shopify. The Sales / Pricing tab is selected. Below the tabs, there are sections for Sales Description (Item Defined Cost), Cost Estimate Type (Average Cost), Shipping (Item Weight, Package), and Pricing (Quantity Pricing Schedule, Default Discount %). A red box highlights the Pricing section, specifically the 'By Line Quantity' table. This table shows price levels for quantities 0, 10, 100, and 1,000. The table includes columns for Price Level, Default Discount %, QTY 0, QTY 10, QTY 100, QTY 1,000, and QTY. The data is as follows:

PRICE LEVEL	DEFAULT DISCOUNT %	QTY 0	QTY 10	QTY 100	QTY 1,000	QTY
List Price	100.00	90.00	80.00	70.00		
10% Discount Level	-10.0%	90.00	81.00	72.00	63.00	
15% Discount Level	-15.0%	85.00	76.50	68.00	59.50	
5% Discount Level	-5.0%	95.00	85.50	76.00	66.50	
100% Sample Pricing	-100.0%	0.00	0.00	0.00	0.00	

At the bottom of the page, there are buttons for Edit, Back, Print Label, and Actions.

Important: Do not create quantity-based price mappings in NetSuite Connector if the feature is not enabled in NetSuite. Else, the price mapping will fail.

To map quantity-based price field to product sync:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current price mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the price mapping, and click **Special Mapping – Click to Edit**.

The price Mapping window opens.



Note: If you do not see the price field or the field that storefront uses for price, click **Add Mapping** and add the field to your mapping.

6. From the **Map Price from the Following Price Level** list ,select a price level.
7. Check the **Using Quantity-Based Price Levels** box.
8. In the **Quantity Tier** field, enter the required quantity for the price.



Note: Make sure you enter the correct quantity that matches the quantity for the price level added in NetSuite item record.

9. Click **Save Changes**.
10. Click **Save**.

Mapping Price Levels from Specific Currency

If multiple currencies is enabled in NetSuite, the actual price in the price level can vary with currency. When mapping quantity-based price field to product sync, you can specify the currency in which NetSuite Connector takes for the price.

To map price levels from a specific currency:

1. Log in to app.farapp.com.
 2. Select the connector and the relevant account.
 3. Go to Mappings > Products.
 4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current price mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
 5. From the product mappings list, locate the price mapping, and click **Special Mapping – Click to Edit**.
- The price Mapping window opens.



Note: If you do not see the price field or the field that storefront uses for price, click **Add Mapping** and add the field to your mapping.

6. From the **Map Price from the Following Price Level** list ,select a price level.
7. Check the **Multiple Currencies Setup in NetSuite** box.
8. From the **Map Price Using The Following Currency** list, select the currency that you want to use.
9. Click **Save Changes**.
10. Click **Save**.

Mapping Price from a Custom Field During Product Sync in NetSuite Connector

NetSuite Connector's default method for syncing price on items during product sync is to designate a price level in NetSuite to pull the price from. To designate a custom field on items where you can enter the price that you want NetSuite Connector to sync, follow this procedure:

To designate a custom field on items where the price for syncing can be entered:

1. Log in to [app.farapp.com](#).
2. Select the relevant connector and account.
3. Go to **Mappings > Products**.
4. Click the button of the sync type you are mapping the price for.



Note: If you are mapping the price for a marketplace like Amazon or Walmart, a dropdown field appears instead of buttons. From the dropdown field, select the generic marketplace category, such as Amazon or Walmart.

5. Locate the price field you want to sync, and then click the **Special Mapping - Click to Edit** link of the field.



Note: If the field that a particular storefront uses for price does not appear, click **Add Mapping**.

6. On the Price Mapping page, check the **Draw Price From Custom Field Rather Than Price Level** box.
Checking this box means you are using a custom field instead of selecting a price level.
7. In the **Custom Price Field**, enter the field ID of the custom field that NetSuite Connector should reference the price from.
8. (Optional) Perform arithmetic operations on the mapped value before sending it to the storefront. For more information, see [Configuring Price Calculations Before Sync in NetSuite Connector](#).
9. Click **Save Changes**.
10. (Optional) If the Reload Items is displayed, click **Yes, Reload My Items**. NetSuite Connector will reload your items and pull in the data from the custom field you mapped. The more items you have in NetSuite Connector, the longer it will take to reload them from NetSuite.
11. Click **Save**.

Configuring Price Calculations Before Sync in NetSuite Connector



Note: NetSuite Connector discourages performing arithmetic operations with pricing because that causes the pricing in NetSuite to not match with the pricing synced to the storefront. The leading practice is to map price directly from a price level in NetSuite. If that is not possible, perform the additional operations on the prices directly in NetSuite using a custom field.

You can perform arithmetic operations on the price of the item before NetSuite Connector syncs to the storefront. You can perform actions like discount all prices by 10% before syncing and add a \$2 to the price to offset credit card processing fees.

In the following procedure, a field value sent from the price is reduced by 10% by multiplying the field value by 90% or .9.

To perform mathematical operations on price before sync:

1. Log in to [app.farapp.com](#).
2. Select the connector and the relevant account.
3. Go to Mappings > Products.

4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current price mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
 5. From the product mappings list, locate the price mapping, click **Special Mapping – Click to Edit**.
The price Mapping window opens.
- Note:** If you do not see the price field or the field that storefront uses for price, click **Add Mapping** and add the field to your mapping.
6. From the **Map Price from the Following Price Level** list ,select a price level.
 7. Click **Add Operation Row**.
 8. From the **Operations** list, select **Multiply**.
 9. In the **Value** field, enter **.9**.
 10. (Optional) To add more mathematical operations, repeat steps **7** and **8**.
 11. Click **Save Changes**.
 12. Click **Save**.

You can also perform the following additional operations in the price Mapping window:

Accommodating Repricer in NetSuite Connector Product Sync

If you have a repricer in the storefront to dynamically adjust item prices, NetSuite Connector overrides the repricer set prices if the following conditions are true:

- You are using NetSuite Connector for product sync.
- You have a mapping for the price field.

If you do not want NetSuite Connector to override the prices:

- Do not map the price field. Without a price mapping, product sync will not update the price on the items and repricer's price will remain.
- If you use full product sync and create items in the storefront with the sync, then storefront requires a price to create items. In such a case, set your price mapping to use a custom field and make sure by default no data is populated in that field. After the item is created in the storefront, remove the data from the custom field so that the repricer can control the price.

Note: If you manually delete the item from NetSuite Connector, NetSuite Connector forgets that it has posted the item, causing the next posting to include the price.

Configuring Quantity Calculations Before Sync in NetSuite Connector

You can perform one or more arithmetic operations on the item quantity before NetSuite Connector syncs the item quantity to the storefront. You can perform addition, subtraction, multiplication, and

division operations on the quantity. A common use of this feature is to create a buffer for the quantity by always subtracting a set amount from the quantity before syncing.

In the following procedure, a buffer of 5 is set on the synced quantity by subtracting the 5 from the syncing quantity.

To perform mathematical operations on inventory before quantity sync:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:
 - For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The inventory Mapping window opens. Make sure you have selected the locations or custom field to map the inventory.



Note: If you do not see the inventory field or the field that storefront uses for inventory, click **Add Mapping** and add the field to your mapping.

6. Click **Add Operation Row**.
A new operation row displays on the window.
7. From the **Operations** list, select **Subtract**.
8. In the **Value** field, enter **5**.
9. (Optional) To add more mathematical operations, repeat steps **7** and **8**.
10. Click **Save Changes**.
11. Click **Save**.

Subtracting Backordered Quantities for Multiple Locations in NetSuite Connector

By default, NetSuite Connector does not consider backordered quantity when syncing inventory from multiple locations. For example, assume there are two syncing locations A and B. An item has 19 backordered quantity with 0 available quantity at location A, and 1 backordered with 25 available at location B. NetSuite Connector syncs 25 as the quantity for the item. You can also set NetSuite Connector to subtract the sum of backordered quantities from the sum of the available quantities and sync 5 quantity. Use the following procedure to control the behavior of subtracting backordered quantities.

To subtract backordered quantity when syncing multiple locations:

1. Log in to app.farapp.com.
2. Select the connector and the relevant account.
3. Go to Mappings > Products.
4. Perform one of the following actions:

- For marketplace connectors like Amazon or Walmart, from the **Category** list, select the category where your current inventory mapping resides.
 - For other connectors, from the **Select Sync Type** field, select **Price/Quantity Only** or **Full Product**.
5. From the product mappings list, locate the inventory mapping, and click **Special Mapping – Click to Edit**.

The inventory mapping window opens.



Note: If you do not see the inventory field or the field that storefront uses for inventory , click **Add Mapping** and add the field to your mapping.

6. Check the **Subtract Backordered Quantity At Selected Locations From Total** box.
7. Click **Save Changes**.
8. Click **Save**.

NetSuite Connector reduces the quantity synced from your locations by the sum of the backordered quantities from those locations.

Information for Troubleshooting Inventory Sync Issues in NetSuite Connector

Following are some important points for inventory and quantity fields:

- Inventory syncs to the storefront and not from the storefront. Therefore, you should never expect to see inventory syncing back to NetSuite, though the items that appear on the orders through the order sync affect inventory.
- Inventory syncs from NetSuite are not instantaneous. Though most of the connectors support real-time price and quantity sync, syncing them can take a few minutes. This time depends on other processes running in the storefront, connector, and NetSuite. If you are not using real-time price and quantity sync, then the sync runs only once in an hour, by default.
- When syncing inventory from locations in NetSuite and not to a custom field, NetSuite Connector syncs the quantity available and not the quantity on hand. Quantity on hand includes everything you have in the warehouse, including quantities already committed for orders. Available quantity is the amount of inventory in the warehouse that has not been committed to orders and is therefore the actual available quantity to sell.

For further troubleshooting steps, read the help topic [Troubleshooting Product Sync Issues](#).

Troubleshooting Product Sync Issues in NetSuite Connector

Following are some useful articles for resolving product sync issues:

- [Troubleshooting Product Sync Issues](#)
- [Troubleshooting Common Product Sync Errors in NetSuite Connector](#)
- [Troubleshooting Item Not Matching Category Rule Errors in NetSuite Connector](#)
- [Troubleshooting Missing Retrieved Products in NetSuite Connector](#)
- [Troubleshooting Wrong Values Syncing to Items in the Storefront](#)

- Troubleshooting Issues with Posting Variation Item Families in NetSuite Connector

Troubleshooting Product Sync Issues

Updating massive amounts of items in NetSuite continuously may clog your sync. This action may also cause more items to sync than the API limits of the associated endpoints can handle. When items flagged to sync receive an update to the item record, NetSuite Connector pulls the item in to sync again. Frequent mass item updates are rarely valid and often a sign of an automation that unnecessarily updates items.

If you notice significant delays in your sync, then you may have mass item updates occurring in NetSuite. To help you determine whether you have mass updates, you can check the last modified date on your items in NetSuite.

If NetSuite Connector does not update your items in the marketplace/cart during product sync, you can check the following areas.

Summary	Description
Check the sync status.	<p>Verify if your sync is active in NetSuite Connector. Select the connector and the relevant account. Then, go to Data Flows > Manage Data Syncs. If the sync is turned off, you need to toggle the sync on. Take note of the following:</p> <ul style="list-style-type: none"> ■ Only one product sync type should be active at any time on any single connector-account combination. To avoid causing sync issues, do not enable Price and Quantity sync and Full Product sync at the same time. ■ You cannot switch dynamically between two different sync types. If both Price and Quantity and Full Product syncs are turned off, turn on only the sync that you tested and configured.
Check the SKU.	<p>Go to Data Flows > Products. Use the search box to enter your SKU.</p> <ul style="list-style-type: none"> ■ If the SKU shows as pending posting, wait until the scheduled sync runs. Then, recheck the status of the SKU. ■ If the SKU shows as synced, recheck the item in the marketplace/cart. The item may have already synced while you were troubleshooting. ■ If the SKU has an error, hover over the SKU to display the error. Typically, the error message tells you what to do to fix the issue. If you do not know how to fix the error, try searching the Help Center. <p>If a specific product has an error, then that product will not sync. A large number of products with errors may also prevent other items without errors from syncing. NetSuite Connector can only sync a certain number of items during each sync, and will always retry items with error. That means NetSuite Connector could waste time and sync slots only to retry unfixed errors, which may leave other items out of sync.</p> <ul style="list-style-type: none"> ■ If the SKU shows as excluded, hover over the SKU to display the reason for exclusion and take appropriate action as needed. ■ If you do not find the SKU when you search, ensure that you are searching in the correct connector-account combination. If you still cannot find the SKU, then NetSuite Connector must have missed pulling the item in to the sync, or categorized the item incorrectly. Take note of the following: <ul style="list-style-type: none"> □ Verify if you set the item to sync in NetSuite for the relevant connector and account. Each account requires a separate flag field. For more information on how to locate and flag products to sync in NetSuite, read Flagging Existing NetSuite Items to Sync. □ When you use Amazon, Walmart, or eBay for matrix items, you must flag the entire family to sync in all cases. This step also applies to Full Product sync for shopping cart connectors, such as Shopify, BigCommerce, Magento 2. □ In NetSuite Connector, try to manually retrieve the SKU. Click the Retrieve button, which is directly below the search box.

Summary	Description
Check the list-based storefront flag field.	If you manually create a list-based storefront flag field, ensure that your list contains the exact order and values, as described in Creating a List-Based Storefront Flag Field Manually .
Check the category of your items on the Product Dashboard.	<ul style="list-style-type: none"> ■ If the Price and Quantity sync is on, then you should see the PriceQty or a similar name in the category. For instance, having ShopifyPriceQty means that NetSuite Connector properly set the items to use the Price and Quantity sync for Shopify. Amazon typically uses the AmazonInventoryPrice name, but it may also use InventoryLoader for the Price and Quantity sync. ■ If the Full Product sync is on, you should see the following: <ul style="list-style-type: none"> □ For shopping carts, you should see the name of the connector. For example, if you use Shopify Full Product sync, you should see Shopify. □ For Amazon, you should see the name of the custom template. This field contains the name specific to your account that usually includes a date. Ensure that this name is not blank nor shows Amazon, InventoryLoader, or AmazonInventoryPrice. Any other value is likely correct. □ The eBay connector is a special case. No matter which sync you turn on, the Category field should show eBayVariations or eBayFixedPrice. □ If your products are missing or do not show the correct category, make sure you have properly mapped your Category field. For more information, refer to the following sections: <ul style="list-style-type: none"> ▪ For Amazon, read Mapping Amazon Categories in NetSuite Connector. ▪ For shopping carts such as Shopify and Magento 2, read Mapping Cart Categories in NetSuite Connector.

If you verified all areas but your product is still not syncing, contact NetSuite Customer Support.

Troubleshooting Common Product Sync Errors in NetSuite Connector

If you get an error with product sync in NetSuite Connector, hover over the product row to see the error details.

This topic discusses some common errors during product sync.

Error	Description
Too many requests	This error indicates that NetSuite Connector has reached API limits from the marketplace or cart and need to wait before posting additional item updates. You need not take any action as this error usually resolves on its own the next time NetSuite Connector attempts to sync the item.
Mapping errors: All variants must have a unique combination of options. Child SKU XYZ that is already in <Storefront Name> but not in NetSuite has options [Size - 5ml], which is the same as SKU <ABC> in NetSuite	<p>Storefronts do not let two SKUs to contain the same combination of options. Therefore, the storefront generates this error when posting an item that already matches with another item in the storefront. In the example message, SKU XYZ is same as SKU ABC because their size options are same.</p> <p>This error can occur if you change the SKU name in NetSuite. For example, assume that for Shopify, you change SKU ABC to SKU XYZ in NetSuite. Shopify has no way to tell that the SKUs are same items and tries to create a new item with the same combination of options. However, this action is not permitted and Shopify generates an error. To fix this, either make sure the NetSuite SKU exactly matches the Shopify SKU or delete one SKU from Shopify.</p>

Error	Description
USER ERROR: Please specify email to send to	NetSuite gives this error when it tries to save the order record or when the same data is used to create an order manually. For more information, read the SuiteAnswers article Transactions > Manufacturing > Enter Work Order > Save > Notice: Please specify an email address to send to.

Troubleshooting Item Not Matching Category Rule Errors in NetSuite Connector

This error occurs when an item that NetSuite Connector retrieves from NetSuite is not categorized correctly.. NetSuite Connector automatically categorizes items and decides what fields to use based on the mapping set in the Category field. To fix this error, you need to verify that you correctly set up your Category field mapping.

The following are common cases encountered where this error occurs:

Situation	Action Required
You have multiple connectors or accounts using product sync but not all stock-keeping units (SKUs) are syncing to all connectors or accounts. Getting this error on those connectors or accounts where you are not syncing the SKU is expected behavior.	None
All SKUs are synced to all connectors or accounts. The error appears on most or all of your products.	Verify that you have created a Category field mapping.
All SKUs are synced to all connectors or accounts. The error appears on select items.	Verify Category field mapping for those items.

To verify the Category field mapping:

1. Log in to app.farapp.com
2. Select the connector and relevant account.
3. Go to Mappings > Products.
4. Depending on the affected connector or cart, do one of the following:

Connector or Cart	Action
Wayfair, Overstock or carts such as Shopify and Magento	See Mapping Cart Categories in NetSuite Connector .
Amazon	See Mapping Amazon Categories in NetSuite Connector .
Ebay	No Category Mapping is required. Your products are automatically categorized based on item variations.
Walmart or NewEgg	For each category, verify that the NetSuite Field ID and value accurately reflect the field and values your items have to categorize to the specific Walmart or NewEgg category.
✖ Warning: You should never categorize to the generic Walmart or NewEgg category.	

5. Check the value on the Category field mapping.
 - An empty value while using a shopping cart or Wayfair and Overstock is the correct setting.
 - If the value is not empty, ensure the affected item has that value in the designated field.

Troubleshooting Missing Retrieved Products in NetSuite Connector

When you attempt to retrieve a product using the Products page, the following occurs:

1. NetSuite Connector attempts to retrieve and store the product's details.
2. NetSuite Connector uses product mappings to categorize the product.

If the product was imported into NetSuite Connector but is not appearing in the product table, there is an issue in the second step. Follow these steps to resolve the issue in NetSuite Connector:

1. Ensure that you have set the SKU field for the storefront and account in which you are working. You can verify the SKU field mapping from the Product Mappings page (Mappings > Products) of your connector and account combination.
2. Ensure that you have added at least one mapping category for the storefront on the Product Mappings page. For the category, you must have at least the flag field and category mappings completed.
3. Ensure that your item has its flag field populated and is classified under the category in the **CATEGORY** mapping.
4. Ensure that the item does not have a duplicate, that is, no two items in NetSuite have the same SKU.
5. Refresh the page and try retrieving the product again.

If you are still unable to retrieve the product, contact NetSuite support.

Troubleshooting Wrong Values Syncing to Items in the Storefront

Following are the possible reasons for NetSuite Connector not syncing correct values in fields for items in the storefront:

- The mapping is not using the intended source field in NetSuite.
- The value in the source field in NetSuite is not what you are expecting.

To resolve the issue, check if the mappings of affected fields are set up correctly set up. For more information, see the topics about mapping under [NetSuite Connector Product Sync Management](#).

Troubleshooting Issues with Posting Variation Item Families in NetSuite Connector

Variation items represent items with various options available using a parent and child relationship. The parent item contains the common information about the item, and the child items contain the unique variations of the items. To post item families correctly, the flag fields for these families must be set appropriately.

If you see posting errors for your child items, check the following:

- In NetSuite, check that both the parent and child item's flag fields are set to **Add/Update** or **Y**. Alternatively, if you want to post the child items as standalone items, set your parent item to **Post Children as Standalone** or **C**.

- When using full product sync, the variation child SKUs and variation field values for each SKU must match the values set on existing items in the storefront. Also, the entire family must be set to sync. If any of these requirements are not met, the error message provides the details so that you can correct the issue.
- If the error messages contain errors related to particular product fields, map those product fields for that category. Note that the product fields differ by both category and storefront.

NetSuite Connector Product Sync FAQ

The following are some frequently asked questions for NetSuite Connector product sync.

Why is an item still posting when I deactivated or removed it from NetSuite?

NetSuite Connector determines whether to post an item based on the flag fields of the item, not based on whether the item is active in NetSuite. If you remove or deactivate an item from NetSuite, but still have the flag field set to **Y**, NetSuite Connector will post the item. Also, because the item is deactivated, NetSuite Connector does not reload the item. NetSuite Connector continues to post all the data for that item it last processed before deactivating or removing from NetSuite. For more information about fixing this issue, read [Handling Marking Items Inactive](#).

I deleted an item from NetSuite Connector. Why does it keep resyncing?

NetSuite Connector uses the mapped flag field to determine if you want NetSuite Connector to sync an item. Deleting an item from NetSuite Connector does not update the flag field in NetSuite. Therefore, the item continues to sync.

Why is my item showing scheduled to delete even though I have set the flag field to Add/Update Item?

To resolve this issue, check the following:

- Check the flag field ID set under Mappings > Products for the relevant connector and account combination. The field with the same field ID must be set to **Add/Update Item** in NetSuite. You usually find the field ID in NetSuite by clicking the field name. Else, check the configuration of the custom field to find the field ID.
- If you manually create the list that the flag field is referencing, make sure the list is using the exact ordered values. For more information, read [Creating a List-Based Storefront Flag Field Manually](#).
- Try deleting the item from NetSuite Connector and then retrieving the item.

I have set my Storefront Flag Field in NetSuite to remove an item. Why does the product still exists in the storefront?

The following are the potential reasons:

- You are using Price/Qty sync. This sync does not permit removing or deleting products from the storefront. Only Full Product sync permits deleting products using the storefront flag field.
- The automated sync process has not cycled fully. The automated process must first pull your changes from NetSuite and then send those changes to the storefront. This process typically takes an hour to complete.

I updated a custom list, custom record, or image in NetSuite. Why didn't my update post to the storefront?

NetSuite Connector posts updates to storefronts for items that had modifications. Updating a custom list, custom record, or image in NetSuite does not actually update the item record. Though the item record

may reference the updated value, the item itself is not updated in NetSuite. Do any of the following to get NetSuite Connector post the update:

- Modify the item record. For example, you can change the flag on the item to blank and give NetSuite Connector enough time to load the changes. Then change the flag back to **Y**. If real-time sync is enabled, you need not wait for NetSuite Connector to load the changes.
- If the storefront is eBay or any other storefront that has a dashboard in NetSuite Connector, you can force a reload operation from NetSuite Connector. For more information, read [Reloading Products in NetSuite Connector](#).

If an SKU is changed in NetSuite, can it update in the storefront?

NetSuite Connector cannot update the SKU (usually the itemID) in the storefront as part of full product sync. NetSuite Connector uses the SKU to match to the item in the storefront. If the SKU is changed in NetSuite, then NetSuite Connector cannot match to the item in the storefront.

If you must change the SKU in NetSuite, you must also manually change it in the storefront.

Is it possible to map multiple storefront products to a single product in NetSuite?

Yes, NetSuite Connector supports mapping multiple storefront products to a single product in NetSuite provided the products in the storefronts have unique SKUs. For more information, read [Handling Inventory Items with Alias SKU](#).

My product sync is disabled. Why is NetSuite Connector still importing items to sync?

In product sync, syncing SKUs to the storefront and importing products from NetSuite are separate processes. The process to load items flagged to sync into NetSuite Connector is always enabled and runs at 45 minute intervals. The process to sync those items to the storefront can be enabled or disabled.

If you are seeing items in NetSuite Connector then those items are flagged to sync in NetSuite. If you do not want the items importing into NetSuite Connector, then the flag field on the items should be set to **Ignore Item** or blank. Then, the next time NetSuite Connector pulls updated items from NetSuite, NetSuite Connector changes those items to a **Product Not Flagged to Sync** status.

Why do my products still have inventory in the marketplace or cart after disabling the inventory mapping?

When you disable an inventory mapping, NetSuite Connector stops sending a value for inventory to the marketplace or cart. The items in the marketplace or cart will have the last posted quantity prior to disabling the mapping. If your marketplace or cart tracks inventory, the inventory for those items will decrease over time in the marketplace or cart. This decrease happens if customers place orders for those products. However, the inventory level is not affected by the inventory level of the items in NetSuite.

If you do not want the inventory in the marketplace or cart to be zero for your items, try one of the following options:

- Leave the inventory mapping enabled and set the inventory level to zero on items in NetSuite. When the automatic product sync runs, NetSuite Connector posts this amount to your items in the marketplace or cart.
- Disable the inventory mapping and set your inventory levels to zero directly in the marketplace or cart.

Can I sync either quantity or price with price and quantity sync or full product sync?

You can sync either quantity or price, or both during price and quantity or full product sync. NetSuite Connector syncs only the mapped fields. Therefore, if you do not want to sync a field, do not create a mapping for that field.

If you have mapped price or quantity sync, but want to disable the sync, read [Disabling a Mapping in NetSuite Connector](#).

For information about mapping price, read [Mapping Prices in NetSuite Connector](#).

For information about mapping quantity, read [Mapping Inventories in NetSuite Connector](#).

Can I sync noninventory items with product sync?

Yes, NetSuite Connector can sync most item types with product sync.



Note: Most storefronts do not have the concept of different item types. Therefore, all items sync into the storefront as standard items. The exception is type items (family items), which, by default, sync with a matching relationship among the family members between NetSuite and the storefront.

You may typically need to sync the following item types:

- Assembly (including lot numbered and serialized items)
- Kit/Package
- Inventory (including lot numbered and serialized items)
- Item groups

The following item types can also be synced including for purchase, sale, or resale:

- Noninventory items
- Other Charge items
- Service items

NetSuite Connector can also sync additional item types. However, the success of syncing those item types depend on the data points that you want to sync and the storefront field to which you sync. To check, try syncing a test item of the type.

Can NetSuite Connector sync product data from storefront to NetSuite?

You should use NetSuite as the master record so that your data updates in NetSuite and syncs to the storefront, through NetSuite Connector. With this setup, you need not update data in multiple storefronts. Instead, update the data in NetSuite and sync to multiple storefronts.

When setting up NetSuite, if you want to migrate product data from the storefront to NetSuite, first export the data from storefront to a CSV file. Then, import the CSV file into NetSuite. For more information about CSV import process in NetSuite, read the help topic [CSV Imports Overview](#). After the data is in NetSuite, set up NetSuite Connector to automatically sync to the storefront.

Managing Connectors

In this chapter, you will learn more about the connectors provided for different storefronts and 3PLs. This chapter covers the following main sections:

- Managing the Amazon Connector in NetSuite Connector
- Managing the BigCommerce Connector in NetSuite Connector
- Managing the eBay Connector in NetSuite Connector
- Managing Magento 2 Connector in NetSuite Connector
- Managing the Shopify Connector in NetSuite Connector
- Managing the Walmart Connector in NetSuite Connector
- Managing the WooCommerce Connector in NetSuite Connector
- Managing Third-Party Logistics (3PL) Connectors in NetSuite Connector
- Managing the ShipStation Connector in NetSuite Connector
- Troubleshooting Common Connector Issues

Managing the Amazon Connector in NetSuite Connector

Using the Amazon connector, you can connect NetSuite with Amazon storefront. This section discusses the following topics about the Amazon connector:

- Amazon Seller Central vs. Vendor Central
- Supported Amazon Orders
- Default Mappings for Amazon Connector
- Amazon Category Inheritance for NetSuite Connector
- Amazon Inventory Adjustment in NetSuite Connector
- Setting the Handling Time for Amazon in NetSuite Connector
- Setting Amazon Gift Wrap Items in NetSuite Connector
- Checking Valid Variations in Amazon
- Amazon Merchant Shipping Groups in NetSuite Connector
- Delays in Amazon Order Imports
- Automatic SKU Creation For Amazon Settlement Charges
- Manually Creating Amazon Settlement SKUs in NetSuite Connector
- Resolving Amazon SKU Data Conflicts in NetSuite Connector
- Importing Amazon Settlement Reports in NetSuite Connector
- Syncing Sale Price for Amazon Items in NetSuite Connector
- Assigning a Fixed Customer to All Amazon Orders in NetSuite Connector
- Adding a New Value to the Amazon Product Type Custom Field in NetSuite
- Supporting Multiple Unified Amazon Accounts on One Connector

- Syncing Orders of Other Amazon Marketplaces in the Same Region
- Managing Amazon Fulfillments Without Tracking Numbers in NetSuite Connector
- Configuring NetSuite Connector to Use Amazon Standard Identification Number (ASIN) When Posting Products on Amazon
- Using ASIN to Identify an Amazon Product
- Amazon Tax Remittance
- Categories, Product Types, and Item Types in the Amazon Flat File Template
- Managing Fulfillment by Amazon (FBA) Items in NetSuite Connector
- Verifying that Amazon and NetSuite Connector MWS Authentication Tokens Are Matching
- Managing Amazon Settlement Sync on NetSuite Connector
- Amazon Seller Fulfilled Prime Add-On in NetSuite Connector
- Troubleshooting Amazon Connector Issues
- Amazon Connector FAQ

Amazon Seller Central vs. Vendor Central

Amazon has the following two platforms for selling products on its marketplace:

- Seller Central
- Vendor Central

NetSuite Connector connects only with the Seller Central platform. Seller Central is intended for B2C (Business to Consumer) businesses because it lets you sell directly to Amazon customers. With Vendor Central, you sell your stock directly to Amazon, who then resells your products. Vendor Central is intended for B2B (Business to Business) selling.

Seller Central is built on an API, which is a framework for connections between different software systems. Whereas, Vendor Central is built on an Electronic Data Interchange (EDI) protocol, which NetSuite Connector does not support.

Supported Amazon Orders

NetSuite Connector three types of Amazon orders. The following are the order types and how NetSuite Connector handles them through Order Sync:

- **Fulfillment by Amazon (FBA) orders**

These orders are shipped out of an Amazon warehouse, where the merchant has sent their inventory beforehand. The merchant is not involved in the fulfillment process for these orders, it is done entirely by Amazon. Amazon FBA orders are imported to NetSuite Connector after their order status has been updated to "shipped" by Amazon. They are posted to NetSuite as Cash Sales, by default, because they do not require fulfillment. Cash Sales also commit inventory immediately, so your inventory in NetSuite will be kept up-to-date.

- **Merchant Fulfilled Network (MFN) orders**

MFN orders are fulfilled by the merchant. Packing, shipping, and fulfillment of the order is completed by the merchant. MFN orders are imported as Sales Orders after they have been paid in Amazon. Typically, the inventory for MFN orders will be committed after the sales order is in **Pending Fulfillment** status. Then, the merchant will have to fulfill and ship out the order from NetSuite. NetSuite Connector will sync the tracking information from NetSuite back to Amazon.

- **Seller Fulfilled Prime (SFP) or MFN Prime orders**

SFP orders are a hybrid of the two previous order types, and allow merchants to list their products as Prime eligible, while also maintaining control of their fulfillment operations. There are several requirements to be eligible for SFP. For more information, check for seller fulfilled prime on Amazon's website. Because SFP orders are technically a subcategory of MFN orders, NetSuite Connector handles them in the same way. The orders are imported as sales orders after they have been paid in Amazon and the inventory is typically committed after the sales order is in the **Pending Fulfillment** status. Because the shipping label for SFP orders must be purchased through Amazon, Amazon already knows the tracking and other fulfillment information related to the order. Therefore, NetSuite Connector does not need to sync the fulfillment information to Amazon and instead marks the order **Complete** after syncing it to NetSuite.

Default Mappings for Amazon Connector

This section lists the default mappings for the Amazon connector product sync and order sync.

Amazon Seller Central Product Sync Default Mappings

Storefront Field	NetSuite Field
asin	custitem_fa_amz_asin
brand	manufacturer
bullet-point#	custitem_fa_amz_bullet_#
colormap	custitem_fa_amz_color_map
condition-note	custitem_fa_amz_cond_note
condition-type	custitem_fa_amz_cond_type
currency	custitem_fa_currency
Department	custitem_fa_amz_dept
description	storeDetailedDescription
fulfillment-center-id	custitem_fa_amz_fba
is-discontinued-by-manufacturer	custitem_fa_amz_discontinu
is-gift-message-available	custitem_fa_amz_gift_msg
is-giftwrap-available	custitem_fa_amz_gift_wrap
item-package-quantity	custitem_fa_amz_pkg_qty
item-type	custitem_fa_amz_item_type
leadtime-to-ship	custitem_fa_amz_leadtime
main-image-url	storeDisplayImage.internalId
manufacturer	manufacturer
mfr-part-number	mpn
model	custitem_fa_amz_model_name

Storefront Field	NetSuite Field
other-image-url#	custitem_fa_oth_img_url#
other-item-attributes#	custitem_fa_amz_other_#
product-tax-code	A_GEN_TAX (taxable)
ProductCategory	custitem_fa_amz_category
product_type	custitem_fa_amz_prod_type
restock-date	custitem_fa_amz_restock_dt
sale-end-date	custitem_fa_amz_sale_end
sale-start-date	custitem_fa_amz_sale_start
search-terms#	searchkeywords
sku	itemId
standard-product-id	custitem_fa_amz_asin
subject-content#	custitem_fa_amz_subject_#
target-audience#	custitem_fa_amz_targaud_#
title	storeDisplayName
update-delete	custitem_fa_amz_flag
used-for#	custitem_fa_amz_used_for_#
weight	weight

Amazon Seller Central Order Sync Default Mappings

Storefront Field	NetSuite Field
Address_Field_One	addr1
Address_Field_Three	addr2
Amazon_Order_ID	tranid
Buyer_Email_Address	tobeemailed
Buyer_Name	addressee
Buyer_Phone_Number	addrphone
City	city
Country_Code	zip
Country	country
Currency	currency
OrderTotal	total
Order_Date	trandate

Storefront Field	NetSuite Field
Phone_Number	phone
Postal_Code	zip
Ship_Start_Date	shipdate
billtoaddress1	addr1
billtoaddress2	addr2
billtocity	city
billtocountry	country
billtostate	state
billtozip	zip
fulfillmentstatus	orderstatus

Amazon Category Inheritance for NetSuite Connector

When you add an Amazon connector, NetSuite Connector creates a generic category called **Amazon** with a default set of mappings to facilitate product sync. This category is created because Amazon requires sellers to create separate templates to categorize their products. The Amazon category enables you to map all the common fields such as price, quantity, and SKU.

The Amazon category is used to set the common fields in custom flat files. A custom flat file combines all the deprecated category-specific templates into one. To create this file, read [Import Amazon Custom Flat File Templates to NetSuite Connector](#).

Adding the Standard Amazon Category in NetSuite Connector

To sync inventory, price, or both to existing Amazon listings, add the standard mapping category **AmazonInventoryPrice** for the Amazon connector before you map flag fields.

To add the mapping category for Price and Quantity sync:

1. Log in to app.farapp.com.
 2. Select Amazon connector and the relevant account.
 3. Go to Mappings > Products.
- The Product Mappings page appears.
4. In the **Category** list, select **Add Category**.
 5. On the Add New Amazon Category/Template popup window, click **No**.
This step ensures that updates will change only the price and quantity of existing items. New items that do not exist yet in Amazon will not be created.
 6. On the product mappings page, verify that the **AmazonInventoryPrice** option appears in the **Category** list.
 7. Click **Save**.

You can now complete your product mappings for Price and Quantity sync. For more information, read [Mapping Amazon Categories in NetSuite Connector](#).

Mapping Amazon Categories in NetSuite Connector

Before creating a category mapping for Amazon product sync, make sure you perform the following tasks:

1. Determine the type of product sync.
2. Perform one of the following tasks:
3. Determine the field ID of Amazon flag field in NetSuite.

Determining the Field ID of Amazon Flag Field in NetSuite

To complete the category mapping, note the field ID of the flag field that you will use for Amazon product sync in NetSuite.

To determine the field ID of Amazon flag field in NetSuite:

1. Go to Lists > Accounting > Items.
 2. Click **Edit** on an inventory item.
 3. Locate the flag field that will be used for the Amazon storefront flag field:
 - For flag field automatically created using the NetSuite Connector SuiteApp, the flag field will by default be located in the **Amazon** subtab.
 - If the flag field is manually created or if the automatically created flag field's placement is changed, search the field in the item record.
 4. Click the flag field name.
A Field Help popup window opens. If the **Show Internal IDs** preference is enabled in NetSuite, the field ID displays in the popup window.
 5. Note the field ID.
The ID starts with the string **custitem**.
- For more information, read [NetSuite Connector Communication with Internal IDs and Field IDs](#).



Note: Each account in the Amazon connector has a separate flag field and a flag field ID. Make sure you note the field ID for each account's flag field.

Creating the Flag Field Mapping and Category Mapping

Create flag field mapping and category mapping for your Amazon connector.

To create the flag field mapping and category mapping:

1. Log in to app.farapp.com.
2. Select Amazon connector and the relevant account.
3. Go to Mappings > Products.
4. From the **Category** list, select one of the following categories:
 - For price and quantity sync, select **AmazonInventoryPrice**.

- For full product sync, select the category created using a custom template.

Note: Do not select the generic **Amazon** category.

The mapped fields for the category display in the Product Mappings page.

5. In the **NetSuite Field ID** column, enter the field ID for the flag field.
6. Click **Add Mapping**.
The Add Amazon Field Mapping window opens.
7. From the **Required** list, select **CATEGORY**.
8. Click **Add Mapping**.
9. Click **Close**.
10. Locate the newly added **CATEGORY** mapping and click the **Special Mapping – Click to Edit**.
The CATEGORY Mapping window opens.
11. In the **NetSuite Field ID** field, enter the field ID of the flag field.
Make sure the field ID entered in this field matches the field ID entered in step 5 and the **Value** field is blank.
12. Click **Save**.
13. (Optional) If the Reload Items window opens, click **Yes, Reload my items**.
14. Click **Save**.

Amazon Inventory Adjustment in NetSuite Connector

Amazon Inventory Adjustment sync adjusts the inventory of items in NetSuite at a specific location to reflect the accurate inventory for these items in Amazon warehouses. This sync is useful for Fulfillment by Amazon (FBA) items to update the inventory levels in NetSuite to reflect availability in the Amazon warehouses. This sync enables you to keep track of your FBA and Merchant Fulfilled Network (MFN) inventory in NetSuite.

When you enable this feature, NetSuite Connector checks the latest inventory reported by Amazon in its warehouses and adjusts NetSuite inventory to match Amazon's inventory. NetSuite Connector does not replace the inventory value in NetSuite, it only makes an inventory adjustment transaction in NetSuite.

For example, if NetSuite shows an inventory of 100 but Amazon shows an inventory of 99, NetSuite Connector makes an inventory adjustment of -1. NetSuite Connector does not check the transactions, inventory events, or other things that affect the inventory.

Note: Inventory adjustment sync currently does not support bin logic.

The inventory adjustment feature is an add-on feature that requires additional integration and ongoing fee. To add inventory adjustment sync to your account, contact NetSuite Customer Support with the following information:

- NetSuite internal ID (IID) of the location where you want to set up Amazon inventory adjustments. If you have multiple Amazon accounts and need to set up multiple Amazon inventory syncs, provide the internal ID of all the locations for each account.
- Email address where you will receive adjustment sync error reports.
- An item SKU that can be used to test the inventory adjustment sync.

Setting the Handling Time for Amazon in NetSuite Connector

Amazon uses the **leadtime-to-ship** field to set the handling time. Use the following procedure to map this field.

To map the handling time field:

1. Log in to app.farapp.com.
2. Select the Amazon connector and the relevant account.
3. Go to Mappings > Products.
4. From the **Category** list, select **Amazon**.
5. Click **Add Mapping**.
The Add Amazon Field Mapping popup window opens.
6. From the **Standard** list, select **leadtime-to-ship**.
7. Click **Add Mapping**.
The mapping is added to the Product Mappings page.
8. Click **Close**.
9. In the newly added mapping row, perform one of the following actions:
 - To send the same value for every item:
 1. Set the **Mapping Type** column to **Use Default**.
 2. Enter the value you want to send in the **Default Value** field.
 3. If you have multiple Amazon accounts, from the dropdown list in the **Account** column, select the appropriate option.
 - To map the value you send from a NetSuite field:
 1. Set the **Mapping Type** column to **Item Field**.
 2. In the **NetSuite Field ID** column, enter the field ID.
 3. In the **Default Value** field, enter the default value to send when the mapped field is blank.
10. Click **Save**.
11. If you are prompted to reload your items, click **Yes, Reload My Items**.
If you have many items, the process may take some time to complete.

Setting Amazon Gift Wrap Items in NetSuite Connector

Amazon stores gift wrapping charges as a separate item in order data. However, NetSuite has no inherent gift wrap charge feature. To get around this, you may map the Amazon gift wrap charges to an item you designate in NetSuite. The gift wrap revenue will then be accounted for in both Amazon and NetSuite order totals.

To map the Amazon gift wrap item:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.

3. Go to Settings > Orders.
4. Click **Advanced Options**.
5. On the **Amazon Gift Wrap Item** field, enter the NetSuite internal ID of the gift wrap charge item.
6. Scroll down to the bottom of the page then click **Save**.

To find the internal ID in NetSuite, read [Finding the Internal ID of Records](#).

Checking Valid Variations in Amazon

You can check which variations such as size, color, and weight are valid parent and child variations in the Amazon category.

To check valid variations in Amazon:

1. Go to the Amazon website and look for the flat file templates.
 2. Open the flat file template for the desired category in Amazon.
 3. In the **Valid Values** worksheet, check the **Variation Theme** column.
- The column lists all the valid variations.

Amazon Merchant Shipping Groups in NetSuite Connector

To set up Amazon merchant shipping groups, first set up the shipping templates for the groups in Amazon. To map the merchant shipping group for an item, create a mapping for the **merchant-shipping-group-name** field in the Amazon product mappings.

Delays in Amazon Order Imports

Delays to import FBA (Fulfillment by Amazon) orders are normal. The orders are already complete when NetSuite Connector retrieves the orders and Amazon controls inventory on FBA orders. Therefore, NetSuite Connector syncs FBA orders by default every five hours.

A common reason for a delay in importing MFN (Merchant Fulfilled Network) orders is that the order is pending in Amazon for an unusually long time. NetSuite Connector does not import orders that are in pending state.

Automatic SKU Creation For Amazon Settlement Charges

NetSuite Connector supports creating the Amazon Settlement charge SKUs under the following conditions:

- You have a single subsidiary or no subsidiaries.
- You are using the **itemId** field as your SKU field.

NetSuite Connector provides an automated process for creating the SKUs.

For information on using the automated process, see [Importing Amazon Settlement Reports in NetSuite Connector](#).

If you need to create the SKUs manually, see [Manually Creating Amazon Settlement SKUs in NetSuite Connector](#).

Manually Creating Amazon Settlement SKUs in NetSuite Connector



Important: If you have ran the automatic creation of SKUs previously, use the manual creation process only for the SKUs that were not created automatically. You may get errors if you add an SKU that already exists in NetSuite.

NetSuite Connector represents fees and credits that Amazon reports through the settlement report as other charge for sale SKUs in cash sales and cash refunds. Therefore, if you do not use the automatic SKU creation process, you may have to create many SKUs in NetSuite.

If you do not use the automated SKU creation process, you can use one of the following options:

- CSV import
- Create individual SKUs

Using CSV Import

For creating SKUs using CSV import, note the following:

- Create an other charge for sale item for each fee or credit name listed in the section [Creating Individual SKUs](#).
- Assign the correct value to your designated SKU fields.
- If you are using multiple subsidiaries, assign a subsidiary to the item that matches the subsidiary on your Amazon orders.
- Set the items to not be fulfillable.
- Assign an income account to the item.
- Assign a nontaxable tax schedule to the item.

Creating Individual SKUs

Create the following SKUs as other charge for sale items. Make sure that the **Can be Fulfilled** box is not checked and the items are set as nontaxable.



Note: Do not add trailing spaces in the SKU names.

- A2ZGuaranteeRecovery
- Adjustment
- Amazon Capital Services
- Amazon Shipping Charges
- Amazon Shipping Reimbursement
- BalanceAdjustment
- BuyerRecharge
- ChargeBackRecovery

- COD
- COD Tax
- CODFee
- Commingling VAT
- Commission
- Commission Adjustment
- Commission IGST
- CommissionCorrection
- COMPENSATED_CLAWBACK
- Cost of Advertising
- CouponRedemptionFee
- CRETURN_WRONG_ITEM
- CS_ERROR_ITEMS
- CS_ERROR_NON_ITEMIZED
- Current Reserve Amount
- Customer Return
- Customer Service Issue
- Customs Reimbursement
- Damaged:Inbound
- Damaged:Outbound
- Damaged:Warehouse
- Discount Not Applied
- DisposalComplete
- Early Reviewer Program fee
- ExportCharge
- FBA Inventory Storage Fee
- FBA Pick & Pack Fee
- FBA transportation fee
- FBACustomerReturnPerOrderFee
- FBACustomerReturnPerUnitFee
- FBACustomerReturnWeightBasedFee
- FBAInboundDefectFee_BAGGING
- FBAInboundDefectFee_LABEL_ISSUE
- FBAInboundDefectFee_SUFFOCATION_STICKER
- FBAInboundDefectFee_TAPING
- FBAInboundTransportationFee
- FBAInboundTransportationProgramFee
- FBAPerOrderFulfillmentFee
- FBAPerUnitFulfillmentFee
- FBAWeightBasedFee

- Fee correction
- Fixed closing fee IGST
- FixedClosingFee
- FREE_REPLACEMENT_REFUND_ITEMS
- GiftWrap
- GiftwrapChargeback
- GiftWrapTax
- Goodwill
- Imaging Services Fee
- INBOUND_CARRIER_DAMAGE
- INCORRECT_FEES_ITEMS
- INCORRECT_FEES_NON_ITEMIZED
- Inventory Placement Service Fee
- Inventory Storage Overage Fee
- Item Not Received
- LightningDealFee
- LightningDealFee Special
- Lost:Inbound
- Lost:Outbound
- Lost:Warehouse
- LowValueGoods-Principal
- LowValueGoods-Shipping
- LowValueGoodsTax-Principal
- LowValueGoodsTax-Shipping
- Manual Processing Fee
- Manual Processing Fee Reimbursement
- MarketplaceFacilitatorTax-Giftwrap
- MarketplaceFacilitatorTax-Other
- MarketplaceFacilitatorTax-Principal
- MarketplaceFacilitatorTax-RestockingFee
- MarketplaceFacilitatorTax-Shipping
- MarketplaceFacilitatorVAT-Principal
- MarketplaceFacilitatorVAT-Shipping
- MiscAdjustment
- MISSING_FROM_INBOUND
- MISSING_FROM_INBOUND_CLAWBACK
- MULTICHANNEL_ORDER_LOST
- NonSubscriptionFeeAdj
- Paid Services Fee
- PAYMENT_RETRACTION_ITEMS

- POS_NOSKU
- PREPFEE_REFUND
- Previous Reserve Amount Balance
- Principal_Promotion
- Premium Experience Fee
- Premium Placement Service Fee
- RE_EVALUATION
- RefundCommission
- RemovalComplete
- REMOVAL_ORDER_DAMAGED
- REMOVAL_ORDER_LOST
- RestockingFee
- Return Adjustment
- ReturnShipping
- REVERSAL_REIMBURSEMENT
- SAFE-T reimbursement
- SalesTaxServiceFee
- Shipping
- Shipping label purchase
- Shipping label purchase for return
- Shipping Reimb.
- Shipping tax
- ShippingChargeback
- ShippingHB
- ShippingServicesRefund
- ShippingTax
- Storage Fee
- StorageRenewalBilling
- Subscription Fee
- SubscriptionFeeCorrection
- Tax
- TaxDiscount
- TCS-CGST
- TCS-IGST
- Unplanned Service Fee - Barcode label missing
- VariableClosingFee
- WarehousePrep
- WAREHOUSE_DAMAGE
- WAREHOUSE_DAMAGE_EXCEPTION
- WAREHOUSE_LOST
- WAREHOUSE_LOST_MANUAL

Resolving Amazon SKU Data Conflicts in NetSuite Connector

When you encounter an error that SKU data conflicts with the Amazon catalog, this means that Amazon matched the item data against one of the items in its catalog, but some of the data conflicts with the catalog data so Amazon is rejecting it.

To resolve this, on the item, you must create a field for the Amazon Standard Identification Number (ASIN) that Amazon has assigned. NetSuite Connector can then use this field to match it with the correct item in the Amazon catalog. This will eliminate most, but not all of the conflict errors. There are some data conflicts that Amazon will not accept even if the ASIN is provided.

For more information on using the ASIN and how to locate it on existing items in Amazon, see [Using ASIN to Identify an Amazon Product](#).

Importing Amazon Settlement Reports in NetSuite Connector

NetSuite Connector can import your fees, refunds, and other transactions from your Amazon settlement reports. These reports are generated depending on your report frequency settings in your Amazon account.

NetSuite Connector retrieves the following specific reports from your Amazon account:

- **Flat File Settlement Report** – retrieved through the GET_V2_SETTLEMENT_REPORT_DATA_FLAT_FILE
- **Flat File V2 Settlement Report** – retrieved through the GET_V2_SETTLEMENT_REPORT_DATA_XML
- **Flat File V2 Settlement Report** – retrieved through the GET_V2_SETTLEMENT_REPORT_DATA_FLAT_FILE_V2



Important: If you do not see V2 reports enabled in your account contact Amazon support.

NetSuite Connector posts Amazon's settlement adjustments to your NetSuite account as three types of adjustments:

1. **Order Modifications** – Are fees that Amazon charges you per order. An example would be **FBA Per Order Fulfillment Fee**. These order modifications are posted as additional fees or credits to the billed record of your orders. These fees are represented as SKUs to reflect the adjustments from Amazon.
2. **Non-Order Modifications** – Are fees or reimbursements that apply to a whole period rather than a specific order. An example would be **FBA storage fees** or **Reimbursements for FBA stock that Amazon lost or damaged**. Non-order Modifications are posted as the sum of the adjustments from Amazon that do not apply to specific order.
 - **Cash Refund** transactions are created if the resulting total shows that you owe Amazon money.
 - **Cash Sales** transactions are created if the resulting sum shows that Amazon owes you money.
3. **Refunds** – Are orders that are refunded to the customer. These are posted as **Refund Authorizations**, **Cash Refunds**, or **Credit Memos** based on the original order. Cash Refunds or Credit Memos are only created if you prefer to manually create Return Authorizations and Item Receipts.

Syncing your settlement reports with NetSuite is a paid and optional feature.



Note: NetSuite Connector does not support combining different fees into a single line item whose quantity and total are the sum of the fee for the entire report. NetSuite Connector supports a single transaction for the entire report but fees or credits are not combined.

Setting Up Order Fees

NetSuite Connector creates an **Other Charge For Sale** item type in NetSuite to represent fees and credits. The following describes the steps to setup Order Fees:

To Setup Other Charge for Sale Item Type:

1. Go to app.farapp.com.
2. Select the Amazon connector in the navigation menu.
3. Select an account.
4. Go to Settings > Other Transactions.
5. Click the **Settlement** tab.
6. Click **Create Settlement Items**.
7. Select an Income Account. If you want to use different accounts for all the items, you will need to update those items in NetSuite. For more information, see the help topic [Account Information on Items](#).

For OneWorld Accounts, the Other Charge for Sale item type will be created under the subsidiary with an ID of 1.



Important: You will create invalid settlement items if any of the following conditions are met:

- You are using a subsidiary that does not have the subsidiary ID 1. To confirm your subsidiary ID, see the help topic [Editing Subsidiary Records](#).
- Your item records are not using the **Item Name/Number** field as your SKU field.

Importing Transactions from Settlement Reports

The following are the transactions that are imported from your Settlement Reports to NetSuite:

- **Refunds:** Refunds are imported from the settlement report. These are imported against the corresponding orders in NetSuite by default. You can also import one summary transaction for all the refunds combined.

If the imported settlement report contains transactions covering two months, this will split the summary transaction into two.

- **Service Fees:** Service fees that are not associated with specific transactions in NetSuite are imported as new summary transactions:

- If the summary is positive, it is posted as a Cash Sale.
- If the summary is negative, it is posted as a Cash Refund.

These summary transactions include line items for all the service fees in the settlement report.

If the imported settlement report contains transactions covering two months, this will split the summary transaction into two.

- **Other Transactions:** Other transactions in the settlement report that are not associated with specific transactions in NetSuite are imported as new summary transactions:

- If the summary is positive, it is posted as a Cash Sale.
- If the summary is negative, it is posted as a Cash Refund.

These summary transactions include line items for all the service fees in the settlement report.

If the imported settlement report contains transactions covering two months, this will split the summary transaction into two.

Setting the Amazon Settlement Report Cutoff Date

NetSuite Connector enables you to control how far back in time to look for settlement reports from Amazon.

To set the Amazon Settlement Report cutoff date:

1. Log In to NetSuite Connector.
2. Select the Amazon connector and the desired account.
3. From the **Account** dropdown, select an account.
4. Go to Settings > Other Transactions.
5. Select the **Settlement** tab.
6. Select **Advanced Options**.
7. Enter the number of days in the **Days Back To Search For Settlement Reports** field.

This is the number of days back NetSuite Connector will look for settlement reports from the current date. In standard seller accounts, Amazon generates a settlement report every 2 weeks. Setting a minimum of 20 days (which is the default number of days) ensures that all settlement reports for your accounting period are captured.

8. Click **Save**.

Identifying Settlement Fees Posted As Order Modifications

NetSuite Connector includes any fee that has an order ID associated with it as an order modification. In the Amazon settlement report, if you see an order ID for any fee, such orders are included in order modifications. This condition also applies to fees labelled as **other-transaction**. However, in some cases, such as a cash sale deposit, NetSuite Connector creates a new cash sale or cash refund record as appropriate. For such cases, NetSuite Connector does not place the fee or credit into the original billed record.

Retrieving Amazon Settlement Reports into NetSuite Connector Manually

NetSuite Connector lets you manually retrieve settlement reports into NetSuite Connector with or without sync enabled. This feature helps testing single reports when the sync is off or when there is an issue in the report being generated automatically.

You can retrieve reports for the last 30 days. Reports older than 30 days may be limited by Amazon.

Note: Before proceeding, make sure you know either the ID of the order that appears in the settlement report that you want to import, or the report ID. Note that report ID is not the same as the settlement ID.

The following procedure uses report ID to retrieve the settlement report.

To retrieve Amazon settlement reports into NetSuite Connector manually:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Data Flows > Settlements.
4. Click **Retrieve**.
The Retrieve Settlement into FarApp window appears.
5. In the **Settlement Report ID** field, enter the report ID.
If you are using an order ID, enter the order ID in the **Order ID** field.
When retrieval is complete, a success message appears.
6. Click **Close**.
The settlement appears in the list of reports.

If settlement sync is enabled, the report syncs when the sync runs. If sync is off or if you want to post the report immediately, click the pencil icon in the **Action** column and select **Post**.

Amazon Refund Sync Frequency

Refunds are imported as part of the settlement reports. However, refunds can only be posted when the settlement report is released, which is typically every two weeks for Amazon.

Reconciling the Data NetSuite Connector Imported to NetSuite from Amazon Settlement Reports

NetSuite Connector copies Amazon settlement reports data to NetSuite using a functionality similar to the one that is used to copy order data into NetSuite. The settlement sync data is pulled through Amazon's API and the corresponding fees and credits are applied to the orders in NetSuite using the settlement sync process. You can safely assume that the order settlement data on the Amazon report matches the order settlement data placed in NetSuite by NetSuite Connector.

 **Note:** Ensuring the accuracy of the fees Amazon charges or calculating the cumulative totals of such fees is outside of scope of NetSuite Connector.

Settlement Report from Amazon Seller Central Has Not Synced Yet

Report availability in Amazon Seller Central and report availability through the API are not necessarily the same, though usually they are within a day of each other. Typically, settlement sync also runs one time per day only. There will still be a delay of several days between report availability in Amazon Seller Central and report syncing by NetSuite Connector.

However, if the delay is more than 2-3 days before your settlement report appears in NetSuite Connector, you should check the following:

- Ensure your Amazon Settlement Sync is enabled by setting it On in NetSuite Connector from Data Flows > Manage Data Syncs, on the Amazon connector.
- Occasionally, Amazon will indicate that a report is available, but then change the status back to processing. To be sure that the report is really available, download the report directly from Amazon Seller Central.

Syncing Sale Price for Amazon Items in NetSuite Connector

Price and Quantity sync type only syncs the item price and quantity fields of your items in Amazon. If you need to sync the sale price for your items, you need to use the Full Product sync.

Assigning a Fixed Customer to All Amazon Orders in NetSuite Connector

You can configure NetSuite Connector to send all orders to NetSuite using the same customer. This approach is useful in the following situations:

- Orders are from Amazon's FBA service, where Amazon has already fulfilled the order and captured the payment on your behalf
- You do not want to have a new customer record created for each new customer on an order

Configuring a Fixed Customer in NetSuite

You can either use a new or existing customer as a fixed customer. Make sure the customer is taxable so that both taxable and nontaxable orders can be assigned to the customer.

After setting up the customer in NetSuite, note the customer's internal ID. To locate the internal ID, read [Finding the Internal ID of Records](#).

Configuring a Fixed Customer in NetSuite Connector for All Amazon Orders

After you configure the fixed customer in NetSuite, configure the fixed customer in NetSuite Connector for all Amazon orders.

To configure a fixed customer in NetSuite Connector for all Amazon orders:

1. Log in to app.farapp.com.
 2. Select the Amazon connector and relevant account.
 3. Go to Settings > Orders.
- The Order Settings page opens.
4. Enter the customer ID in one of the following fields:
 - **Post All Orders Against Customer** – Enter the customer ID in this field to post all the Amazon orders to that customer.
 - **Post FBA (Fulfilled by Amazon) Orders Against Customer** – Enter the customer ID in this field to post only the Amazon FBA orders to that customer.
 5. Click **Save**.

Adding a New Value to the Amazon Product Type Custom Field in NetSuite

One of fields that you set when setting fields for Amazon product sync using the NetSuite Connector SuiteApp is the **Amazon Product Type** field. If the dropdown list in this field does not show the value you want, use the following procedure to add a new value to the list.

To add a new Amazon product type:

1. Go to Customization > Lists, Records, & Fields > Records Types.
2. From the Record Types list page, locate **Amazon Product Type** row and select **New Record** in that row.
The Amazon Product Type page opens.
3. In the **Name** field, enter the new product type value.
4. Click **Save**.

Supporting Multiple Unified Amazon Accounts on One Connector

NetSuite Connector supports Unified Amazon Accounts. NetSuite Connector can be configured to receive orders from different countries in a single NetSuite Connector.



Note: Unified Accounts only connect marketplaces that are in the same region.

While all of your orders and settlements would come in on a single connector, you would have to set up separate connectors if you also want to use Product Sync and/or Price/Qty Sync for the other countries on the connector. For example, if you set up an Amazon.com connector to also take in orders from Amazon.ca, you would need to set up a separate Amazon.ca connector if you will use Product Sync for Amazon.ca. This is a limitation of Amazon Application Programming Interface (API), where the API does not have a field where NetSuite Connector can specify which country an item should be listed in. This makes making separate connections necessary.

To set up NetSuite Connector with a Unified Amazon Account, you will enter the Amazon account access information.

Syncing Orders of Other Amazon Marketplaces in the Same Region

For a unified Amazon account, NetSuite Connector can sync orders from other Amazon marketplaces in the same region without requiring a separate connector for each marketplace. For example, assume that you have orders in Amazon.com, Amazon.ca, and Amazon.mx, and your Amazon account is a unified account. You can then sync orders from all three marketplaces on a single Amazon connector.



Note: This process does not work for product sync. To sync products to more than one marketplace, you need a separate Amazon connector for each marketplace.

To sync orders from other Amazon marketplaces in the same region:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Settings > Orders.
4. Click **Advanced Options**.
5. From the **Additional marketplaces to import orders for using this account** list, select the additional marketplaces from which you want to sync orders.
6. Click **Save**.

For more information about unified North America accounts, read https://sellercentral.amazon.com/gp/help/help.html?itemID=G201642980&language=en_US&ref=efph_G201642980_cont_200399470

Managing Amazon Fulfillments Without Tracking Numbers in NetSuite Connector

If an Amazon order in NetSuite is fulfilled but does not contain a tracking number, NetSuite Connector can prevent syncing the fulfillment back to Amazon. To prevent this syncing, enable the **Prevent From Posting NetSuite Fulfillments Without a Tracking Number To Amazon/Default** setting in NetSuite Connector. You can access this setting by going to Amazon > Settings > Fulfillments in NetSuite Connector.

When this setting is enabled, if NetSuite Connector detects an Amazon order in NetSuite that has been fulfilled but does not contain a tracking number, then the fulfillment will not be synced back to Amazon. The order will remain in an **Order Posted, Waiting For Shipment** status in NetSuite Connector.

Each time NetSuite Connector checks NetSuite for fulfillments that need to be synced to Amazon, it will check the order again to see if a tracking number has been added. If NetSuite Connector detects the tracking number, it will sync the fulfillment and mark the order as Complete; otherwise, the order will remain in NetSuite Connector until the next check is run where the process is repeated.

You can enable this setting on other marketplaces or carts like Shopify or eBay, and it works in the same way.

Configuring NetSuite Connector to Use Amazon Standard Identification Number (ASIN) When Posting Products on Amazon

When posting products on Amazon, the leading practice is to identify the products using Amazon Standard Identification Number (ASIN). For more information, read [Using ASIN to Identify an Amazon Product](#).

Before proceeding, make sure you have created a custom field on your products in NetSuite and populated the field with the ASIN of the item. For more information, read [Adding NetSuite Custom Fields](#). When you install the NetSuite Connector SuiteApp, the custom ASIN field is automatically created. You should only populate the ASIN values in the field. If you do not have any Amazon product mappings, the SuiteApp creates the default mappings for ASIN and Universal Product Codes (UPC) fields.

To configure NetSuite Connector to use the ASIN field:

1. Log in to app.farapp.com.
 2. Select the Amazon connector and then select the relevant account.
 3. Go to Mappings > Products.
- The Product Mappings page opens.
4. From the **Category** dropdown list, select **Amazon**.
 5. Locate the **standard-product-id** mapping field and click **Special Mapping – Click to Edit** next to the mapping field.
- The standard-product-id Mapping popup window opens.
6. In the **ASIN Field (Default)** field, enter the field ID that you use to store the ASIN.
 7. In the **UPC Field (Default)** field, enter the field ID that you use to store the UPC.

If you keep the **ASIN Field (Default)** field blank, NetSuite Connector uses the data provided in the **UPC Field (Default)** field.

8. Click **Save Changes**.

Using ASIN to Identify an Amazon Product

The Amazon Standard Identification Number (ASIN) is a 10-digit alphanumeric identifier that is assigned by Amazon to a product upon initial creation in their marketplace. NetSuite Connector uses a product's ASIN to match with an existing listing on Amazon.

Since the ASIN is Amazon's product identifier, if NetSuite Connector sends Amazon the ASIN when syncing products, usually, Amazon will not reject listings if they contain product data that is inconsistent with the data already on the item. Instead, Amazon will simply ignore the data discrepancy on most fields and accept the listing. There are a few fields that Amazon will reject for data that conflicts with its own, and of course, your seller listing specific data (price, condition, quantity, or others) does not need to match the existing data on Amazon. For this reason, you should add ASINs to your items in NetSuite.

The product ASIN is indicated in the URL for your product, it is the series of numbers immediately after the product's name and "dp". It is also indicated in the product information section at the bottom of your product's listing.

There are online software tools that can lookup the ASIN assigned to a product. Typically, they let you to import your product identifiers (UPC, ISBN, and others) and then identify the ASIN assigned to that identifier. This can help you save time if you have many items in NetSuite with fields where the ASIN should be entered.

Amazon Tax Remittance

As required by law, Amazon collects and remits sales tax on all taxable marketplace sales shipped to certain states. Consult Amazon for information on the localities in which Amazon remits tax.

You can set NetSuite Connector to omit tax from orders where Amazon has remitted the tax on your behalf. For more information, read [Omitting Remitted Tax in NetSuite Connector](#).

Categories, Product Types, and Item Types in the Amazon Flat File Template

NetSuite Connector syncs items to Amazon using flat file templates. For more information about importing a template file to NetSuite Connector, read the SuiteAnswers article [Import Amazon Custom Flat File Templates to NetSuite Connector](#).

Amazon uses the following attributes from the template file to classify an item:

- **Category** – You can use several Amazon categories during product sync. Some categories can inherit product attributes from other templates.
For information, read [Import Amazon Custom Flat File Templates to NetSuite Connector](#)
- **Product Type** – The product type is a required field for some templates. The value of this field helps further classify an item into categories. To check if product type is included in your template, in the **Data Definitions** worksheet of the template, check for **product_type** or **feed_product_type** field in the **Local Label Name** column. If the field is included and if the rightmost cell of the row has the **Required** value, you must populate the product type. Amazon permits only valid values for this field. The **Accepted Values** column indicates where you can find the valid values. Usually, the **Product Type**

column in the **Valid Values** worksheet contains the list of valid values that you can populate in your list in NetSuite.

- **Item Type** – The item type is an optional field and not included in all templates. If included, the field lets Amazon classify the item in the Amazon browse tree so that a customer finds the item by browsing through Amazon categories. To check if item type is included in your template, in the **Data Definitions** worksheet of the template, check for **item_type** or **item_type_keyword** field in the **Local Label Name** column. If the field is included and if the rightmost cell of the row has the **Required** value, you must populate the item type. The **Accepted Values** column indicates where you can find the valid values. Usually, these values are available in the Amazon's Browser Tree Guide (BTG) for the corresponding flat file template. You can find and download the BTG files from the Amazon seller central website help.

After you have the BTG corresponding to your template, look in the main worksheet that has the **Node ID**, **Node Path**, and **Query** columns. The values at the right of the text **item_type_keyword:** in the **Query** column are the valid values for the **item type** field. Do not include the text **item_type_keyword:** when adding these values to your list in NetSuite.

Managing Fulfillment by Amazon (FBA) Items in NetSuite Connector

To post Fulfillment by Amazon (FBA) items correctly, you need to map various fields between NetSuite and NetSuite Connector. If the mapping is done incorrectly, items will revert to merchant-fulfilled items when the product data is posted.

You can customize NetSuite records to add a custom field for FBA products. For example, add a box that you can check to indicate if an item is FBA. After creating the custom field, you can set up the product mappings in NetSuite Connector.

To map fields for FBA products:

1. Log in to app.farapp.com.
 2. Select Amazon connector and the relevant account.
 3. Go to Mappings > Products.
- The Product Mappings page appears.
4. Map the **fulfillment-center-id** field.
 - a. Under the **Amazon Field** column, locate the fulfillment-center-id.

If this field is not on the list, you must add the field mapping first. For more information, read [Creating the Flag Field Mapping and Category Mapping](#).

 - b. In the same row, locate and click **Translation Mapping - Click to Edit** listed under the NetSuite Field ID column.

The fulfillment-center-id Translation Mapping window appears.

 - c. In the **NetSuite Field**, enter the NetSuite field ID.

This field will mark the fulfillment center for the items.

 - d. In the **Amazon Value** field, select the appropriate option.
- The following table lists the values that correspond to each fulfillment center.

Amazon Value	Guidelines
blank (item is merchant-fulfilled)	Select this option for any the following: ■ Fulfilled By Merchant (FBM) items

Amazon Value	Guidelines
	<ul style="list-style-type: none"> ■ Merchant Fulfilled Network (MFN) items ■ Merchant-fulfilled items
AMAZON_NA	Select this option for FBA items sold in North America.
AMAZON_EU	Select this option for FBA items sold in Europe.
AMAZON_JP	Select this option for FBA items sold in Japan.
AMAZON_CN	Select this option for FBA items sold in China.
AMAZON_IN	Select this option for FBA items sold in India.

- e. Click **Save Changes**.
5. Map the **quantity** field.
To configure the quantity mapping, contact NetSuite support. This mapping needs to send a blank value to Amazon for FBA items. Amazon determines quantities based on inventory items in their warehouses.
6. Map the **leadtime-to-ship** field.
To configure the leadtime-to-ship mapping, contact NetSuite support. This mapping needs to send a blank value to Amazon for FBA items.

For more information, read the following articles:

- [Posting FBA Orders in NetSuite Connector](#)
- [Using FBA as a 3PL Provider in NetSuite Connector](#)

Posting FBA Orders in NetSuite Connector

Posting Amazon FBA orders involves one required setting and two mappings that you should complete.

Creating an Amazon Fixed Customer

Amazon does not provide complete customer information for FBA orders. Therefore, to sync these orders to NetSuite, create a fixed customer that all FBA orders will be associated with in NetSuite. The name and available address information for the order still appears on the order itself.

To create an Amazon Fixed Customer:

1. Create a customer and note its internal ID.
If you configure NetSuite as recommended by NetSuite Connector, you should see the internal ID of the customer in your list of customers.
Else, refer to the topic [Viewing Internal IDs in NetSuite](#).
To locate the internal ID for an object, you can also check the topic [Finding the Internal ID of Records](#).
2. Log in to app.farapp.com.
3. Select the connector and the relevant account.
4. Go to Settings > Orders.
The Order Settings page opens.
5. In the **Post FBA (Fulfilled by Amazon) Orders Against Customer** field, enter the internal ID of the customer.

6. Click **Save**.

Mapping a Location for FBA Orders

You should create an inventory location for FBA items. When you replenish FBA items to Amazon, move them into that inventory location. After posting FBA orders to NetSuite, NetSuite Connector posts that inventory location for the orders, decreasing the FBA inventory.

Mapping FBA orders to a location is done like any other order location mapping. For more information, read [Mapping Order Location in NetSuite Connector](#). The only difference is that with Amazon connector, you get an additional column to select whether the mapping applies to MFN, FBA, or both. In this case, you should set that column to **FBA**.

Mapping a Shipment Method for FBA Orders

Use the following procedure to assign a shipment method to FBA orders.

To assign a shipment method to FBA orders:

1. Perform one of the following actions:
 - Identify an existing shipping item.
 - Create a new shipping item for using with FBA orders.
2. Note the internal ID of the shipping item.
To locate the internal ID for an object, read [Finding the Internal ID of Records](#).
3. Log in to app.farapp.com.
4. Select the connector and the relevant account.
5. Go to Mappings > Orders.
6. Click the **Shipping** tab.
The shipping items are retrieved from NetSuite.
7. From the **Default Shipment Method to Use for FBA Orders** list, select the FBA shipment method.
8. Click **Save**.

Using FBA as a 3PL Provider in NetSuite Connector

NetSuite Connector enables you to set up Amazon FBA as a 3PL so that you can send orders from other storefronts to Amazon for fulfillment. This process is also called multi-channel fulfillment. To set up Amazon FBA as a 3PL, contact NetSuite support, with answers to the following questions:

- Do you want all orders to post to the Amazon FBA 3PL?
- If all orders should not be posted as Amazon FBA 3PL:
 - What criteria should be used to determine the orders that will go to Amazon FBA?
 - Will just orders at a specific location go to Amazon FBA?
 - Will there be a custom field set up on the order that can be used to determine the orders that will go to Amazon FBA?

Syncing FBA Inbound Shipping in NetSuite Connector

The Amazon FBA Inbound Shipping Sync can be used by companies that offer products fulfilled by Amazon through the Amazon connector in NetSuite Connector. This sync enables our users to send a

shipment of items to the Amazon FBA Warehouse for FBA orders. The following are the two types of this sync and the processes involved.

Full Sync for Sending Stock to Amazon

1. In NetSuite, a client creates a transfer order (TO), pending fulfillment.
2. NetSuite Connector creates an inbound-shipping plan in Amazon.
3. Amazon indicates to NetSuite Connector where the client should ship the package according to Amazon.
4. NetSuite Connector gets the shipping plan in return and creates a pick-ticket per inbound-shipment in the plan.
5. The client adds the appropriate number of packages with no carton content details (weight of the Item Fulfillment is marked and Fulfillment is set as "Packed"). Relevant information should be entered on the Packages subtab of the fulfillment record. Records must not be added to the Fulfillment Packages custom record because this is how NetSuite Connector will determine the number of packages the client wants.
6. NetSuite Connector detects what package the client has and creates a blank custom record for the client to enter the quantities of the items that are in the packages. The items for the package should already be there.
7. The client ensures that the packages are filled in properly - fills in the package dimensions, makes necessary adjustments in their cartons if needed, and sets fulfillment as Shipped.
8. NetSuite Connector relays the information to Amazon, gets a quote based on this information, prints the label, and syncs it back to the client's NetSuite account.
9. After the client marks the transfer order as Fulfilled in NetSuite, NetSuite Connector notifies Amazon that it is shipped.
10. NetSuite Connector monitors Amazon at regular intervals for item receipts. When NetSuite Connector detects the receipts, it syncs those back to NetSuite.

Receipt-only Sync

NetSuite Connector also has a Receipt-only sync for Amazon FBA Inbound Shipping. In this scenario, you can generate the Amazon inbound-shipping plan, add the Amazon Shipment ID to a corresponding transfer order in NetSuite, and ship the product to Amazon. These actions are done outside of NetSuite Connector, which just checks Amazon for new item receipts and then posts those to NetSuite based on the mapped Shipment ID.

Setting Up Amazon FBA Inbound Shipping

To set up an Amazon FBA Inbound Shipping sync, contact NetSuite Customer Support. Indicate if you want a Full Sync or Receipt-only Sync and the NetSuite Connector support team will assist you in setting it up.

You should prepare the following requirements and answer questions to provide information about the sync you want.

Full Sync

- Will you allow NetSuite Connector to create custom records that will show up as a custom sublist of your fulfillment record in NetSuite, which will be used to input carton contents for each package?
- Will you be using case-packed shipments? If so, you must create a custom transaction column field of type integer that will be available on transfer orders and purchase orders. This will indicate for a given line item on a transfer order or purchase orders the number of the item per case. After setting up this field, provide the field ID to the NetSuite Connector support team.

- Will you use Amazon-partnered carriers, as these are cheaper?
- What is the originating address for the shipments, or where will you be sending the shipments from? This is usually the address for your warehouse and is a fixed value that the NetSuite Connector support team can map for you. If you are shipping from more than one warehouse, you should provide the names of the warehouses and their addresses.
- You should create a folder in the NetSuite File Cabinet where NetSuite Connector can store label files. Then, you should provide the folder ID to the NetSuite Connector support team.
- You should create a transaction body field of type document, that will be available on fulfillments and purchase orders, where the label can be stored. Then, provide the field ID to the NetSuite Connector support team.
- You should provide the field where SKUs are stored.
- You should create a transfer order to use for testing and provide the internal ID to the NetSuite Connector support team.

Receipt-only Sync

You should provide a Transfer Order ID in the Pending Receipt state. It should have items on it that were shipped to Amazon and has the Amazon Shipment ID entered in the custom field you specified. The Transfer Order will be used to post the Item Receipts from Amazon.

NetSuite Connector will need the field ID of the custom field on your Transfer Order form, where you will enter the Amazon Shipment ID.

Mapping Transaction Dates for FBA Orders in NetSuite Connector

By default, NetSuite Connector uses the purchase date that Amazon reports to Fulfillment By Amazon (FBA) orders as the order date. This date is then mapped as the transaction date on the order when the order is imported to NetSuite. Amazon also makes two additional dates, ship start date and ship end date available in the order data that NetSuite Connector retrieves. These two dates are always the same for Amazon FBA orders. NetSuite Connector only reports the dates provided by Amazon and cannot verify if the two dates accurately reflect the shipping date of the order. For accounting purposes, you can map the ship start date and ship end date as the transaction date on the order in NetSuite using the following procedure.



Important: Sometimes, Amazon reports the ship start date as 1/1/1980 on some FBA orders for unknown reasons. If you map that date as your NetSuite transaction date, NetSuite rejects your order and does not let you post the order. For this reason, you should not map the ship dates as your transaction date and leave the default transaction date logic in place.

To map ship start date or ship end date as transaction date:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Mappings > Orders.
4. Click the **Order** tab.
5. Click **Add Mapping**.
6. From the **NetSuite Field** dropdown list, select **Date**.
7. Click **Add Mapping**.

A confirmation message appears indicating the field is added to the end of your list of mappings.

8. Click **Close**.

A new row for the **Date** field displays at the end of the mappings list.

9. In the newly added row in the mappings list, from the dropdown list in the **Mapping Type** column, select **Order Header**.
10. From the dropdown list in the **Amazon Field/Fixed Value** column, either select **Ship Start Date** or **Ship End Date** as both fields contain same values in the orders.
11. From the dropdown list in the **FBA/MFN** column, select **FBA – Fulfilled By Amazon**.
12. Click **Save**.

A banner message appears indicating the mappings are updated.

Mapping the Restock Field on Item Receipts Created on Refunds

NetSuite Connector creates return authorizations and item receipts during refund transactions for Amazon FBA orders with settlement sync. Additionally old customers may be using connectors or syncs that also create return authorizations and item receipts with refund transactions. However, that functionality is no longer available. You may want to map the restock field on refund item receipts to false to prevent NetSuite from automatically restocking the refunded item. The following procedure provides steps to creating this mapping in refunds created through Amazon Settlement Sync. The process is the same for other marketplaces and carts with minor differences noted in the procedure.

To map the restock field on refund item receipts:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Mappings > Settlements.
(Optional) For other connectors, go to Mappings > Other Transactions.
4. Click the **Settlement Item** tab.
(Optional) For other connectors, click the **Transaction Item** tab.
5. Click **Add Mapping**.
The Add NetSuite Mapping popup window appears.
6. From the Transaction Type list, select **Item Receipt**.
7. From the NetSuite Field list, select **Restock**.
8. Click **Add Mapping**.
A success message appears on the popup window.
9. Click **Close**.
10. Locate your new mapping in the settlement or transaction item mappings list. Under the NetSuite Field/Fixed Value column, select **Unchecked** to tell NetSuite not to restock items.
11. Click **Save**.

You can create more advanced conditional mappings by selecting Logic under the Mapping Type column in step 10. Note that conditional mappings require that the marketplace or cart include a piece of data on the refund receipt to base your logic conditions on. Amazon does not provide such data although other connectors might.

For more information on custom order mappings, read [Setting up Custom Order Mappings in NetSuite Connector](#).

Converting FBA Items to MFN Items in NetSuite Connector

You can use NetSuite Connector to convert your items from FBA to MFN during product sync. NetSuite Connector checks the items that you are syncing to see if it is marked as FBA, and syncs it to Amazon. If you installed the NetSuite Connector SuiteApp, the **Amazon FBA** box is visible on the **Amazon** subtab of your item record. Clearing this box resyncs the item and converts it to MFN.

Verifying that Amazon and NetSuite Connector MWS Authentication Tokens Are Matching

Before verifying the Amazon MWS authentication token, you must have installed the NetSuite Connector application into your Amazon account and have generated the credentials to connect NetSuite Connector.

The following procedure verifies that the MWS authentication token you have entered in NetSuite Connector is the same as the one assigned to the NetSuite Connector application in Amazon.

To verify that Amazon and NetSuite Connector MWS Authentication Tokens Are Matching:

1. Get the MWS authentication token value in Amazon. Refer to relevant documentation on Amazon Seller Central.
 2. After obtaining MWS authentication token value in Amazon, log in to app.farapp.com.
 3. On the left menu, select your Amazon connector and the account that you want to verify the authentication token.
 4. Go to Settings > Credentials.
- The Amazon MWS token value is displayed under Credential Settings, in the **MWS Auth Token** field. Verify that the token value matches the value you got from Amazon. If it does not match, change it to the value from Amazon
5. Click **Save and Test Connection**.

Managing Amazon Settlement Sync on NetSuite Connector

This section covers the following topics about Amazon Settlement Sync:

- [Amazon Settlement Sync Dashboard in NetSuite Connector](#)
- [Amazon Settlement Sync Settings in NetSuite Connector](#)
- [Field Mapping for Settlement Sync in NetSuite Connector](#)
- [Mapping the Account Field for Amazon Settlement Sync in NetSuite Connector](#)

Amazon Settlement Sync Dashboard in NetSuite Connector

The Amazon Settlement Sync dashboard displays the following:

- Settlement transactions that are synced from your settlement reports
- Errors associated with the sync
- History of synced reports
- Progress of previously synced reports

Accessing the Amazon Settlement Sync Dashboard

Access the settlement sync dashboard from the **Settlements** tab in the Data Flows section.

To access the Amazon Settlement Sync dashboard:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Data Flows> Settlements.

Dashboard Layout

The Amazon Settlement Sync dashboard contains the following columns:

Column Name	Description
Check box	When using bulk actions, check this box to run the bulk action. Checking the box in the column header selects all objects on that page.
Account	Displays the account of the settlement report. This account should always be the same as the account selected for the Amazon connector.
Report ID	Displays the Amazon assigned ID assigned to each report.
Settlement ID	Displays the settlement ID. In a report, each row has a settlement ID. Usually, a single report contains only one settlement ID.
Status	Displays the report status. Status can be: <ul style="list-style-type: none"> ■ Complete – Indicates the report is completely synced ■ Error posting to NetSuite – Displays when NetSuite Connector encounters an error when posting to NetSuite. The error status shows the percentage of report synced. The percentage updates after the sync completes the run. NetSuite Connector skips errors and continues processing. NetSuite Connector also retries the settlement items with errors on the next sync run a limited number of times, before automatically canceling the posting. ■ Posting in Progress – Displays only when you manually post the report. ■ In FarApp (Automatic Sync Disabled) – Displays only when a report is pulled into NetSuite Connector but the sync is not active.
Available Date	Displays the date Amazon made the report available through the API. This date may not be the same date when the report was brought into NetSuite Connector or began processing. Factors like timing of availability on a certain day versus the time the sync runs, and existing settlements affect the retrieval and processing.
Action	The pencil icon lets you perform an action on the settlement report in that row. Hovering the pointer on the pencil icon, opens the following action options: <ul style="list-style-type: none"> ■ Post – Posts the selected report. This action will not run when the scheduled settlement sync is running, because the posting is already in progress. ■ Cancel – Cancels all future postings of the report but does not delete the report from NetSuite Connector. Cancellation takes effect in the next scheduled sync run and does not cancel the postings that are in progress. After selecting Cancel, the label changes to Uncancel. ■ Show Mappings – Opens the Settlement Mappings window that lets you view the mapping for a specific order or for the non-order modifications for a report. The popup window has the following settings: <ul style="list-style-type: none"> □ Show Order Adjustments – Displays the order mappings for the order ID entered in the Amazon Order ID field.

Column Name	Description
	<ul style="list-style-type: none"> □ Show Non-Order Adjustments – Displays the mappings for the non-order modifications for the report. ■ Delete from FarApp – Removes the settlement report from NetSuite Connector. However, this option does not remove any settlement data that synced to NetSuite before removal. ■ View Details – Lets you view the various adjustments of the selected report. Clicking View Details opens the View Settlement windows with the following tabs: <ul style="list-style-type: none"> □ Order Fees – Displays fees and credits tied to an order. For example, commission of a n order fee. □ Refunds – Displays the fees and credits that are associated with a refund. You will see a commission fee on refunds also, because Amazon refunds the commission. □ Inventory Adjustments – These are not commonly found on the reports but reflect report changes to your FBA inventory due to settlement. <p>Note: You cannot view non-order modification fees and credits on the dashboard.</p>

View Details Window

The View Details window contains the following:

Setting	Description
Search field	Used to search a transaction in the report. Enter the transaction ID of the order or refund that you want to find and click the adjacent magnifying glass icon.
Columns	The report provides the following columns: <ul style="list-style-type: none"> ■ Marketplace – The Amazon site from where the transaction is originated. For example, US orders show Amazon.com and Canada orders show Amazon.ca. ■ Fulfilled By – Displays whether the transaction was on the Amazon Fulfilled Network (AFN or FBA), or the Merchant Fulfilled Network (MFN). ■ Period Ending – The ending date of settlement period represented by the report. ■ Transaction ID – The order ID associated with the transaction. ■ Transaction Date – The date on which the transaction occurred. ■ Total – The total of fees and credits associated with the transaction. ■ SKUs – Clicking the magnifying glass opens a popup windows displaying fees and credits for the settlement and the SKU for which the fees and credits are applied.
Page Number	Lets you select the page you wish to view and shows the page number that you are viewing. Also shows the total number of pages and buttons to move to the first (<<) or last (>>) page.
Transactions per page	Lets you select the number of transactions that appear per page. The larger the number, the longer the page takes to load.

Amazon Settlement Sync Settings in NetSuite Connector

The Amazon settlement sync has some required and optional settings. You should keep the default settings, unless you need to change specific settings.

Accessing the Amazon Settlement Sync Settings

Access the Amazon settlement sync settings from the Settlement Settings page.

To access Amazon settlement sync settings:

1. Log in to app.farapp.com.
 2. Select Amazon connector and the relevant account.
 3. Go to Settings > Other Transactions.
 4. Click the **Settlement** tab.
- The Settlement Settings page opens displaying the basic settings.
5. (Optional) To view the advanced options, click **Advanced Options**.

Following are the basic settings:

Setting	Description
NetSuite SKU Field	Enter the field ID for settlement sync. This field should match the SKU field used in your order.
Customer Internal ID	<p>Some fees and credits on the settlement report are not connected to a particular order (for example, FBA warehouse fees). NetSuite Connector creates a new cash sale or cash refund to represent the fees. Enter the internal ID of the customer that you want to associate with the new cash sale or refund records created by NetSuite Connector.</p> <p>You can set a different customer from the customer used for FBA orders. In that case, you can find the transactions created by the settlement sync by checking the settlement customer's transactions.</p>
Create Settlement Items	<p>Click this button to create items in NetSuite that represent fees and credits on the settlement reports.</p> <p>You should use this setting only during setup.</p>

Following are the advanced options:

Note: You should not change the advanced options unless you have a compelling justification and understanding of the expected behavior.

Setting	Description
Days Back to Search for Settlement Reports	Sets the past number of days NetSuite Connector searches for settlement reports.
Merge All Refunds on Report Into a Single Summary Refund	Check this box to post all refunds in a single summary refund rather than posting the refunds against each order. This setting does not compile the line items. Rather, the setting takes individual refunded line item that would have existed on its own refund and places it into a single refund transaction record.
Merge All Order Fees Into a Single Transaction	Check this box to post all order fees and credits into a single summary transaction rather than posting the feed and credits in individual order. This setting does not compile the line items. Rather, the setting takes individual fee or credit line item that would have existed in the individual order and places it into a single transaction record.
When Posting Summary Transactions, Split Them Into Two Transactions If They Span Two Months	If checked, you can use this setting along with the merge settings to split the summary records so that they do not cross months.
Post Refunds as Credit Memos	Normally, refunds are created as cash refunds or refund authorizations. However, if you check this box, refunds are posted as credit memos in NetSuite.

Field Mapping for Settlement Sync in NetSuite Connector

Mapping a field for settlement sync is similar to mapping a field for order sync. The only difference is that settlement sync can have multiple transaction types, and each type requires a separate mapping for your field.

The following section explains mapping a location field for Amazon and provides a brief explanation of the different settlement transaction types. The same procedure can be used for any other fixed mapping field by substituting the location field with the field that you want to map.

Mapping the Location Field for Amazon Settlement Sync in NetSuite Connector

You may have forms configured to require the location field when the associated transaction records are created. For such forms, if NetSuite Connector does not send the location field value for a transaction, NetSuite rejects the posting with an error when syncing the settlements. To know which location should be used for a transaction type, you must create a mapping in NetSuite Connector. Before proceeding to create the mapping for location field, note the following prerequisites:

- The location field can be set at both order header level as well as line item level on your forms. If you are not sure where the location field is required, you can map both the header and line item level locations.
- Cash Sale, cash refund, return authorization, and item receipt are the most common transaction types for settlement sync. You will have to create a location mapping for all transactions whose forms require a location field. If you do not map location, the error in NetSuite Connector indicates the transaction that was attempted in settlement sync.
- Order modifications can also apply to cash sale and cash refund records. Therefore, whenever you create a cash sale or cash refund location mapping, you should also create the same mapping for the order modification transaction type.

The following procedure provides steps for creating a mapping for an order header level location field on the cash sale transactions created by settlement sync.

To create a location mapping for settlement sync:

1. Log in to app.farapp.com.
 2. Select Amazon connector and then select the relevant account.
 3. Go to Mappings > Settlements.
 4. Click the **Settlements** tab.
- To map a line item level location, click the **Settlements Items** tab.
5. Click **Add Mapping**.
- The Add NetSuite Mapping popup window opens.
6. From the **Transaction Type** dropdown list, select the transaction type for which you are creating the mapping. In this procedure, select **Cash Sale**.
 7. From the **NetSuite Field** dropdown list, select **Location**.
 8. Click **Add Mapping** and click **Close..**
- The new mapping appears in the Settlement Mappings page. The **NetSuite Fixed/Field Value** column for the new mapping is a dropdown list.
9. If the field in the **NetSuite Fixed/Field Value** column for the new mapping is not a dropdown list, click **Reload NetSuite Lists**.

10. Wait for the mapping to reload.
11. From the dropdown list, select the location that you want to use for your transaction type.
12. Click **Save**.

Mapping the Account Field for Amazon Settlement Sync in NetSuite Connector

NetSuite Connector does not map an account on your Amazon settlement transactions by default. NetSuite will usually assign a default account value or generate an error if no default is configured. The following procedure provides steps to mapping specific accounts for any given transaction type NetSuite Connector posts to with settlement sync.

To create an account mapping for settlement sync:

1. Log in to app.farapp.com.
 2. Select Amazon connector and then select the relevant account.
 3. Go to Mappings > Settlements.
 4. On the **Settlement** tab, click **Add Mapping**.
- The Add NetSuite Mapping popup window opens.
5. From the **Transaction Type** dropdown list, select the transaction type for which you are creating the mapping.



Note: The Order Modification type applies when an existing order record is changed. All other types apply only when a new record of that type is created by settlement sync.

6. From the **NetSuite Field** dropdown list, select **Account**.
 7. Click **Add Mapping**.
- A success message appears on the popup window.
8. Click **Close**.
- The new mapping should appear in the Settlement Mappings page. The **NetSuite Fixed/Field Value** column for the new mapping will either be a text field or a dropdown list.
9. In the **NetSuite Fixed/Field Value** column, enter the internal ID of the account you want to map to the transaction type or select it if the column shows a dropdown list.
 10. Click **Save**.

Amazon Seller Fulfilled Prime Add-On in NetSuite Connector

Amazon Seller Fulfilled Prime (SFP) is a program that allows qualified sellers to get the Prime badge on their products and to deliver directly to Prime subscribers using their own shipping operation.

NetSuite Connector does not require any additional setup to process Amazon SFP orders. SFP orders sync using the same NetSuite Connector process that Merchant Fulfilled Network (MFN) orders do. NetSuite Connector can automate the process of buying shipping labels from Amazon and bringing them into NetSuite for printing on SFP orders. To run this automation, you must set up the SFP add-on in NetSuite Connector. This section provides you an overview of the process and setup of the add-on.



Note: This add-on requires an additional integration fee and ongoing fee.

Workflow for Using Amazon SFP Add-On

As a leading practice, NetSuite Connector purchases the cheapest label that still meets the criteria of the order. The workflow for this feature is as follows:

1. NetSuite Connector pulls a Seller Fulfilled Prime order from Amazon.
2. NetSuite Connector prepares to purchase a label from Amazon, including getting a list of available services based on shipping locations and costs.
3. NetSuite Connector purchases a label from Amazon using the shipping services ID.
NetSuite Connector receives the results of the shipment creation, which includes shipment ID, tracking number, and shipment label.
4. NetSuite Connector processes the resulting shipment and syncs the label to a custom field on the sales order.
5. After NetSuite Connector completes steps **1 to 4**, you must perform the following actions in NetSuite:
 - a. Open the sales order.
 - b. The shipping label is stored in the folder you specified in the File Cabinet. Print the label.
 - c. Ship the item.
 - d. Mark the order fulfilled.

The process varies depending on whether the shipment label is printed in Amazon or in NetSuite. In the workflow described, the shipment label is printed in NetSuite, requiring the user to print the label, ship the item, and mark the order fulfilled. If the shipping label is printed in Amazon, the order is marked completed/shipped in Amazon. There is no need to sync fulfillment data back to Amazon as the fulfillment information is already in Amazon. The sales order then needs to be fulfilled in NetSuite using the Amazon shipping label to ship the item from the client's warehouse.

Troubleshooting Amazon Connector Issues

This section covers troubleshooting of the following common Amazon connector issues:

- [Troubleshooting Amazon Order Import Issues in NetSuite Connector](#)
- [Troubleshooting Missing Field Settlement Error](#)
- [Troubleshooting Pending Status Issues for Products Posted to Amazon](#)
- [Troubleshooting Incomplete Customer Information From Amazon MFN Orders](#)

Troubleshooting Amazon Order Import Issues in NetSuite Connector

For general order import issues in NetSuite Connector, read [Troubleshooting Order Import Issues in NetSuite Connector](#). Additionally, there are following cases specific to Amazon:

- The order is still in **Pending** status. By default, NetSuite Connector does not import orders with a **Pending** status.
- The order is an FBA order and the sync has not yet run. By default, FBA orders only sync every five hours.

Troubleshooting Missing Field Settlement Error

If your transaction form in NetSuite has a required field that is not mapped in NetSuite Connector, you get a missing field error during settlement sync. You can resolve the error in one of the ways:

- By mapping the missing field
- By adjusting the transaction form in NetSuite

Mapping the Missing Field

Map the missing field in the Settlement Mappings page.

To map the missing field:

1. Log in to app.farapp.com.
2. Select Amazon connector and then select the relevant account.
3. Go to Mappings > Settlements.
4. Perform one of the following actions:
 - If the missing field is at item level, click the **Settlement Item** tab.
 - If the missing field applies to the entire transaction form, click the **Settlement** tab.
5. Click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
6. From the **Transaction Type** list, select the transaction that you are trying to post.



Note: If the missing field is part of an order modification, meaning NetSuite Connector is posting adjustments to an order that is already posted to NetSuite, select **Order Modification**. Otherwise, select the appropriate form from the **Transaction Type** list.

For the same missing field on multiple transaction forms in NetSuite such as cash sale and cash refund, create a separate mapping for each transaction form.

7. From the **NetSuite Field** list, select the NetSuite field that you want to map.
8. Click **Add Mapping**.
9. Click **Close**.
10. Locate the new field mapping and from the **NetSuite Field/Fixed Value** column, either select or enter the field that you want NetSuite Connector to post.
11. Click **Save**.

Changing Transaction Form in NetSuite

You can fix the missing fields error by clearing the **Mandatory** box for the field in the transaction form in which you are getting the error.

To fix the missing fields error by changing the transaction form:

1. Go to Customization > Forms > Transaction Forms.
2. Locate your transaction form and click **Customize**.
3. Click the **Screen Fields** subtab.
4. Locate the missing field and clear the **Mandatory** box next to the field.
5. Click **Save**.

Troubleshooting Pending Status Issues for Products Posted to Amazon

If you post products using a NetSuite Connector Amazon connector, you may notice that the status of some product listings is **Product posted to Amazon, result pending**. This status means that Amazon received the posting but has not confirmed successful processing, so the result is pending. It is possible that after Amazon issues a result, the posting may be rejected and the product will fail to update in Amazon. Typically, it only takes a few minutes for Amazon to confirm if the posting was successful. But it takes a few hours for that status to clear in NetSuite Connector after the update goes through. The default frequency for a product sync to Amazon is every 60 minutes.

Amazon limits the number of API calls that NetSuite Connector can make to it and checking for Amazon's response is counted against this limit. For this reason, NetSuite Connector only check for a response from Amazon a couple of times a day until the status of the item is updated in NetSuite Connector. This does not mean that an update is still being processed after a few hours, rather, it is just the status in NetSuite Connector that may be delayed.

Troubleshooting Incomplete Customer Information From Amazon MFN Orders

Amazon normally sends complete customer data for all of the Merchant Fulfilled Network (MFN) orders to you to fulfill. If you are receiving incomplete or no customer data for your MFN orders you might have authorized a different developer ID for your NetSuite Connector.

After you have authorized your developer ID, you should now be receiving customer data for your orders.



Note: Authorizing a new developer ID will not sync or pull orders already present in NetSuite Connector. To force a sync:

1. Delete the orders from NetSuite Connector. Make sure to take note of the order numbers before deleting the orders from NetSuite Connector.
2. Manually retrieve the deleted orders. For more information, read [Importing Orders from the Storefront in NetSuite Connector](#).

Amazon Connector FAQ

Review the questions and answers below to learn more about the Amazon connector.

How does NetSuite Connector handle Amazon settlement reimbursements?

NetSuite Connector posts reimbursements as cash sales. If Amazon owes you money for the settlement, NetSuite Connector posts it as a cash sale.

If I sync orders from multiple regions, do I need a different MWS token for each Amazon region?

In NetSuite Connector, you must have a separate Amazon connector for each region that you intend to sync. Also, each connector must have its own set of credentials.

However, if you have a unified account in Amazon, NetSuite Connector lets you sync orders for marketplaces in the same region using a single connector. For example, US, Canada, and Mexico for the North American region.

If you want to use product sync for any marketplace in the region other than the primary marketplace, you need a separate connector for that marketplace. You cannot use a unified connector for this case.

[Does NetSuite Connector support LTL \(Less Than Load, or relatively small\) and small parcel?](#)

Yes, FarApp does support LTL and small parcel type of shipments. You'll provide us with a field that indicates the shipment name/type during setup.

[Does NetSuite Connector support syncing purchase orders in NetSuite, so that a vendor can ship stock to Amazon on behalf of the client?](#)

Yes, this works similarly to the transfer order functionality except that purchase orders aren't fulfilled in NetSuite.

[Which carriers does NetSuite Connector support for FBA Inbound Shipping?](#)

NetSuite Connector supports Amazon's preferred carriers and well as using your own carrier.

[Can you include kit items on transfer orders?](#)

NetSuite does not allow you to include kit items on transfer orders. This does not work for kits because cannot inventory the master item. However, assembly items can work as assemblies and have an inventory value unless you disassemble them.

[How will NetSuite Connector know where inventory is, what is in your warehouse, what has been already shipped to Amazon, and what is still on the way?](#)

When you created the inventory transfer (either a transfer order or purchase order), the inventory was still in your warehouse. Once the inventory transfer is shipped, it takes the inventory out of the available quantity in your warehouse but does not add it to the Amazon warehouse yet. Once the receipt is posted against the inventory transfer, the inventory is made available in your Amazon FBA warehouse location in NetSuite. When the transfer is shipped and the receipt is posted, NetSuite will show that inventory in In Transit status. It is still reflected on the books as an asset, but it will not be available at any location. When looking at a specific item you have shipped, NetSuite will show the quantity available at each location as well as that quantity that is in transit. You can view the open transfer or purchase order in the relevant orders screen. In-transit inventory should also be considered like your reorder-point, so NetSuite should not generate further purchase orders just because the inventory is not in a particular location yet.

[What happens when Amazon over- or under-receives inventory that you sent?](#)

When Amazon under-receives inventory, NetSuite Connector posts an item receipt to NetSuite for the lower amount reported by Amazon. When Amazon over-receives, NetSuite Connector tries to sync the quantity that Amazon reported, which causes NetSuite to generate an error. In this scenario, the customer will get an error, which should be clarified with Amazon.

[Can NetSuite Connector sync partial receipts?](#)

If the **useItemCostAsTransferCost** field on your transfer order is set to false and there are multiple line items on the transfer order, NetSuite Connector can sync item receipts to partially receive the transfer order. However, you cannot partially receive a line item.

Why is pricing information not associated with pending Amazon orders?

Amazon pending orders do not contain complete order information when they are posted to NetSuite.

How are Merchant Fulfilled Network (MFN) and Fulfillment By Amazon (FBA) accounts handled differently?

Following are the differences between MFN and FBA account handling:

- Listing items and/or inventory: NetSuite Connector can list MFN and FBA items under a single Amazon account or an account can be dedicated to just MFN or just FBA items. Listing FBA items is very similar to listing MFN items, except that NetSuite Connector does not sync inventory or fulfillment latency, and toggles the fulfillment center ID to indicate it's an FBA item.
- Importing orders:
 - MFN orders are imported as sales orders once they've been paid in Amazon.
 - FBA orders are imported once they have been shipped by Amazon. The orders are imported as cash sales since they don't require fulfillment. Cash sales also commit inventory immediately.
- FBA Inventory:
 - (NetSuite specific) You should create an Amazon location in NetSuite and let FarApp know the location. FarApp will import cash sales against this location so that the inventory in this location in NetSuite matches the inventory in Amazon's warehouse.
 - NetSuite Connector can also perform inventory adjustments if necessary using inventory changes in Amazon. This is an additional sync for which additional charges may apply. For more information, contact NetSuite Customer Support.
- What happens when an order contains both MFN and FBA items?
 - In cases like this, Amazon will automatically split the order so that MFN items appear together on one order and FBA items will appear together on a second order. This occurs prior to syncing the order to NetSuite Connector. After Amazon splits the order, NetSuite Connector will then import both orders separately so that they are designated MFN and FBA accordingly.
- Does NetSuite Connector handle my inbound shipments to Amazon for FBA?
 - Yes, NetSuite Connector provides support for this as an add-on feature for which additional charges may apply. By default, this feature will function as follows:
 - You create a transfer order to replicate the shipping plan you create in Amazon.
 - You place the shipment ID in a designated field on the transfer order.
 - You create an item fulfillment for the transfer order to represent the shipment to Amazon.
 - NetSuite Connector checks if Amazon has received the items. When Amazon reports received amounts, NetSuite Connector creates item receipt records for those items.

What happens when an order contains both MFN and FBA items?

In cases like this, Amazon will automatically split the order so that MFN items appear together on one order and FBA items will appear together on a second order. This occurs prior to syncing the order to NetSuite Connector. After Amazon splits the order, NetSuite Connector will then import both orders separately so that they are designated MFN and FBA accordingly.

Does NetSuite Connector handle inbound shipments to Amazon for FBA?

Yes, NetSuite Connector provides support for this as an add-on feature. By default, this will function as follows:

- You will create a transfer order to indicate what items you'll be sending to Amazon. FarApp will let Amazon know what items are being shipped.

- Amazon will indicate where to ship the items. NetSuite Connector will sync this information back so that you can create your shipping labels.
- Once you ship your items, NetSuite Connector will sync the shipping details to Amazon.
- Once Amazon receives the items, NetSuite Connector will sync those details back to the transfer order in your system.

[When an Amazon order has multiple tracking numbers, why is only a single tracking number sent to Amazon?](#)

Sometimes, clients will create multiple shipments for a single item on an Amazon order in NetSuite. There are a few reasons for this, but the most common is when the item in question is a kit item that is actually composed of other items. In these cases, clients may have a fulfillment, shipment, and tracking number for each component item of the kit. However, Amazon will only accept a single tracking number for a single item in the order. This is not a NetSuite Connector restriction and any additional tracking numbers that are sent to Amazon are simply ignored.

[Which field in the marketplace or cart does the field in NetSuite Connector product mappings correspond to?](#)

Data fields will often have two names or identifications. One is used for low level processes like sending and receiving data through the API, and the other is used when displaying the data to the user. For example, a custom field in NetSuite will have an ID field and Description. When viewing that custom field on NetSuite forms, the description is a better identifier than the field ID.

The field name displayed in NetSuite Connector Product Mappings is the name of the field that the marketplace or cart accepts data in, but that name may not always be the same as the name displayed for products in your marketplace or cart. For example, the field **Item Description** in Shopify needs to be sent to a field named **body_html** through the API.

NetSuite Connector provides a tooltip for each field that will either give you the name of the field in the marketplace or cart or a short description. The tooltip displays when you hover the mouse over the marketplace or cart field name on the Product Mappings page.

Amazon Fulfilment By Amazon (FBA) FAQ

Review the questions and answers below to learn more about fulfillment by Amazon and NetSuite Connector.

[Can NetSuite Connector map the shipping date on Amazon FBA orders?](#)

No. The shipping date on Amazon orders is not available through API so NetSuite Connector cannot map the shipping date to NetSuite.

[Can NetSuite Connector post FBA inventory to NetSuite?](#)

Yes. For more information, read [Amazon Inventory Adjustment in NetSuite Connector](#).

[Can I post my Amazon FBA orders as sales orders?](#)

No. Amazon fulfills Amazon FBA orders, so fulfillment is not necessary for such orders. Cash sales in NetSuite do not require fulfillment. Therefore, NetSuite Connector posts the Amazon FBA orders as cash sales instead of sales orders.

[Why are names and addresses missing from FBA orders?](#)

For security reasons, Amazon does not provide the buyer's name and address for FBA orders when those orders are retrieved through their API. If Amazon does not provide the data, NetSuite Connector cannot

send that information to NetSuite. Since Amazon is fulfilling the orders on your behalf, complete customer information is not necessary for the order. The FBA orders should be posted to a fixed customer in NetSuite. For more information, see [Managing Fulfillment by Amazon \(FBA\) Items in NetSuite Connector](#).

[Why does NetSuite Connector post FBA orders as cash sales?](#)

FBA orders are fulfilled by Amazon using inventory for the various products that you have already shipped to Amazon to store and fulfill orders on your behalf. Because FBA orders do not require you to directly fulfill them, they can be treated as already fulfilled. By default, NetSuite Connector will post these orders to NetSuite as cash sales, because unlike sales orders, cash sales do not require fulfillment.

[I am using Amazon FBA as a 3PL to ship my non-Amazon, like ecommerce or other orders. Does NetSuite Connector handle this?](#)

Yes, NetSuite Connector supports this feature but additional charges may apply. For more information, contact NetSuite Customer Support. Your non-Amazon orders will first sync to NetSuite. NetSuite Connector retrieves them from your system and syncs them to Amazon FBA for shipping. Once they are shipped, NetSuite Connector syncs the shipping details back to your system. From there, if the orders originated in a marketplace or cart connected to your system through NetSuite Connector, the shipping details will sync to the marketplace or cart.

NetSuite Connector can filter which orders are synced to Amazon MCF (Multi-Channel Fulfillment) for fulfillment. Contact NetSuite Customer Support and send the criteria you want to use to filter the orders. For example, a location or a setting, and support will set it up.

[Does inventory in a transfer state for FBA shows as available in NetSuite? Can we sync the quantity in reserve instead of the quantity available?](#)

No. The in-transit NetSuite Connector inventory is not made available to NetSuite Connector by Amazon. Amazon does not consider the in-transit quantity as available FBA stock until that transfer has completed. NetSuite Connector is given the quantity value by Amazon, and the value passed is the available quantity. NetSuite Connector cannot map the reserved quantity because Amazon does not consider that amount available. That amount is usually in transit between Amazon warehouses and can take some time to become available. When Amazon considers that quantity as available and if using Amazon FBA inventory adjustment sync, NetSuite Connector can sync the quantity at that point. For more information, see this Amazon community help document where other sellers discuss the concept: <https://sellercentral.amazon.com/forums/t/fba-inventory-quantity-available-vs-reserved/359398>

Amazon Settlement Sync FAQ

The following are some frequently asked questions about Amazon settlement sync.

[How do I set up Amazon Settlement Sync for NetSuite Connector?](#)

For setup and charges for Amazon settlement sync, contact NetSuite Customer Support.

[Where are the settings for settlement sync and what do they mean?](#)

For information about settlement sync settings, read [Amazon Settlement Sync Settings in NetSuite Connector](#).

[Can NetSuite Connector post FBA inventory adjustments to NetSuite?](#)

Yes, you can use the Amazon Inventory Adjustment sync to update the FBA items inventory levels in NetSuite to reflect the availability in the Amazon warehouses. For more information, read [Amazon Inventory Adjustment in NetSuite Connector](#).

What happens if I deposit an order in one NetSuite accounting period but it is not settled until a new NetSuite accounting period?

The leading practice is to wait to deposit payments on orders and hold cash sales in **Undeposited Funds** until Amazon settles or reconciles the period for the order. Alternatively, to deposit orders in the period they were placed and not when they were settled or reconciled, post the order modifications and refunds as summary transactions for an entire settlement report. You can then split those summary transactions and the non-order modifications summary transactions, across months.

To merge order modifications or refunds into a single summary transaction:

1. Go to Settings > Other Transactions, and click the **Settlement** tab.
2. Click **Advanced Options**.
3. Check the following boxes as required:
 - Merge All Refunds on Report into a Single Summary Refund
 - Merge All Order Fees into a Single Transaction
 - When Posting Summary Transactions, Split them into Two Transactions if They Span Two Months
4. Click **Save**.

When recording refunds with Amazon Settlement sync, how does NetSuite Connector differentiate between a partial refund as a customer concession and a full refund for a returned product?

Amazon Settlement Reports may sometimes indicate that a refund was given for a specific SKU, when in fact it was a partial refund and the item was not returned to inventory. This is completely on Amazon's end. If Amazon is reporting a refund for a specific SKU, NetSuite Connector has no way of detecting if it should be a partial refund.

Does NetSuite Connector clear transactions when processing settlement reports, or do I need to reconcile them?

The transactions are posted as cash refunds or cash sales, which are already billed records in NetSuite. Therefore, you do not need to do any billing.

Amazon gives a financial concession to a customer or to us for an SKU on a settlement, and NetSuite Connector puts that as a cash refund using that SKU. Will NetSuite increase the quantity to incorrectly change the inventory totals?

A concession or similar entity is posted as an adjustment to the original transaction but does not affect the inventory for any SKU on the order. For more information, read [Concession and Refund Handling in NetSuite Connector](#)

If an order has been already invoiced, how does NetSuite Connector post modifications to that order?

Settlement fees and credits are synced to the billed record of the order, if available. If an order has not been invoiced yet, NetSuite Connector posts the changes to the original sales order. For more information, read [Managing Amazon Settlement Sync on NetSuite Connector](#).

Why is the REVERSAL_REIMBURSEMENT SKU importing individually as a cash refund?

NetSuite Connector sometimes creates a new Cash Refund record with the settlement fee, instead of adding the fee as part of the original cash sale. For more information, read [Q:](#)

How can I tell which fees will post as part of the order modifications and which will post as part of the non-order modifications?

NetSuite Connector includes any fee that has an order ID associated with it as an order modification. For more information, read [Identifying Settlement Fees Posted As Order Modifications](#).

Why is NetSuite applying tax to my settlement items?

NetSuite is applying tax mostly because the item is not assigned to a nontaxable tax schedule. For more information, read the help topic [Applying Tax to Settlement Items](#).

Why do I see the same fee multiple times on one order?

There are some fee types that Amazon will charge multiple times per order. Commission is one example. If an order has five items, each will have a commission fee. You will not see a single line item that is the sum of all the item commissions; instead, there will be five different commission fees. Other fees work similarly.

Why was a settlement fee added in a new cash refund instead of adding the fee as part of the original cash sale?

NetSuite Connector sometimes creates a new Cash Refund record with the settlement fee, instead of adding the fee as part of the original cash sale. This new record is created if NetSuite Connector detects that the original cash sale has already been deposited. Another reason is that NetSuite would not allow it to add an item to the cash sale, because the order may already be refunded, or the order has a return authorization.

How do I view or download settlement reports in Amazon?

To view or download the settlement reports in Amazon, read the SuiteAnswers article [View/Download Settlement Reports in Amazon](#).

Can NetSuite Connector combine all my settlement fees and credits from a settlement report into a single transaction line with the sum of the quantity and amount fields?

NetSuite Connector does not support combining different fees into a single line item whose quantity and total are the sum of the fee for the entire report. NetSuite Connector supports a single transaction for the entire report but fees or credits are not combined. For more information, read [Importing Amazon Settlement Reports in NetSuite Connector](#).

How can I set NetSuite Connector to automatically create the other charge for sale items during settlement sync?

Read [Importing Amazon Settlement Reports in NetSuite Connector](#).

How long does NetSuite Connector store Amazon settlement reports?

NetSuite Connector stores only the most recent three months of settlement reports.

Does NetSuite Connector sync settlement reports?

Yes. For more information, read [Importing Amazon Settlement Reports in NetSuite Connector](#).

Can I summarize all fees in one transaction for each settlement or two transactions for those periods that span two months?

Yes. To enable summarizing fees into one transaction per settlement on your account, contact NetSuite Customer Support.

You can get the lump sum fees for each settlement period.

Note: This feature does not combine each fee into a multiple quantity single line entry. For example, if two orders in the settlement have an **FBAPerUnitFulfillmentFee** SKU, you will see two line items with a quantity of one for each fee. You will also see the fee amount from each order. However, you will not see a single line item with two quantity and the sum of the fee from both orders.

Amazon Seller Fulfilled Prime (SFP) FAQ

Review the questions and answers below to learn more about Amazon seller fulfilled prime and NetSuite Connector.

[Do SFP orders import as cash sales or sales orders?](#)

NetSuite Connector imports SFP orders as sales orders because they need to be fulfilled in NetSuite.

[Does NetSuite Connector automatically mark orders as shipped in NetSuite?](#)

No, NetSuite Connector does not mark the orders as fulfilled because NetSuite Connector does not know if the order was shipped. NetSuite Connector only sends the labels to NetSuite. After the order is shipped, you should fulfill the order.

[I have a ShipStation connector with NetSuite Connector. What does the flow look like with ShipStation involved?](#)

The labels for SFP orders should be printed out directly from NetSuite and not from ShipStation. The SFP orders will be filtered out of the ShipStation connector.

Amazon Inventory Adjustment FAQ

[Does NetSuite Connector sync quantity available or quantity on hand?](#)

By default, NetSuite Connector compares the quantity on hand. However, if you want NetSuite Connector to compare quantity available, contact NetSuite Customer Support.

[Does inventory adjustment work with custom record alias SKUs?](#)

Inventory adjustments do not work with custom record alias SKUs. If you want to have both alias SKUs and inventory adjustment, you need to ensure that the FBA item is set up as the inventory item in NetSuite and the other item, which is most likely an MFN item, is set up as the custom record alias.

[Are kits included in inventory adjustments?](#)

Kits are included in inventory adjustments because they are treated as inventory items in Amazon.

Managing the BigCommerce Connector in NetSuite Connector

This section discusses the following topics about BigCommerce connector:

- [Mapping Categories for Full Product Sync in BigCommerce](#)

- Mapping BigCommerce Call For Pricing Message
- Default Mappings for BigCommerce Connector
- BigCommerce Order Status
- Refund Syncing Issues in BigCommerce Connector
- Troubleshooting the Issue That An Item has No Inventory, But Customers can Still Order the Item in BigCommerce
- BigCommerce Connector FAQ

Mapping Categories for Full Product Sync in BigCommerce

BigCommerce requires the categories field on all new items. When creating new items and using full product sync, you must map categories or the sync will fail. The categories field only applies on full product sync. If you use price/quantity sync, you must manually create each item and set the category field directly on each.

i Note: Currently, NetSuite Connector only allows a single NetSuite field to be used for category mapping. You must provide the complete category path otherwise you may see root category duplicated created. For example, if your item already exists on BigCommerce with the category Apple/iPhone/iPhone X and you map the categories field to a NetSuite field on that item that contains only iPhone X, syncing will create duplicates. You will see the item in BigCommerce with both Apple/iPhone/iPhone X and iPhone X categories.

To map categories for full product sync in BigCommerce:

1. Log in to app.farapp.com
2. Select the BigCommerce connector and relevant account.
3. Go to Mappings > Products.
4. On the Products page, click **Full Product**.
5. Click **Add Mapping**.
The Add BigCommerce Field Mapping window opens.
6. From the **Required** list, select **categories**.
7. Click **Add Mapping**.
8. Click **Close**.
9. From the Product Mappings list, locate your new mapping. The Mapping Type column defaults to Item Field.
10. In the NetSuite Field ID column, enter the field ID of the field in NetSuite from which the categories will sync.
11. (Optional) In the Default Value column, enter a value that NetSuite Connector will use when it does not find a value in your mapped NetSuite field from the previous step.
12. Click **Save**.

Mapping BigCommerce Call For Pricing Message

NetSuite Connector allows you to map the **Call for Pricing** field on a BigCommerce product listing. This feature uses **is_price_hidden** attribute in the NetSuite Connector product mapper. You can find the **Call for Pricing** attribute in BigCommerce and the corresponding mapping in NetSuite Connector in the following:

To map Call for Pricing message in NetSuite Connector:

1. Enable the **Call for Pricing** attribute in BigCommerce. See the help topic in BigCommerce [Adding Products](#) for more information.
2. Add a mapping with the attribute **is_price_hidden** in the NetSuite Connector product mapper. For more information, see the help topic [Setting Up Full Product Sync Mapping](#).

Default Mappings for BigCommerce Connector

This section lists the default mappings for the BigCommerce connector product sync and order sync.

BigCommerce Default Product Sync Mappings

Storefront Field	NetSuite Field
availability	custitem_fa_bc_avail
availability_description	custitem_fa_bc_availdesc
BigCommerce Flag Field	custitem_fa_bc_flag
brand_id	custitem_fa_bc_brand
categories	class
condition	custitem_fa_bc_condition
cost_price	custitem_fa_bc_costprice
custom_fields	custitem_fa_bc_custfld#
depth	custitem_fa_bc_depth
description	storedetaileddescription
fixed_cost_shipping_price	custitem_fa_bc_fixedshpprc
height	custitem_fa_bc_height
inventory_tracking	custitem_fa_bc_invtrack
inventory_warning_level	custitem_fa_bc_invwarnlv
is_condition_shown	custitem_fa_bc_condshown
is_featured	custitem_fa_bc_featured
is_free_shipping	custitem_fa_bc_freeship
is_price_hidden	custitem_fa_bc_prchidden
is_visible	isonline
meta_description	custitem_fa_bc_metadesc
meta_keywords	custitem_fa_bc_metakwds
name	storedisplayname
other-image-url#	custitem_fa_oth_img_url#

Storefront Field	NetSuite Field
page_title	custitem_fa_bc_pagetitle
preorder_release_date	custitem_fa_bc_pordrldt
price_hidden_label	custitem_fa_bc_prchdlbl
related_products	custitem_fa_bc_rltdprods
retail_price	custitem_fa_bc_retailprice
sale_price	custitem_fa_bc_saleprice
search_keywords	searchkeywords
sku	itemid
sort_order	custitem_fa_bc_sortorder
tax_class_id	custitem_fa_bc_taxclsid
type	custitem_fa_bc_prodtype
upc	upccode
warranty	custitem_fa_bc_warranty
weight	weight
width	custitem_fa_bc_width

BigCommerce Default Order Sync Mappings

Storefront Field	NetSuite Field
id	custbody_fa_channel_order
status	orderstatus
custom_status	status
shipping_cost_tax	shippingCost
total_tax	total
billing_address	billingAddress
shipping_addresses	shippingAddress

BigCommerce Order Status

When BigCommerce orders are imported, NetSuite Connector automatically changes their status in the storefront to **Awaiting Shipment**. This process cannot be modified to update the status of the order.

You can, however, change the label of the status in BigCommerce. For instance, you can change the label from **Awaiting Shipment** to **Sent to NetSuite Connector**. The BigCommerce interface displays **Sent to NetSuite Connector**, but in the API, the status is still **Awaiting Shipment**. For more information about renaming the order status in BigCommerce, check the BigCommerce website article <https://support.bigcommerce.com/s/article/Order-Statuses#rename>.

Refund Syncing Issues in BigCommerce Connector

Triggering a refund to the payment method is supported only under the BigCommerce v3 API. However, NetSuite Connector currently only supports BigCommerce v2 API. Therefore, NetSuite Connector does not refund the customer's payment method when syncing refunds.

Troubleshooting the Issue That An Item has No Inventory, But Customers can Still Order the Item in BigCommerce

If customers can place orders for products even if the inventory of a product is 0, check the following:

- Make sure the **inventory_tracking** field is not set to **None** for the product.
- If you are syncing the **inventory_tracking** field from NetSuite, make sure the mapped field in NetSuite is not set to **None**. If you are not syncing the **inventory_tracking** field, then check and change the field directly in the BigCommerce connector.

BigCommerce Connector FAQ

To learn more about BigCommerce connector, review the questions and answers below.

[Can I separate product image description and image address on BigCommerce product sync?](#)

No, the BigCommerce's product API does not permit separating those fields.

[Why is NetSuite Connector making a Put request to my BigCommerce](#)

NetSuite Conenctor updates the order status in BigCommerce after the order status is retrieved through an additional **Application Programming Interface (API)** call. The status update is done through a **Put** API call.

For more information about status updates and how you can customize the label of statuses that are updated, see [BigCommerce Order Status](#).

Managing the eBay Connector in NetSuite Connector

This section discusses the following topics about eBay connector:

- [Managing eBay Categories in NetSuite Connector](#)
- [Default Mappings for eBay Connector](#)
- [Mapping the eBay Return Profile ID in NetSuite Connector](#)
- [Mapping eBay Buyer IDs to Customer Records in NetSuite](#)
- [Mapping eBay Images in NetSuite Connector](#)
- [Posting 0 Quantity During Product Sync in NetSuite Connector](#)
- [Troubleshooting eBay Product Sync Errors in NetSuite Connector](#)

- [Troubleshooting eBay Sync Issues](#)
- [eBay Connector FAQ](#)

eBay Markdown Manager Support in NetSuite Connector

eBay's Markdown Manager is a feature that lets customers mark down item prices within eBay. However, eBay does not provide the functionality to NetSuite Connector item updates to work with the Markdown Manager. eBay does not provide the functionality for NetSuite Connector to sync an update that omits prices. Prices set through the Markdown Manager are overwritten whenever NetSuite Connector syncs updates for an item.

Default Mappings for eBay Connector

This section lists the default mappings for the eBay connector product sync and order sync.

eBay Product Sync Default Mappings

Storefront Field	NetSuite Field
Brand	manufacturer
ConditionID	custitem_fa_ebay_condition
Country	custitem_fa_ebay_country
Currency	custitem_fa_currency
Description	storedetaileddescription
DispatchTimeMax	custitem_fa_ebay_shiptime
eBay Flag Field	custitem_fa_ebay_flag
eBayCategory	custitem_fa_ebay_category
eBayStoreCategory	custitem_fa_ebay_store_cat
FreeShipping#	custitem_fa_ebay_free_shp#
HitCounter	custitem_fa_ebay_hit_count
InternationalPackagingHandlingCosts	custitem_fa_ebay_i_pkcost
InternationalShippingService#	custitem_fa_ebay_i_shpsvc#
InternationalShippingServiceAdditionalCost#	custitem_fa_ebay_i_shpac#
InternationalShippingServiceCost#	custitem_fa_ebay_i_shpcst#
InternationalShipToLocation#	custitem_fa_ebay_i_shplocs
ListingDuration	custitem_fa_ebay_duration
Location	custitem_fa_ebay_location
PackageDepth	custitem_fa_ebay_pkg_depth
PackageLength	custitem_fa_ebay_pkg_len

Storefront Field	NetSuite Field
PackageWidth	custitem_fa_ebay_pkg_width
PackagingHandlingCosts	custitem_fa_ebay_pkg_costs
PaymentMethods	custitem_fa_ebay_pmt_meths
PayPalEmailAddress	custitem_fa_paypal_email
PictureURL	storeDisplayImage.internalId
PostalCode	custitem_fa_ebay_zipcode
RefundOption	custitem_fa_ebay_refundopt
ReturnPolicyDescription	custitem_fa_ebay_rtrn_plcy
ReturnsAcceptedOption	custitem_fa_ebay_rtrns_ok
ReturnsWithinOption	custitem_fa_ebay_rtrn_wthn
ShippingCostPaidByOption	custitem_fa_ebay_shp_pd_by
ShippingIrregular	custitem_fa_ebay_shp_irreg
ShippingPackage	custitem_fa_ebay_ship_pkg
ShippingService#	custitem_fa_ebay_shipsvc#
ShippingServiceAdditionalCost#	custitem_fa_ebay_shpsvcac#
ShippingServiceCost#	custitem_fa_ebay_shpcost#
ShippingSurcharge#	custitem_fa_ebay_surchg#
ShippingType	custitem_fa_ebay_shiptype
ShipToLocation#	custitem_fa_ebay_shptolocs
sku	itemid
SubTitle	custitem_fa_ebay_subtitle
Title	storedisplayname
UseTaxTable	custitem_fa_ebay_usetaxtbl
VariationSKU	itemid
WeightMajor	weight
eBayCategory	custitem_fa_ebay_category
eBayStoreCategory	custitem_fa_ebay_store_cat
sku	itemId

eBay Default Order Sync Mappings

Storefront Field	NetSuite Field
billtoaddress1	addr1

Storefront Field	NetSuite Field
billtoaddress2	addr2
billtocity	city
billtocountry	country
billtophone	addrphone
billtostate	state
billtozip	zip
buyerfullname	addressee
clientorderid	order internalId
ebaypaymentstatus	status
itemtotal	amount
orderid	custbody_fa_channel_order
shippingtotal	shippingCost
shiptoaddress1	addr1
shiptoaddress2	addr2
shiptocity	city
shiptocountry	country
shiptoemail	tobeemailed
shiptofirstname	addressee
shiptofullname	addressee
shiptolastname	addressee
shiptophone	addrphone
shiptostate	state
shiptozip	zip
status	orderstatus

Managing eBay Categories in NetSuite Connector

eBay requires a main Category field, and optionally allows a Store Category to be set. The values for both must be posted to eBay as numeric Category IDs, unless the user only has one or two categories, which must be read from a field in NetSuite.

There is a standard eBay category rule in NetSuite Connector that is designed to read from NetSuite fields. You must create the custom fields in NetSuite to accommodate the eBay categories you need.

To create a custom list for eBay categories:

1. Go to Customization > List, Records, & Fields > Lists > New.

2. Enter appropriate information if the following fields:
 - **Name** - Enter the name of the list
 - **ID** - Enter alphanumeric ID for the list
 - **Matrix Option List** - Check if the list is for matrix items
 - **Value** - Enter Category Name from eBay Categories
 - **Abbreviation** - Enter Category ID from eBay Categories
3. Click **Add**.
4. Click **Save**.

For more information, see the help topic [Creating a Custom List](#).

To create a custom item field for eBay categories:

1. Go to Customization > List, Records, & Fields > Item Fields > New.
 - Name: Enter Name for the List
 - ID: Enter Alphanumeric ID for the List
2. Enter appropriate information if the following fields:
 - **Name** - Enter the name of the list
 - **ID** - Enter alphanumeric ID for the list
 - **Type** - Select List/Record
 - **List/Record** - Select the previously created custom List
3. Click **Save**.

The same format can be used for Store Category fields, just ensure to note that store categories, and their IDs, are different than the eBay categories; also, the field format must be the same, though the field values are not.

eBay Category Reference

eBay Category IDs can be found using four methods:

- I Sold What? has a current list of eBay categories. To find a leaf category, you should expand a particular category until you cannot expand it anymore, and the category ID is marked with a leaf.
- If you go to ebay.com you can browse for a particular category whose category ID is displayed in the URL. For example, if you browse for Cell Phone Accessories, <http://www.ebay.com/sch/Cell-Phone-Accessories-/9394/i.html>. The category ID is 9394. However, this is not a leaf category, so you are not allowed to list items anymore. You have to select one of the child categories until there are no more child categories left. If you select Chargers & Cradles, there are no more child categories, so this is a leaf category, <http://www.ebay.com/sch/Chargers-Cradles-/123417/i.html>. The leaf category ID you can use to list in this category is 123417.
- NetSuite Connector has a list of available eBay categories. To view the list, log in to NetSuite Connector, select Manage Account, and then select eBay Dashboard. Select the eBay Categories tab, to view the available categories. You can use the Find button to search for categories you want. Some categories cannot be listed. This means that you cannot use those category IDs to post to eBay as they are parent categories, and only child categories are valid.
- The list of categories are at <http://pages.ebay.com/sellerinformation/news/categorychanges.html>.

Mapping the eBay Return Profile ID in NetSuite Connector

You can set up return policies in eBay to define how you want to handle product returns. For instance, you can choose the time frame customers can return products or set who pays for return shipping. You can also set up multiple policies under Return Preferences in your eBay dashboard. For more information, read the eBay help article <https://www.ebay.com/help/selling/managing-returns-refunds/handling-return-requests/setting-rules-return-requests?id=4368>.

Each return policy you set up in eBay will have a Return Profile ID you can map in NetSuite Connector. NetSuite Connector will apply the mapped return policy when it posts to eBay during a full product sync.

To map the eBay Return Profile ID:

1. Log in to app.farapp.com.
2. Select eBay connector and then select the relevant account.
3. Go to Mappings > Products.
4. Click **Fixed Price Items**.
5. Click **Add Mapping**.
The Add eBay Field Mapping popup window opens.
6. From the Standard list, select **ReturnProfileID**.
7. Click **Add Mapping**.
A success message appears on the popup window.
8. Click **Close**.
9. Locate your new mapping in the product mappings list. Under the Mapping Type column, select **Use Default**.
10. In the Default Value column, enter the return profile ID.
11. (Optional) If you are using an item field in NetSuite to define the return profile ID, select **Item Field** under Mapping Type for step 9. Enter the NetSuite field in the NetSuite Field ID column.
12. Click **Save**.
13. (Optional) If you have variation items, repeat steps 4-12 for each item but select **Variation Items** in step 4.

Mapping eBay Buyer IDs to Customer Records in NetSuite

The following procedure describe the instructions on mapping the buyer ID from the eBay order data to the customer record in NetSuite.

To map eBay Buyer IDs to customer records in NetSuite

1. Identify the NetSuite field on the customer record that NetSuite Connector to post the Buyer ID to. To create a custom field, see the help topic [Creating Custom Entity Fields](#).
2. In NetSuite Connector, navigate to the eBay mappings page eBay connector > Mappings > Orders and select **Customer** tab.
3. On the **Customer Mappings** page, click **Add Mapping**.
4. On the **Add NetSuite Mapping** window, select the NetSuite field to map the eBay Buyer ID to from the dropdown list. If you are using a custom field, select **Custom Field**.

If **Custom Field** is selected:

1. A **Custom Field ID** field will appear below the **NetSuite Field** dropdown. Enter the **Field ID** of the custom field you created in NetSuite.
2. Click **Add Mapping**.
3. Click **Close**.
5. In the **Mapping Type** column, select **Order Header** from the dropdown list.
Selecting Order Header will change the **eBay Field/Fixed Value** column field into a dropdown.
6. Under the **eBay Field/Fixed Value** column, select **Buyer ID** from the dropdown list.
7. Click **Save**.

Your new mapping will be applied to all new orders. This mapping will not apply to orders already posted to NetSuite.

Mapping eBay Images in NetSuite Connector

This section provides guidelines for mapping images in eBay.

NetSuite Configuration

The NetSuite configuration for mapping images is common for all connector types. For more information, read [NetSuite Configuration](#).

Mapping eBay Images in NetSuite Connector

After mapping images in NetSuite, you can map the images in NetSuite Connector.

To map eBay images in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select eBay connector and then select the relevant account.
3. Go to Mappings > Products.
4. Click the appropriate mapping category: .
 - If mapping images for stand-alone items, click **Fixed Price Items**
 - If mapping images for variation items, click **Variation Items**.
5. Click **Add Mappings**.

The Add eBay Field Mapping popup window opens.
6. From the **Standard** dropdown list, select the appropriate field:
 - If the mapping is for fixed price items category, select **PictureURL**.
 - If the mapping is for variation items category, select **VariationPictureURL**.
7. Click **Add Mapping**.

A success message appears on the popup window.
8. Click **Close**.
9. Locate your new mapping in the product mappings list and click **Special Mapping – Click to Edit** link on the mapping.

The field mapping popup window opens.

10. In the **Image Field 1** field, enter the field ID of the custom image field.
11. (Optional) To add multiple image fields, click the **Add Row** button and enter the field IDs in the new image fields.
12. Click **Save Changes**.
13. If a popup window to reload the items opens, click **Yes**.

Posting 0 Quantity During Product Sync in NetSuite Connector

The Out-of-Stock feature in eBay lets NetSuite Connector send 0 quantity for your product listings. When this feature is enabled, NetSuite Connector can decrease the inventory below 1. If you sell eBay items on other connectors, this feature prevents overselling.

For more information, check the following eBay documentation <https://developer.ebay.com/DevZone/guides/features-guide/default.html#development/listings-useoutofstock.html>.

Note: After enabling the Out-of-Stock feature, reload the item from NetSuite to NetSuite Connector manually. Else, you can make some change to the item and trigger the sync of the item. Perform this action if your item was excluded in NetSuite Connector during product sync because the Out of Stock option was not enabled in eBay. For more information about reloading items, read [Reloading Products in NetSuite Connector](#).

Troubleshooting eBay Product Sync Errors in NetSuite Connector

Following are some common eBay product sync errors and their possible resolutions.

Error	Description
Error 107 – The category is not valid, select another category.	Indicates that the eBay category selected for the item is not valid. Make sure you select the correct category for the item. Note that eBay category is different from your store category. For more information about eBay categories, read Managing eBay Categories in NetSuite Connector .
Error 21916608 – Variation cannot be deleted during restricted revise.	Indicates that one or more orders are created on a variation listing and a new variation child is being added to the listing. eBay raises a flag that transactions have happened on the listing and is incorrectly restricting the options that can be revised. To resolve this error, follow these steps: <ol style="list-style-type: none"> 1. Manually end the listing. 2. On the eBay dashboard, create a new listing (Map & Post Single Item).
Error 10007 – Internal error to the application.	Indicates an error on the eBay server side. Wait until eBay fixes their issue. After eBay fixes the issue, the resolves by itself.
Error 21917098 – The shipping rate table is not applicable for this site.	Indicates you are attempting to relist an item and your shipping options have changed. The error usually resolves when you post the item as a new listing instead of relisting.

Troubleshooting eBay Sync Issues

This topic describes some common issues eBay sync issues.

eBay Issue Generating Gallery Images

If you get this error, the color model of your image is CMYK. You must convert the CMYK image to RGB. For more details, check the CMYK Color Format section in the eBay website page <https://developer.ebay.com/DevZone/guides/features-guide/default.html#development/pictures-inlisting.html>.

The page also gives instructions on detecting whether an image is in CMYK color format and how to convert it to RGB. After updating the image, you must upload the image again to eBay. The following page provides more information about this error: <https://ebaydts.com/eBayKBDetails?KBid=669>.

eBay Order Mail Missing

eBay only returns the email on an order if the checkout or payment is completed within few days after placing the order. For more information, check the following link on eBay website <https://ebaydts.com/eBayKBDetails?KBid=349>.

If a buyer completes the checkout process more than a few days after placing the order, the buyer's email address is returned as **Invalid Request**. In such situations, NetSuite reports an error because NetSuite requires a valid email address. To resolve this error, from the NetSuite Connector's Orders dashboard, edit the order with the buyer's correct email address.

eBay Order Sync Issues

You may get the following issues when importing orders into NetSuite Connector:

- Order was not imported into NetSuite Connector.
- Order was imported late into NetSuite Connector.

For these issues, check if the customer has completed the checkout in eBAy. NetSuite Connector only imports orders that have the checkout status of **Complete**. This status means that the customer is done with adding items to the order.



Note: It is possible for a customer to pay for an order but not complete the checkout.

eBay Connector FAQ

To learn more about eBay connector, review the questions and answers below.

Does NetSuite Connector support eBay's Volume Discount feature?

Currently NetSuite Connector does not support eBay's Volume Discount feature.

Can I map the Promote a Listing field on my eBay products in NetSuite Connector?

The Promote a Listing feature uses the eBay Marketing API, which is different from the FixedPriceItem API used by NetSuite Connector. Therefore, you cannot use the feature with NetSuite Connector. You must manage this feature from eBay.

Does NetSuite Connector support creating eBay auction listing?

No, you can only use NetSuite Connector to sync fixed price listings. NetSuite Connector does not sync to existing items that are listed as auction items. You must delete those listings and recreate them as fixed price listings.

What do I need to sync certified refurbished items?

There is not special setup needed in NetSuite Connector. Set the **Condition ID** to the value for **Certified Refurbished** (Condition ID = 2000) in the NetSuite field mapped to the **ConditionID** in your eBay product sync.

Why are my images not updating on eBay?

eBay checks the URL or file name coming into the site. If the URL or file name has the same value as in eBay's system, it assumes that the image has not changed and does not update the image on the eBay site. You can fix the issue using one of the following methods:

- Change the name of the image. With this method, eBay detects that the image is new and updates it.
- Set the image field temporarily blank. After the update pushes through, set the image back to the original value. The blank value will overwrite the value on the eBay system. When you switch back, the blank value will be overwritten by your new image.
- Make sure each image meets eBay's minimum size requirements. Currently, the requirement is 500 x 500 pixels, but eBay passes an error message to NetSuite Connector providing the minimum size requirement. Check every image that you sync to the Stock Keeping Unit (SKU).

How to Prevent an eBay Listing from Ending

Enable the eBay **Out of Stock** setting for NetSuite Connector to update eBay with zero quantity listing. Enabling this setting also means that your eBay listing will not end when the item goes out of stock. If the item remains out of stock for the entire 30-day billing period for 3 consecutive 30-day billing periods, eBay ends the listing.

For more information on enabling this setting, see [Posting 0 Quantity During Product Sync in NetSuite Connector](#).

Managing Magento 2 Connector in NetSuite Connector

This section discusses the following topics about Magento 2 connector:

- [Default Mappings for Magento 2 Connector](#)
- [Mapping Magento 2 Custom Product Attributes in NetSuite Connector](#)
- [Creating a Magento 2 Order Invoice in NetSuite Connector](#)
- [Troubleshooting Order Status Not Changed in Magento 2 After Shipping](#)

Default Mappings for Magento 2 Connector

This section lists the default mappings for the Magento 2 connector product sync and order sync.

Magento 2 Default Product Sync Mappings

Storefront Field	NetSuite Field
backorders	custitem_fa_mag_backorders
cost	cost
country_of_manufacture	countryofmanufacture
custom_design	custitem_fa_mag_cust_des
custom_design_from	custitem_fa_mag_cust_desfr
custom_design_to	custitem_fa_mag_cust_desto
custom_layout_update	custitem_fa_mag_cust_layou
description	storedetaileddescription
enable_googlecheckout	custitem_fa_mag_enbl_gglco
enable_qty_increments	custitem_fa_mag_enblqtyinc
gift_message_available	custitem_fa_mag_gift_msg
has_options	custitem_fa_mag_hasoptions
image	storeDisplayImage.internalId
image_label	storedisplayimagedescrip
is_decimal_divided	custitem_fa_mag_is_dec_div
is_qty_decimal	custitem_fa_mag_is_qty_dec
Magento Flag Field	custitem_fa_magento_flag
manage_stock	isdropshipitem
manufacturer	manufacturer
max_sale_qty	custitem_fa_mag_maxsaleqty
meta_description	custitem_fa_mag_meta_desc
meta_keyword	searchkeyworks
meta_title	custitem_fa_mag_meta_title
min_qty	minimumquantity
min_sale_qty	minimumquantity
msrp	custitem_fa_msrp
msrp_display_actual_price_type	custitem_fa_mag_show_price
msrp_enabled	custitem_fa_mag_msrp_enbld
name	storedisplayname
news_from_date	custitem_fa_mag_new_fm_dt

Storefront Field	NetSuite Field
news_to_date	custitem_fa_mag_new_to_dt
notify_stock_qty	reorderpoint
options_container	storeitemtemplate
page_layout	custitem_fa_mag_pagelayout
qty_increments	custitem_fa_mag_qty_incs
required_options	custitem_fa_mag_rqd_opts
short_description	salesdescription
sku	itemid
small_image	storeDisplayImage.internalId
small_image_label	SPECIAL CASE
special_from_date	custitem_fa_mag_spcl_fm_dt
special_to_date	custitem_fa_mag_spcl_to_dt
status	isinactive
tax_class_id	istaxable
thumbnail	storeDisplayImage.internalId
thumbnail_label	SPECIAL CASE
url_key	urlcomponent
url_path	custitem_fa_mag_url_path
use_config_backorders	custitem_fa_mag_use_cfg_bo
use_config_enable_qty_inc	custitem_fa_mag_cenbqtyinc
use_config_manage_stock	custitem_fa_mag_cfgmngstk
use_config_max_sale_qty	custitem_fa_mag_cmaxsalqty
use_config_min_qty	custitem_fa_mag_cfgminqty
use_config_min_sale_qty	custitem_fa_mag_minsaleqty
use_config_notify_stock_qty	custitem_fa_mag_ntfystkqty
use_config_qty_increments	custitem_fa_mag_cfgqtyincr
visibility	isonline
weight	weight
_attribute_set	custitem_fa_mag_attr_set
_category	sitecategorylist
_product_websites	custitem_fa_mag_prod_sites
_store	custitem_fa_mag_store

Magento 2 Order Sync Default Mappings

Storefront Field	NetSuite Field
billtoaddress1	addr1
billtoaddress2	addr2
billtocity	city
billtocountry	country
billtophone	addrPhone
billtostate	state
billtozip	zip
buyerfullname	addressee
clientaddressbookid	<i>internalId</i>
clientcustid	<i>internalId</i>
clientorderid	<i>internalId</i>
orderid	custbody_fa_channel_order
ordertotal	custbody_fa_order_total
shiptoaddress1	addr1
shiptoaddress2	addr2
shiptocity	city
shiptocountry	country
shiptofullname	addressee
shiptophone	addrPhone
shiptostate	state
shiptozip	zip
status	orderStatus
taxtotal	taxtotal

Mapping Magento 2 Custom Product Attributes in NetSuite Connector

Magento 2 lists a number of product attributes within the `custom_attributes` tag. Each of these attributes is identified by an attribute code. To map a value to one of these custom attributes, you'll need to know what the corresponding attribute code is. You should be able to find these values from within your Magento 2 admin site.

Some of the common attribute codes are:

- `preferred_vendor`

- image
- url_key
- description
- cost
- small_image
- options_container
- short_description
- is_hazardous_material
- quantity_on_order
- tax_class_id
- thumbnail
- units
- quantity_and_stock_status
- msrp_display_actual_price_type
- categories
- oem_number
- featured
- sale
- required_options
- has_options
- manufacturer
- country_of_manufacture
- tariff_code
- fitment
- mst_search_weight
- size



Note: Ensure that your attributes are set to **Global** in Magento 2 or else when you try to update the attributes, they will not save and NetSuite Connector will be unable to post to the attributes.

After the attribute code you want to map to is identified, you can create this product mapping from the Product Mapper in NetSuite Connector.

To create a product mapping for Magento 2 in NetSuite Connector:

1. In NetSuite Connector, navigate to the Product Mapping page.
2. Click **Add Mapping**.
3. In the **Standard** dropdown, select custom_attributes.
4. Click the **Add Mapping**.
5. Locate the row you added for custom_attributes and click **Special Mapping - Click to Edit**.
6. In the custom_attributes Mapping window, enter the Magento 2 attribute code in the **Attribute** column on the left. Enter the NetSuite field you want to map to in the **Attribute Value Field** column on the right.

If there are multiple custom attributes you will add, click **Add Row** and repeat this step.

7. Click **Save Changes** and close the mapping popup window.
8. Click **Save**.

Creating a Magento 2 Order Invoice in NetSuite Connector

You can create an invoice in Magento 2 when a fulfillment syncs from NetSuite Connector.

To create an invoice in Magento 2 from NetSuite Connector:

1. Log in to app.farapp.com.
2. Select the Magento 2 connector and the relevant account.
3. Go to Settings > Orders.
The Order Settings page opens.
4. Click **Advanced Options**.
5. Check the **Trigger Magento to Create Invoices When Completing Fulfillments** box.
6. (Optional) To capture the invoice after creating it, check **Check Here To Also Capture Invoice**.
The **Check Here To Also Capture Invoice** box appears only when you check the **Check Here To Also Capture Invoice** box in step 5.
7. Click **Save**.

Troubleshooting Order Status Not Changed in Magento 2 After Shipping

NetSuite Connector does not mark the order as Complete in Magento 2 after the order is shipped to the customer. This is because Magento 2 should be the one to mark it as complete after an invoice is generated for the order. To have NetSuite Connector initiate the creation of an invoice in Magento 2 when it fulfills the order, see [Creating a Magento 2 Order Invoice in NetSuite Connector](#).

Managing the Shopify Connector in NetSuite Connector

This section discusses the following topics about Shopify connector:

- [Setting Up Real-Time Order Sync for Shopify in NetSuite Connector](#)
- [Shopify Inventory Management Mapping](#)
- [Default Mappings for Shopify Connector](#)
- [Mapping Shopify Multilocation Inventory in NetSuite Connector](#)
- [Mapping Multilocation Orders in Shopify](#)
- [Mapping Shopify Images in NetSuite Connector](#)
- [Mapping Shopify Order Risk Data to NetSuite](#)

- Mapping Shopify Metafields in NetSuite Connector
- Tracking Shopify Item Inventory in NetSuite Connector
- Handling Shopify Cancel and Refund Emails in NetSuite Connector
- Syncing Canceled Orders to Shopify Using NetSuite Connector
- Shopify Point of Sale (POS) Orders
- Assigning Default Values to Shopify Point-of-Sale (POS) Orders in NetSuite Connector
- Shopify Gift Cards
- Shopify Product IDs
- Using the Published Scope and Published Fields for Mapping in Shopify
- Using Shopify's Compare At Feature in NetSuite Connector
- Troubleshooting Shopify Sync Errors in NetSuite Connector
- Shopify Connector FAQ

Setting Up Real-Time Order Sync for Shopify in NetSuite Connector

Enabling real-time order sync may result in a subscription increase. Make sure you understand your NetSuite Connector pricing and any potential increase resulting from real-time order sync.



Important: This procedure is new as of June 2021. If you were previously using Shopify real-time order sync and want to reactivate, remove the previous webhook directly in Shopify before following the below procedure.

To set up real-time order sync for Shopify:

1. Log in to app.farapp.com
2. Select the Shopify connector and relevant account.
3. Go to Settings > Orders.
4. On the Order Settings page, expand **Advanced Options**.
5. Under Advanced Options, click **Add Real-Time Sync Webhook**.

Default Mappings for Shopify Connector

This section lists the default mappings for the Shopify connector product sync and order sync.

Shopify Default Product Sync Mappings

Storefront Field	NetSuite Field
body_html	salesDescription
handle	custitem_fa_shopify_handle

Storefront Field	NetSuite Field
images	storeDisplayImage.internalId
product_type	custitem_shopify_product_type
published_at	custitem_fa_shpfy_pub_at
published_scope	custitem_fa_shpfy_pubscope
tags	custitem_fa_shpfy_tags
title	displayName
metafields_global_title_tag	custitem_fa_shpfy_metatitl
metafields_global_description_tag	custitem_fa_shpfy_metadesc
vendor	manufacturer
barcode	upcCode
inventory_management	isdropshipitem
inventory_policy	custitem_fa_shpfy_backords
metafield	custitem_fa_shpfy_metafld
requires_shipping	custitem_fa_shpfy_reqship
sku	itemId
taxable	istaxable
title	displayName
weight	weight
weight_unit	weightUnit.name
collections	class
published	isonline
variant_image	storedisplayimage
variant_title	storedisplayname

Shopify Default Order Sync Mappings

Storefront Field	NetSuite Field
billing_address	billingAddress
created_at	createdDate
currency	currencyName
customer	entity
email	tobeemailed
fulfillment_status	orderstatus

Storefront Field	NetSuite Field
id	custbody_fa_channel_order
line_items	itemList
name	addressee
order_number	tranid
payment_gateway_names	paymentoption
phone	addrphone
shipping_address	shipaddress
taxes_included	istaxable
total_price	total

Shopify Inventory Management Mapping

The Shopify connector has a specific inventory_management field that tells Shopify whether or not it should be tracking inventory for items on its platform. The field references the **Track Quantity** box on the Shopify user interface.

You can check or clear this box through the inventory_management mapping. Also, you can also set up the mapping logic to have this box checked for some items and cleared for other items. To do this, create an **Item Field Translation** mapping by following these steps.

To create Item Field Translation mapping:

1. In NetSuite, create a custom check box, because you will only need true or false values.
2. Determine the logic that you want to apply for the mapping. For example, if the box is checked, inventory is tracked, or if the box is cleared, inventory is not tracked.
3. Log in to app.farapp.com and go to product mappings, Shopify > Mappings > Products > Full Product. Locate the inventory_management mapping.
4. In the **Mapping Type** column, select **Item Field Translation**.
5. In the translation mapping popup page, do the following:
 - a. Enter the relevant field ID in **NetSuite Field**.
 - b. Set the on or off logic switch; then, enter the equivalent Boolean value, true or false, under the NetSuite Value column, and the corresponding action (to track inventory or not) under Shopify Value column.



Note: Setting inventory_tracking to false only for some items will only work if you are not passing over an inventory_quantity value from NetSuite. Typically, this mapping should only apply to noninventory items.

- c. Click **Save Changes**.

When syncing items, an error might be encountered indicating a failure to update inventory because the inventory item does not have inventory tracking enabled. This means inventory_quantity is being received with a value even if it is 0, and Shopify will not accept not tracking inventory for those items. In this case, you must contact the NetSuite Connector Support team to get help in setting up another conditional mapping for inventory_quantity, to handle such scenarios.

Mapping Shopify Multilocation Inventory in NetSuite Connector

If you are using multiple locations in Shopify, you must designate the locations in Shopify to which you want to sync inventory by mapping them with the locations in NetSuite.



Note: Before proceeding, note the numeric Shopify location IDs.



Important: If you have previously mapped your inventory without Shopify's multi-location feature enabled, you must delete the previous mappings before following the below procedure.

To map multilocation inventory in Shopify:

1. Log in to app.farapp.com.
2. Select the Shopify connector and the relevant account.
3. Go to Mappings > Products.
The Product Mappings page opens.
4. Click **Add Mapping**.
The Add Shopify Field Mapping window opens.
5. From **Standard** list, select **inventory_quantity**.
6. Click **Add Mapping** and then click **Close**.
7. From the Product Mappings list, locate the **inventory_mapping** and click **Special Mapping – Click to Edit**.
The **inventory_quantity** Mapping window opens.
8. In the **Shopify Location ID** field, enter the location ID.
9. From the **Location Internal ID** list, select the corresponding NetSuite field ID.
10. (Optional) To add more locations, click **Add Row** and repeat steps **8** to **9**.
11. Click **Save Changes** and then click **Save**.

Mapping Multilocation Orders in Shopify

If you are using multiple locations in Shopify, you can map each location from Shopify to the corresponding location in NetSuite for your orders. This mapping will help you in tracking the Shopify location from which the orders are posted.



Note: If you want to enable the multi-location feature for your shop in Shopify, you should do that before completing the following procedure. Otherwise, you will have to delete the existing mappings and add them again.

To map multilocation orders in Shopify:

1. Log in to app.farapp.com.
2. Select the Shopify connector and the relevant account.

3. Go to Mappings > Orders.
4. Click the **Orders** tab.
The Order Mappings page opens.
5. Click **Add Mapping**.
The Add NetSuite Mapping window opens.
6. From **NetSuite Field** list, select **Location**.
7. Click **Add Mapping**.
8. Click **Close**.
The Location row is added at the bottom of the Order Mappings page.
9. On the Order Mappings page, locate the Location row and from the list in the **Mapping Type** column, select **Order Header with Translation**.
10. From the list in the **Shopify Fixed/Field Value** column, select **Location ID**.
The Location ID to Location Mapping popup window opens.
11. To set NetSuite location for existing rows in the popup window, select a value in the **NetSuite Value** column.
12. (Optional) To add a new row that contain new Shopify location:
 - a. Click **Add Row**.
 - b. In the **Shopify Value** column, enter the Shopify location ID.
 - c. In the **NetSuite Value** column, select NetSuite location that you want to map to the Shopify location ID.
13. (Optional) To add more locations, click repeat step **12**.
14. Click **Save**.
15. Click **Close**
16. On the Order Mappings page, click **Save**.

After the above procedure is completed, you can test the mappings by creating a test order in Shopify and following the procedure given in [NetSuite Connector Manual Sync Tests](#).

Mapping Shopify Images in NetSuite Connector

This section provides steps for mapping Shopify images in NetSuite Connector. The NetSuite configuration for mapping images is common for all connector types. For more information, read [NetSuite Configuration](#).

After setting up product image fields in NetSuite, you can map the images in NetSuite Connector.

To map Shopify images in NetSuite Connector:

1. Log in to [app.farapp.com](#).
2. Select Shopify connector and then select the relevant account.
3. Go to Mappings > Products.
4. Click **Add Mappings**.
The Add Shopify Field Mapping popup window opens.
5. From the **Standard** dropdown list, select **images**:

6. Click **Add Mapping**.
A success message appears on the popup window.
7. Click **Close**.
8. Locate your new mapping in the product mappings list and click **Special Mapping – Click to Edit** link on the mapping.
The Images Mapping popup window opens.
9. In the **Image Field 1** field, enter the NetSuite field ID of the image you want mapped to Shopify.
10. (Optional) To add multiple image fields, click the **Add Row** button and enter the field IDs in the new image fields.
11. Click **Save Changes**.

Mapping Shopify Order Risk Data to NetSuite

Shopify has a native fraud detection feature you can enable for your orders. You can also map this order risk data to a custom field in NetSuite. This section provides steps for creating the custom free-form text field in NetSuite and using NetSuite Connector to map the risk data from Shopify.

For more information on custom NetSuite fields, read the help topics [Creating Custom Transaction Body Fields](#) and [Creating a Custom Field](#).

To map Shopify order risk data to NetSuite:

1. Log in to app.farapp.com.
2. Select Shopify connector and then select the relevant account.
3. Go to Mappings > Orders.
4. On the Order tab, click **Add Mapping**.
The Add NetSuite Mapping popup window opens.
5. From the NetSuite Field list, select **Custom Field**.
6. On the **Custom Field ID** field, enter the field ID. This ID begins with **custbody** and is suffixed with the ID name given to the field in NetSuite.
7. Click **Add Mapping**.
A success message appears on the popup window.
8. Click **Close**.
9. Locate your new mapping in the order mappings list. Under the Mapping Type column, select **Order Header**.
10. In the Shopify Field/Fixed Value column, select **Custom Field**.
The Map Custom Order Field popup windows opens.
11. Select the account and enter an existing Order ID containing risk data.
12. Click **View Field Data**.
A list of fields found in the order appears.
13. The risk data Shopify provides is found in a field named **risks.0.recommendation**. Locate this field on the list and click the plus sign next to it.
14. Click **Save**.

Mapping Shopify Metafields in NetSuite Connector

Shopify metafields (also known as Shopify custom fields) let you add additional information to your product listings. For more information on metafields and how to create them, read the Shopify help article <https://help.shopify.com/en/manual/metafields>.

Like any other field that you map from NetSuite, you need the NetSuite field ID. From the Shopify site, you need to find the metafield name and the namespace values.

When configuring your mapping, NetSuite Connector uses the **Key** field for the metafield name and **Namespace** field for the namespace.



Note: Metafield mappings can only be done using Full Product sync.

To map Shopify metafields in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select Shopify connector and then select the relevant account.
3. Go to Mappings > Products.
4. Click **Full Product**.
The Full product button turns gray.
5. Click **Add Mappings**.
The Add Shopify Field Mapping popup window opens.
6. From the **Standard** dropdown list, select **metafields**:
7. Click **Add Mapping**.
A success message appears on the popup window.
8. Click **Close**.
9. Locate your new mapping in the product mappings list and click **Special Mapping – Click to Edit** link on the mapping.
The metafields (AKA custom fields) Mapping popup window opens.
10. Enter the appropriate values in the **NetSuite Field**, **Metafield Name** and **Namespace** fields.
11. (Optional) To add multiple metafields, click the **Add Row** button and enter the corresponding field IDs.
12. Click **Save Changes**.
The popup window closes.
13. Click **Save**.

Tracking Shopify Item Inventory in NetSuite Connector

Before you can sync item inventory to Shopify, you must set Shopify to track inventory for those items. If you attempt to send inventory to Shopify for an item you have not set to track inventory, an error will result.

To set Shopify to track item inventory, you must map the `inventory_management` field for your Shopify products. This field corresponds to the Track Inventory box on your items in Shopify.

To set Shopify to internally track item inventory:

1. Log in to app.farapp.com.
 2. On the left panel, select the Shopify connector and relevant account.
 3. Go to **Mappings > Products**.
 4. On the Products Mappings page, click the button of the sync type you are using, either **Price/Quantity Only** or **Full Product**. The actual field mapping you create will be the same, you should ensure to create it in the fields for the sync you are using.
 5. Click **Add Mapping**.
 6. On the Add Shopify Field Mapping page, do the following:
 - a. In the **Standard** field, select **inventory_management**.
 - b. Click **Add Mapping**.

A confirmation message is displayed indicating that your mapping was added and the new mapping will appear at the end of mappings list. If your mapping list extends off the page, you may not see the mapping you just added.
 - c. Click **Close**.
7. On the row of the added mapping, in the first field under the Mapping Type column, select **Use Default**. Every time the item syncs, NetSuite Connector will, by default, apply the action you define on the next step.
 8. Go to the field under Default Value and select **shopify (track inventory on site)**. This sets Shopify to track inventory on items by default.
 9. (Optional) If you do not want to track inventory, skip steps 7 and 8 because this new mapping field is set not to track inventory by default. You can control whether or not Shopify tracks inventory for each individual item by mapping a field in NetSuite that contains the value you want to use for that item. To enable this, enter the NetSuite field ID for the field on the item that will contain that value in the NetSuite Field ID column and then skip the next step.
 10. Click **Save**.

Handling Shopify Cancel and Refund Emails in NetSuite Connector

Shopify allows you to send an email to the customer when an order is canceled or refunded. The email templates are stored in Shopify but the request to trigger the email is sent by NetSuite Connector. The request is part of the cancellation or refund request sent to Shopify.

If you have configured cancel or refund emails in Shopify but they are not sent when a refund or cancellation is synced, contact NetSuite Support. Request that this feature be enabled in your account..

Syncing Canceled Orders to Shopify Using NetSuite Connector

When you cancel an order in NetSuite, NetSuite Connector can sync that cancellation to Shopify, thereby canceling the order in Shopify. In NetSuite, a canceled order, is one of the following:

- An order pending approval that has been canceled.
- An order pending fulfillment that has been closed.

To sync NetSuite order cancellations to Shopify:

1. Log in to app.farapp.com.
2. Select the Shopify connector and relevant account.
3. Go to **Settings > Orders**.
4. Click **Advanced Options** and then check the **If An Order is Cancelled in NetSuite, Cancel it in Shopify** box.
A new setting is displayed.
5. By default, Shopify will automatically refund paid orders that are canceled. If you want to prevent this, check **Prevent Shopify From Refunding Paid Orders When Cancelling**.
6. Click **Save**.

Shopify Point of Sale (POS) Orders

By default, NetSuite Connector will post Point of Sale (POS) orders as cash sales to NetSuite. This is best practice because typically, orders from your POS do not require shipment or any shipping processes once they are completed by the POS platform. As cash sales in NetSuite directly impact the General Ledger and do not require any shipping or fulfillment data, this practice is used for efficiency. To change this default setting, you can contact the NetSuite Connector Support team.

POS orders can also be unique in that they may not always have an accompanying customer name and email address.

Assigning Default Values to Shopify Point-of-Sale (POS) Orders in NetSuite Connector

Shopify Point-of-Sale (POS) orders may not always have a customer name and email address. Use the following procedure to assign default values when these details are not provided in the POS orders:

To assign default values to Shopify POS orders:

1. Log in to app.farapp.com.
2. Select Shopify connector and then select the relevant account.
3. Go to Settings > Orders > General.
4. Click **Advanced Options**.
5. In the following fields, enter the default values:
 - Default Email
 - Default First Name
 - Default Last Name
6. Click **Save**.

Shopify Gift Cards

If you create a gift card product in Shopify and do not assign it an Stock-keeping Unit (SKU), Shopify automatically uses the SKU **WebOrderGiftCard**. You can then create a product in NetSuite with the SKU **WebOrderGiftCard**. NetSuite Connector automatically matches both SKUs and no further steps are needed.

If you are using a different SKU for Shopify gift card items, make sure you have a matching item in NetSuite set with that SKU value. Typically this is a non-fulfillable item. Do not use a gift certificate item in NetSuite as the item will not sync properly.

Shopify Product IDs

Shopify product IDs are unique item identifiers created by Shopify. Every item that is created in your Shopify account will have a product ID. You can locate an item's product ID by going to the product details page and it is the series of numbers in the last part of the URL.

After a product initially syncs from NetSuite to Shopify, NetSuite Connector will store its product ID. Subsequent postings of the product will automatically include the product ID in the mapping. In this way, NetSuite Connector can post the data to the correct SKU in Shopify. To display the product ID with mapped data, in NetSuite Connector, go to the Product page, and on the item, click **Show Product Mapping**. On the Product Mappings popup page, see the series of number in **product_id**.

If you ever delete an item directly in Shopify, you must also delete it in NetSuite Connector; otherwise, NetSuite Connector will try to match to a nonexistent product ID. This will cause a product posting error. By deleting an item in NetSuite Connector, the stored product ID will also be removed.

Using the Published Scope and Published Fields for Mapping in Shopify

The following publishing fields are provided for mapping:

- **Published** – Determines whether the product is available to the online store. Setting this field to **true** makes the item available to the online store.
- **Published Scope** – Determines whether the product is available to the Shopify POS (point of sale) channel.

The field has following options in the dropdown list:

- web (publish to web store but not point-of-sale)** – Makes the product available on all channels, except POS.
- global (publish to web store and point-of-sale)** – Makes the product available on all channels. If you do not make any selection in the list, it is considered as global.

For more information, read [Q:](#)

You can set the mapping for the field from the Product Mappings page of the Shopify connector.

For more information about these fields, read the Shopify documentation <https://help.shopify.com/en/manual/migrating-to-shopify/transporter-app/csv-products>.

Using Shopify's Compare At Feature in NetSuite Connector

In Shopify, you can the item price to sale price and keep the original price in the **Compare at price** field. This field lets your customers to easily see their savings on your Shopify items.



Note: The **Compare at price** field can only be mapped when using full product sync.

You should set the **Compare at price** field using the price levels in NetSuite and then syncing those prices to Shopify. The **Price** field should be mapped to the price level in NetSuite that contains the price at

which you wish to sell the item in Shopify. Whereas, the **Compare At** field should be mapped to the price level in NetSuite that contains the price at which the item is normally sold.

To set the Compare At pricing:

1. Log in to app.farapp.com.
2. Select the Shopify connector and the relevant account.
3. Go to Mappings > Products.
The Product Mappings page opens.
4. Click **Add Mapping**.
The Add Shopify Filed Mapping window opens.
5. From the **Standard** field, select **compare_at_price**.
6. Click **Add Mapping**.
The new mapping is added at the bottom of the Order Mappings page.
7. On the new mapping, click **Special Mapping – Click to Edit**.
The compare_at_price Mapping popup window opens.
8. In the **Price Level/Field** field, select the price level that you want to use.
9. (Optional) If you use quantity based price levels, check the **Using Quantity-Based Price Levels** box.
10. Click **Save Changes**.
11. Click **Save**.

Troubleshooting Shopify Sync Errors in NetSuite Connector

When you are working with Shopify product sync, you may see the following errors.

Variants have not been changed. The following IDs do not exist or do not belong to the product: <ID>

This error occurs when Shopify already has the variant item in the store, but under a different parent item. NetSuite Connector tries to match the item and adds the item as a variant to the new parent you specified in NetSuite. Then, NetSuite Connector checks the SKU in Shopify and does not know that the item is under another parent SKU. Therefore, NetSuite Connector tries to post the wrong ID. To resolve this error, delete the existing item from Shopify or adjust the item in NetSuite to match the variant item in Shopify.

Exceeded maximum number of variants allowed

This error means that the number of variants associated with the parent item exceeds the maximum limit of 100. This limit is imposed by Shopify.

For information about variants, read the Shopify help article <https://help.shopify.com/en/manual/products/variants/add-variants>.

To resolve this error, reduce the number of subitems in NetSuite to 100 or fewer subitems associated with the parent. For more information, read the help topic [Removing a Subitem From Your Item Matrix](#).

Resolving Location Deactivated Error in Shopify Product Sync

When the Shopify location that NetSuite Connector uses to sync inventory to is deactivated in Shopify, you get an error message upon product sync. The error is Posting errors: [u'Location is deactivated']. The solution to fix this error depends on whether the location was intentionally deactivated in Shopify or not.

Shopify Location Deactivation	Solution
Intentional	Update the mapping for the <code>inventory_quantity</code> field to remove the deactivated location. See Mapping Shopify Multilocation Inventory in NetSuite Connector .
Not Intentional	Reactivate the location in Shopify. NetSuite Connector cannot make changes to the status of an inventory location in Shopify.

Troubleshooting Unsupported Carriers Tracking Error for Shopify in NetSuite Connector

In Shopify, if you use a carrier that is unknown to Shopify, you can get an error when tracking the fulfillment. To resolve the error, create a custom mapping and map the custom field with NetSuite Connector's Carrier field.

To create a custom mapping for custom field containing unsupported carrier:

1. Log in to app.farapp.com.
2. Select Shopify connector and then select the relevant account.
3. Go to Mappings > Fulfillments.
4. Click **Add Mapping**.
The Add Shopify Mapping popup window opens.
5. From the **Shopify Field** dropdown list, select **Carrier**.
6. Click **Add Mapping**.
7. Click **Close**.
The mapping appears at the bottom of the Fulfillment Mappings page.
8. In the **NetSuite Fixed / Field Value** column of the new mapping, enter the custom field name that you are using for tracking the carrier.
9. Click **Save**.

Shopify Connector FAQ

To learn more about Shopify connector, review the questions and answers below.

Can NetSuite Connector map to the "Country/region of origin" or "Harmonized System Code" fields on my Shopify products during product sync?

Yes, you can map both **Country/region of origin** and **Harmonized System Code** fields for Shopify during full product sync. These fields can be mapped to **country_code_of_origin** and **harmonized_system_code** field IDs, respectively.

Can NetSuite Connector mark an order as paid in Shopify if the payment is captured outside Shopify?

No, Shopify does not provide the ability to update the financial status to paid through the API if the payment is captured outside Shopify.

Can I update my existing Shopify connector to point to a different shop?

No, you must not update your existing Shopify connector to point to a different shop. Attempting to do so results in errors and unpredictable behavior. You must add a new Shopify connector for a different shop.

Does deleting the content in the NetSuite field also delete the metafields content in Shopify?

No, NetSuite Connector can only add and update the Shopify metafields. You should manually delete the metafields content directly in Shopify.

Why are some of my Shopify orders importing with "Anonymous Customer" as the customer?

Amazon no longer provides the customer names and addresses for Shopify orders. If a user is importing Amazon orders into your Shopify account, you will see the customer listed as Anonymous Customer with only the city, state, and zip code without the address.

NetSuite Connector, by default posts these orders with **Anonymous Customer**. It then updates the address book for the customer with the city, state, and zip code, associated with the most recent posted order under that name. When posting orders to NetSuite, NetSuite Connector searches for an existing customer using a combination of email and name search. If you want NetSuite Connector to create a new customer for each order coming from Amazon as **Anonymous Customer**, contact NetSuite Connector Support.

Why does NetSuite Connector sync my Shopify wholesale prices as discounts?

The price of an item is not the wholesale price, but rather is the regular price discounted down to the wholesale amount. NetSuite Connector cannot use the discounted price for the item price on the wholesale orders, instead of showing the regular price with a discount. The Shopify API specification identifies the field that should contain the price of an item. All additional data on the order is read, stored, and processed depending on the field designated for the item price. The logic in Shopify and NetSuite Connector is based on which fields represent which data.

Using a field other than the Shopify designated field, without additional coding changes in NetSuite Connector, causes the following issues:

- Break some of the autovalidation checks in NetSuite Connector. These checks run between the storefront order totals and the NetSuite order totals when importing the order to NetSuite.
- May result in unintended consequences in future syncs that use the order data (refund sync, order corrections, or others)
- Affect future updates and enhancements to NetSuite Connector, which will be developed with the expectation that all data fields are used in the expected manner.

If you use a separate Shopify store where you grant access to only wholesale customers, then the pricing can remain the same for all customers. Therefore, the pricing is not discounted to a wholesale price. This method will allow Shopify to provide the data properly. Note that such an additional store will also require an additional Shopify connector to sync with NetSuite Connector, as each connector can have only a single store connected.



Note: Such additional store will also require an additional Shopify connector to sync with NetSuite Connector, because each connector can have only one store connected.

For more information about wholesale pricing in Shopify, see <https://help.shopify.com/en/manual/online-sales-channels/wholesale>.

Can I update my existing Shopify connector to point to a different Shop?

No, you must not update your existing Shopify connector to point to a different shop. Attempting to do so will result in errors and unpredictable behavior. You must add a new Shopify connector for a different shop.

Does deleting the content in the ERP field also delete the content the metafields in Shopify?

No, NetSuite Connector can only add and update the Shopify metafields. You should manually delete the metafields content directly in Shopify.

Can you add an item to every order by default? Will the sales order generated in NetSuite have whatever items were ordered on Shopify, plus the default item?

NetSuite Connector does not support adding items to orders. You can set up a workflow or script in NetSuite to handle such operations.

Can the customer assign the same customer record for every order but still have the shipping address import for each order from Shopify to the sales order?

Yes. For more information, read [Assigning a Fixed Customer to All Orders in NetSuite Connector](#).

Can you leave the payment method blank in NetSuite on every order so that after shipping and billing, an invoice is generated instead of cash sale?

Yes, you should only make sure that payment method is not required on your sales order form and all payment methods are mapped to **Not Mapped**. For more information about payment methods mappings, read [Mapping Order Payment Methods in NetSuite Connector](#).

If I do not have multi-location inventory, how are my syncs affected when the Shopify's Locations feature is turned on?

Enabling the Locations feature in Shopify does not affect NetSuite Connector syncs.

If you enabled multiple locations in Shopify, but you are only using a single location, then you need not change anything. If you enable multiple locations in Shopify and you are using product sync, then set different NetSuite locations to sync to different Shopify locations. For more information, read [Mapping Shopify Multilocation Inventory in NetSuite Connector](#).

For more information about Shopify locations, read <https://help.shopify.com/en/manual/locations>.

I have multiple Shopify stores with Shopify Plus. Can I use a single Shopify connector to access all my Shopify stores?

You need a separate Shopify connector account for every store you want to sync with NetSuite Connector. Although Shopify Plus provides convenient access to multiple stores in Shopify, the structure of the API let you connect only to a single store per connector.

How do I change the variant title field in Shopify?

You cannot set a value for the Variant Title field in Shopify. The Variant Title field is automatically determined by Shopify using the variant field names.

Can NetSuite Connector update incoming quantity or create transfer orders in Shopify?

Currently, NetSuite Connector does not support transfer orders in Shopify. Shopify currently does not support transfer orders in REST API. Therefore, NetSuite Connector cannot update incoming inventory quantities or create transfer orders in Shopify to determine incoming quantity.

Why is my product not showing up in all my Shopify channels?

NetSuite Connector accesses Shopify through a series of API calls. Shopify provides limited control of product channel visibility when adding products using the API.

Shopify has the following two options for taking product values:

- All channels
- All channels except Point-of-Sale

Values are sent to the **published_scope** field, where the value **global** refers to all channels, and **web** refers to all channels except the Point of Sale channel. If no value is sent to the **published_scope** field, then Shopify defaults to the equivalent of **global** or all channels.

If you do not see a product showing up on all your channels, contact Shopify support.

To set up the **published_scope** field mapping in NetSuite Connector, go to Shopify > Mappings > Products.

Why is NetSuite Connector refunding my Shopify order when the order cancellation is synced from NetSuite?

If you configure NetSuite Connector to sync NetSuite cancellations to Shopify, but do not want canceled paid orders to refund automatically, then configure NetSuite Connector accordingly. For more information, read [Syncing Canceled Orders to Shopify Using NetSuite Connector](#).

Managing the Walmart Connector in NetSuite Connector

This section discusses the following topics about Walmart connector:

- [Walmart State Tax Remittance](#)
- [Configuring NetSuite Connector for Walmart Two-Day Shipping Delivery Program](#)
- [Walmart Connector FAQ](#)

Walmart State Tax Remittance

As required by law, Walmart collects and remits state sales tax on all taxable marketplace sales shipped to certain states. You can find the specific states under the 'State-Specific Laws for Marketplace Facilitators' section of Walmart's [Sales Tax Collection Overview](#). By default, NetSuite Connector excludes tax from posting on orders on which Walmart has remitted tax.

Walmart does not provide any indication in the order data that they remitted tax, unlike Amazon. NetSuite Connector determines that Walmart has remitted tax by keeping an internal list of states where Walmart remits tax. If an order comes from one of those states, then NetSuite Connector does not send the tax for the order as long as the **Omit Remitted Tax When Posting Walmart Orders** is enabled.

NetSuite Connector then posts the order to NetSuite with a tax rate of 0.00, with **isTaxable** set to **false**. You can still view the tax amount in NetSuite Connector on the Orders page, by selecting **Edit** under the **Action** button on right of the order.

Configuring NetSuite Connector for Walmart Two-Day Shipping Delivery Program

If you are enrolled in Walmart's Two-Day delivery program, you can add mappings for the following Walmart product attributes in NetSuite Connector:

- twoDayShippingOverride
- twoDayShippingRegion
- twoDayShippingMidwestRegionStates
- twoDayShippingNortheastRegionStates
- twoDayShippingSouthRegionStates
- twoDayShippingWestRegionStates

To map product attributes for Walmart Two-Day Delivery Program:

1. Go to the [Walmart product mapper](#).
2. Click **Add Mapping**.
3. Select the Two-Day Shipping Override attribute that you want to map.
4. Enter the NetSuite field ID as reference for the mapping, or create a default mapping value.
5. Click **Save**.

Walmart Connector FAQ

To learn more about Walmart connector, review the questions and answers below.

[Does NetSuite Connector support Walmart Fulfillment Services \(WFS\)?](#)

NetSuite Connector can import WFS orders using the Walmart connector. Because these orders are already fulfilled, the leading practice is to sync the orders to NetSuite as cash sales. To set up WFS orders for your account, contact NetSuite Customer Support.

[Does NetSuite Connector support Walmart Marketplace Canada?](#)

No, currently NetSuite Connector only supports Walmart Marketplace US.

[Why am I getting NetSuite Connector emails about my Walmart credentials being invalid when they show as valid when I test them in NetSuite Connector?](#)

This behavior can occur if the permissions on the Walmart API key are not set up correctly. You must check the permissions on the API key that NetSuite Connector uses.

For more information, read [How to verify the permissions on the FarApp/NetSuite Connector's Walmart API key](#).

Managing the WooCommerce Connector in NetSuite Connector

This section discusses about WooCommerce connector:

- [Default Mappings for WooCommerce Connector](#)
- [Mapping WooCommerce Images in NetSuite Connector](#)
- [Troubleshooting Saving WooCommerce Credentials Failing with Invalid Signature Error](#)

Default Mappings for WooCommerce Connector

This section lists the default mappings for the WooCommerce connector product sync and order sync.

WooCommerce Product Sync Default Mappings

Storefront Field	NetSuite Field
attributes	custitem_wc_attrib#
categories	custitem_fa_wc_categories
description	custitem_fa_wc_description
enable_html_description	custitem_fa_wc_htmldsc
enable_html_short_description	custitem_fa_wc_htmlshrtdsc
images	storeDisplayImage.internalId
in_stock	custitem_fa_wc_instock
parent_id	parent.internalId
short_description	custitem_fa_wc_shortdescription
sku	itemid
title	custitem_fa_wc_title
weight	weight

WooCommerce Order Sync Default Mappings

Storefront Field	NetSuite Field
id	custbody_gokeyless_storefrontorder
email	email
first_name	attention

Storefront Field	NetSuite Field
last_name	attention
billing	billingAddress
shipping	shippingAddress
company	addressee
address_1	addr1
address_2	addr2
city	city
state	state
postcode	zip
country	country
phone	addrPhone
company	addressee
address_1	addr1
address_2	addr2
id	tranId

Mapping WooCommerce Images in NetSuite Connector

NetSuite Connector allows you to map product images in WooCommerce using full product sync.

To map WooCommerce images in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select WooCommerce connector and then select the relevant account.
3. Go to Mappings > Products.
4. Click **Full Product**.
5. Click **Add Mapping**.
The Add WooCommerce Field Mapping popup window opens.
6. From the Standard list, select **images**.
7. Click **Add Mapping**.
A success message appears on the popup window.
8. Click **Close**.
9. Locate your new mapping in the product mappings list. Under the Mapping Type column, select **Item Field**.
10. In the NetSuite Field ID column, enter the field ID of the NetSuite field containing your product image.
11. Click **Save**.

Troubleshooting Saving WooCommerce Credentials Failing with Invalid Signature Error

In NetSuite Connector, if clicking Save and Test Connection for your WooCommerce credentials fails with an Invalid signature error, you must check your WooCommerce Endpoint setting and ensure it starts with https://. For example, an endpoint of www.mysite.com will cause this error; but changing it to https://www.mysite.com, will successfully establish the connection.

Managing Third-Party Logistics (3PL) Connectors in NetSuite Connector

Using 3PL connectors, you can export orders from NetSuite to 3PLs. When the items are fulfilled in the 3PL, the 3PL connectors enables exporting the fulfillment data from the 3PL to NetSuite. This section discusses the following about 3PL connectors:

- [Third-Party Logistics \(3PL\) Order Statuses in NetSuite Connector](#)
- [Mapping Orders and Fulfillments for a 3PL Connector in NetSuite Connector](#)
- [Retrieving and Syncing an Order from NetSuite to 3PL Manually Using NetSuite Connector](#)
- [Creating Custom Filters for 3PLs in NetSuite Connector](#)
- [Syncing Item Fulfillments to a 3PL from NetSuite Connector](#)
- [Mapping 3PL Shipment Methods in NetSuite Connector](#)
- [Handling Partially Fulfilled Orders in NetSuite Connector](#)
- [NetSuite Connector Carrier Handling](#)
- [3PL Connectors FAQ](#)
- [Troubleshooting Order Import Issues for 3PL Connector in NetSuite Connector](#)
- [Troubleshooting 3PL Fulfillments Posting to NetSuite Without Tracking Numbers](#)

Third-Party Logistics (3PL) Order Statuses in NetSuite Connector

The following table lists the 3PL order statuses and their meanings:

Status	Description
Awaiting Retrieval from (3PL)	The order is pulled from NetSuite Connector and is waiting for the 3PL provider to retrieve the order.
Order Posted, Waiting for Shipment	The order is successfully imported into the 3PL provider.
Complete	The order is posted back to NetSuite from the 3PL provider.
Error Posting Shipment to NetSuite	There is an issue posting the fulfillment to NetSuite. Hovering over the order in NetSuite Connector displays the specific error.

Status	Description
Error Posting Order to 3PL	There is an issue posting the order to the 3PL provider. Hovering over the order in NetSuite Connector displays the specific error.

Order Statuses that Trigger NetSuite Connector to Import Orders from NetSuite and Post to the Third-Party Logistics (3PL) Provider

By default, orders in **Pending Fulfillment** state in NetSuite that have not been previously retrieved are retrieved by NetSuite Connector and synced to the 3PL provider. If you are using NetSuite's Pick, Pack, and Ship feature, you can set up NetSuite Connector to import the item fulfillments created for orders that have the **Packed** status. The ShipStation 3PL supports syncing from item fulfillments, but Amazon MCF (Multi-Channel Fulfillments) does not support syncing from item fulfillments.

Mapping Orders and Fulfillments for a 3PL Connector in NetSuite Connector

For order mappings, you can set up order mappings in the Orders Mappings page in NetSuite Connector. To access the Order Mappings page, in your 3PL connector's account, go to Mappings > Orders, and click the **Order** tab. NetSuite Connector also posts the standard order information such as order number, customer name, and shipping address.



Note: The data that you want to map must be available in the Third-party logistics (3PL) provider's API.

For fulfillments, you can map shipment methods from the Fulfillment Mappings page in NetSuite Connector. To access the Fulfillment Mappings page, in your 3PL connector's account, go to Mappings > Fulfillments. NetSuite Connector includes fulfillment data such as tracking number, carrier, shipping cost, and ship date when posting the order from the 3PL to NetSuite.

Retrieving and Syncing an Order from NetSuite to 3PL Manually Using NetSuite Connector

If you have a 3PL connected with NetSuite Connector and need to manually pull an order from NetSuite, follow this procedure.

To retrieve an order from NetSuite to sync to a 3PL manually:

1. Log in to app.farapp.com.
2. Select the connector and then select the relevant account.
3. Go to Data Flows > Orders.
4. In the **Search** field, enter the order number that you want to retrieve and press Enter.



Note: The order number is the transaction ID in NetSuite and not necessarily the order ID from the storefront.

After the search is completed, the blue banner at the top disappears.

5. To confirm that the order is imported, click **Search** on the order again.

Depending on the 3PL connector, the posting or retrieval of order may be different. For example, for ShipStation, NetSuite Connector does not pull or push orders into their system. Instead, ShipStation pulls orders from NetSuite Connector. So, if you want the orders to be pulled into ShipStation, you will need to trigger a sync directly from ShipStation.

6. For 3PLs other than ShipStation:

- a. Go to Data Flows > Manage Data Syncs.
- b. On the Data Syncs page, click the **Play** icon at the bottom of the **Order Sync** tile.

The order sync starts for the connector and pulls new orders from NetSuite and posts them to the 3PL.

Creating Custom Filters for 3PLs in NetSuite Connector

NetSuite Connector imports orders from NetSuite once the order status is updated to Pending Fulfillment or item fulfillment is in Packed status (if syncing pick tickets). The order or pick ticket is sent to the third-party logistics provider (3PL) for fulfillment. You can use the filters feature to adjust what NetSuite Connector will import and send to your 3PL provider with your set criteria.

There are two types of filters used for 3PL connectors:

- **Global Filters** – This filter controls what NetSuite Connector imports from NetSuite to sync and send to the logistics provider. This filter is available to all accounts with connectors for ShipStation and Amazon MCF.
- **Account Specific Filters** – This is only available for accounts that have more than one 3PL connector. This filter appears in addition to the Global filters. You can use this filter to control what syncs between NetSuite Connector and the third-party logistics provider. If this option is available, you can use the Global filter to select what orders to prevent from being sent to any logistics provider, and the Account Specific Filter to selectively restrict orders to go to specific providers.

Creating a Filter

The following describes the settings and steps to create a filter in NetSuite Connector:

To create a filter:

1. Log into app.farapp.com.
2. Select the third-party logistics provider in the navigation menu.
3. Select a relevant account.
4. Go to Settings > Orders.
5. Expand **Advanced Options**.
6. Click **Edit Order Filters (Global)**.
7. Click **Ok**.
8. Make adjustments to the Edit 3PL Order Filters (Beta) window:
 1. Select **Add Filter Row**.
 2. Select a field in the **Filter Field** column. If you have a custom field to use as a filter:
 - Select **Custom Field**.
 - Enter the custom field in the **Custom Field** field.

9. In the **Filter Operation** column, select a filter operation:
 - Is In List
 - Is Not In List
 - Equals
 - Does Not Equal
 - Is Empty
 - Is Not Empty
 - Contains
 - Does Not Contain
 - Starts With
10. In the **Filter Value** column, enter the value to validate corresponding to the conditions you set in the **Filter Operation** column.
11. Click **Save Changes**.
12. Click **Close**.
13. Click **Save**.



Important: If you have multiple filters active, all of the conditions should be satisfied before it passes the filter. If your order or pick ticket fails at least one filter, it will not be forwarded or synced with your logistics provider.

Syncing Item Fulfillments to a 3PL from NetSuite Connector

By default, NetSuite Connector syncs only sales orders to Third-party Logistics (3PL) providers. To sync item fulfillments, also known as pick tickets, if the connector supports the syncing, first you must make the following configuration changes:

- In NetSuite, enable the Pick, Pack, and Ship feature. For more information, read the help topic [Setting Up Pick, Pack, and Ship](#).
- In NetSuite Connector:
 - Check the **Pick/Pack Ship Setting Is Enabled in Your NetSuite** box. For more information, read [To enable the pick, pack, and ship feature in NetSuite Connector](#): If the box is not present in the connector, it means that syncing pick tickets is not supported for the connector.
 - Check the **Check Here If You Have Pack/Pack/Ship Enabled In Your NetSuite And Want To Export The Pick Tickets Instead of Orders to <Marketplace or cart name>** box in the Order Settings page of your 3PL connector.

Retrieving Item Fulfillments Into NetSuite Connector and Posting the Fulfillments to the 3PL

After completing the preceding configuration, follow these steps to retrieve the item fulfillments into NetSuite Connector and post the fulfillments to the 3PL.

[**To retrieve item fulfillments into NetSuite Connector and post the fulfillments to the 3PL**](#)

1. After NetSuite Connector sends the sales order into NetSuite, create item fulfillments for the items that you want to sync to your 3PL.

The item fulfillments are created with the status **Picked**.

2. Make appropriate adjustments to the item fulfillments and change the status of the fulfillments to **Packed**.

By default, NetSuite Connector considers item fulfillments for syncing only when their status is **Packed**.

NetSuite Connector's automated sync process retrieves the item fulfillments into NetSuite Connector and transmits them to the 3PL.

Mapping 3PL Shipment Methods in NetSuite Connector

The Fulfillment Settings page lets you view and edit all shipment mappings.

If you do not see a shipment method from your 3PL listed on this page, this means that NetSuite Connector has not received any fulfillments with that shipment method. You can only map shipment methods for fulfillments that NetSuite Connector has received.

If you have multiple accounts with your 3PL and multiple connectors in NetSuite Connector, you cannot map the shipping methods for secondary accounts.

To review or edit shipment methods on the Fulfillment Settings page:

1. In NetSuite Connector, click your 3PL in the left menu and select the appropriate account.
2. Go to Mappings > Fulfills.

A table is displayed with shipment methods on fulfillments that were imported to NetSuite Connector from the 3PL/account. Any orders that come in with new shipment methods are added to this table for mapping.

3. After mapping your 3PL shipping methods to your NetSuite shipping methods, click **Save**.

You can reload the shipment methods you have in NetSuite on the Fulfillment Settings page by clicking the icon beside Default Shipment Method To Post To When No Shipping Method Found. You can also click the same icon next to any shipment method listed on the page. Note that it takes a few seconds for the update to be displayed in NetSuite Connector.

If you enable Default Shipment Method To Post To When No Shipping Method Found to Do Not Post, NetSuite Connector generates an error for orders with an unmapped shipping methods. You can then map these, so that they can be posted to NetSuite.

4. (Optional) Under Advanced Options, you can filter orders by weight and total amount if necessary.

Handling Partially Fulfilled Orders in NetSuite Connector

To partially fulfill orders, enable the Pick, Pack, and Ship feature in NetSuite. To configure NetSuite Connector to sync the pick tickets, read [Syncing Item Fulfillments to a 3PL from NetSuite Connector](#).

When a sales order is partially fulfilled, an item fulfillment record is created in NetSuite. The initial status of the fulfillment is **Picked**. The fulfillment syncs to a third-party logistics (3PL) provider (if applicable) when the fulfillment is in **Packed** state. When the fulfillment is marked as **Shipped** in NetSuite, NetSuite Connector syncs the fulfillment back to the storefront. When the rest of the sales order is fulfilled, NetSuite creates another fulfillment. Again, when the fulfillment is in **Packed** state, it syncs to the 3PL provider, and when it changes to **Shipped** state, the fulfillment syncs to the storefront.

A sales order can have multiple item fulfillments, but the process of syncing them to a 3PL provider or to the storefront remains the same each time.

Troubleshooting Order Import Issues for 3PL Connector in NetSuite Connector

Check the following for order import issues from NetSuite for 3PL Connector:

- Check the order status. NetSuite Connector retrieves orders that have a **Pending Fulfillment** status in NetSuite and have not been retrieved and synced to the 3PL previously. If you are using NetSuite's Pick, Pack, and Ship feature and enable the sync pick tickets feature in the 3PL that supports it, NetSuite Connector imports orders in **Packed** status.
- Check if the **Skip Over NetSuite Orders with Background Line Items** setting is enabled in 3PL connector settings in NetSuite Connector. If this setting is enabled, orders with backordered items will not import.
- Check if you have any 3PL filters enabled which may be filtering certain orders based on the criteria that you have set. For more information about filters, read [Creating Custom Filters for 3PLs in NetSuite Connector](#).

Troubleshooting 3PL Fulfillments Posting to NetSuite Without Tracking Numbers

If the tracking numbers are not posting back to NetSuite on the shipments from your 3PL, the reason can be one of the following:

- You are using NetSuite's integrated shipping but NetSuite Connector is not configured to indicate that you are using NetSuite's integrated shipping.
- The carrier to the associated shipping method in the 3PL is not assigned properly.

Configuring NetSuite Connector When NetSuite is Configured for Integrated Shipping

Use the following procedure to configure shipping accounts in NetSuite Connector when NetSuite is configured for integrated shipping.

To configure shipping accounts in NetSuite Connector:

1. Log in to app.farapp.com.
2. Select NetSuite > Settings > General.
3. In the **Shipping Accounts** section, check the boxes of the shipping services that you have directly connected to NetSuite.
4. Click **Save**.

3PL Configuration for Shipping Accounts

Verify that any shipping method from a carrier linked to NetSuite also has the carrier on that method set properly in your 3PL. If a method does not have **UPS**, **USPS**, or **FedEx** set as the carrier, then NetSuite Connector does not know that the associated methods belong to that carrier. Therefore, NetSuite

Connector does not sync the tracking information where the linked carrier accounts are expecting the tracking information.

For example, if you have USPS linked to NetSuite, but in your 3PL the **Priority Mail** shipping method has a blank carrier field. Then the tracking number will not be synced to NetSuite.

3PL Connectors FAQ

Review the following questions and answers for Third-Party Logistics (3PL) connectors.

[How often does NetSuite Connector retrieve fulfillment data from a 3PL provider?](#)

NetSuite Connector makes an API call every 90 minutes to the 3PL provider and pulls fulfillment data from the orders marked as shipped into NetSuite Connector. The order then posts to NetSuite with the fulfillment data.

The ShipStation 3PL works differently. For more information, read [ShipStation Connector FAQ](#).

[When syncing orders to a shipping software or 3PL, from where does NetSuite Connector pull the shipping address?](#)

By default, NetSuite Connector pulls the shipping address from the **Ship To** field on the sales order in NetSuite.

Viewing the Custom Address Sent to the Shipping Software or 3PL

For custom address, use the following procedure to view the custom address that is sent to the shipping software or 3PL provider.

[To view the custom address Sent to the Shipping Software or 3PL](#)

1. On the sales order, click the **Shipping** subtab.
2. Click the **Edit** icon next to the **Ship to Select** field.

A popup window opens. The information in the individual address fields in the popup window is what NetSuite Connector sends as the address to the shipping software or 3PL provider.

[Does NetSuite Connector sync orders manually entered into NetSuite to the 3PL provider?](#)

Yes, NetSuite Connector syncs orders to the 3PL provider regardless of whether the order was entered manually or created using automated sync.

By default, NetSuite Connector checks for any order in a **Pending Fulfillment** status, regardless of the origin of the order. Orders with a **Pending Fulfillment** status are sent to the 3PL provider.

If you configure NetSuite Connector to sync pick tickets, then, by default NetSuite Connector checks for item fulfillments with a **Packed** status. When an item fulfillment is found with **Packed** status, the fulfillment is sent to the 3PL provider.

[Can I filter orders by location for my 3PL provider?](#)

You can filter the orders exported to 3PL based on location and other criteria. For more information, read [Creating Custom Filters for 3PLs in NetSuite Connector](#).

Can NetSuite Connector sync product data to a product record in a 3PL provider?

No, NetSuite Connector does not sync product data to a product record in a 3PL provider. NetSuite Connector only syncs data to an order in a 3PL provider. By default, NetSuite Connector syncs data from the sales order.

Managing the ShipStation Connector in NetSuite Connector

This section discusses the following about ShipStation connector:

- [Custom Store Connection with ShipStation Connector](#)
- [Fields that can be Mapped to and from ShipStation](#)
- [Resending Orders from ShipStation to NetSuite Connector](#)
- [Checking the Order Status in ShipStation Connector](#)
- [ShipStation Connector FAQ](#)

Custom Store Connection with ShipStation Connector

The connection setting on a ShipStation's custom store is built using a single listener URL. Each NetSuite Connector's ShipStation connector has a unique listener URL.

For example, if you have three ShipStation connector accounts named default, account2, and account3, the corresponding listener URLs will appear as following:

- **default** - https://ss.farapp.com/secure_ShipStationListener
- **account2** - https://ss.farapp.com/secure_ShipStationListener/Account2
- **account3** - https://ss.farapp.com/secure_ShipStationListener/Account3

In ShipStation, you would have one of the following:

- Three different custom stores, each using one of the URLs
- Three different ShipStation logins or accounts, where each has a single custom store using one of the URLs.

For more information about configuring a custom store in ShipStation, read the SuiteAnswers article [ShipStation Account Access](#).

Fields that can be Mapped to and from ShipStation

By default, NetSuite Connector maps all the required fields when orders are synced from NetSuite to ShipStation. More mappings can be added to sync additional data to other fields in ShipStation, if the data is available in NetSuite, and ShipStation makes the relevant field available through the current API implementation that NetSuite Connector uses.

NetSuite Connector accesses ShipStation through the Custom Store API implementation that ShipStation offers. When the ShipStation connector was developed, this was the only API implementation that ShipStation provided. Although additional API implementation were later offered by ShipStation, the Custom Store method remains the method that NetSuite Connector uses.

Order and fulfillment fields in ShipStation are documented at https://help.shipstation.com/hc/en-us/articles/360025856192-Custom-Store-Development-Guide#UUID-1bb7ab47-6cf4-c6cd-0e5b-dec61e91201b_UUID-2be5b452-acd6-9a90-0a1b-bcc4109c85bd. These fields can be mapped by NetSuite Connector using the Custom Store API. If the field you want to map is not listed in the documentation, then it is not possible to map it to or from ShipStation.

Note that the fields are listed in the Custom Store Reference Guide Section of the documentation. Fields that can be sent to ShipStation are in the Order Information Field Definitions section. Fields that can be received from ShipStation are in the ShipNotify Field Definitions section. You must know which fields are header level fields and which fields are line level fields, as the source and target fields need to exist at the same level for mappings.

If you need NetSuite Connector to update a mapping or map to a specific field, review the list in the documentation to ensure that the field is available in ShipStation. Order fields can be mapped through the ShipStation connector user interface from Mappings > Orders. The shipment method on returning fulfillments to NetSuite can be mapped, but any other field will require a support ticket with NetSuite Connector to identify the ID of the field you want to map from ShipStation and the ID of the target field in NetSuite.

For more information, contact NetSuite Customer Support.

Resending Orders from ShipStation to NetSuite Connector

NetSuite Connector cannot request to resend an order from ShipStation. However, you can resend an order from NetSuite Connector to ShipStation. Note the following points regarding resending orders:

- NetSuite Connector is set up as a custom store in ShipStation. ShipStation connects to NetSuite Connector and processes its orders in the same way in which NetSuite Connector processes them for a marketplace.
- ShipStation controls the flow of data as well as the drop off and pick up of orders on its own schedule. NetSuite Connector does not push or pull orders from ShipStation. When you resend an order from ShipStation to NetSuite Connector, the data transmission may be delayed.

For more information about resending an order, read the documentation on ShipStation's website.

Checking the Order Status in ShipStation Connector

The ShipStation connector displays the status of every order that NetSuite Connector pulls from NetSuite and sends to ShipStation. You can use this status to determine the status of fulfillment.

Following are the different order statuses that you can see for an order:

- **Error** – Indicates that there is an error in order processing. The most common error that you see is **Error Posting Shipment to NetSuite**. Hover the mouse over the error to see additional details.
- **Awaiting Retrieval from ShipStation** – Indicates that NetSuite Connector has pulled the sales order or item fulfillment (if you are syncing pick tickets) transactions and ShipStation can now do the pick up. NetSuite Connector cannot push orders to ShipStation, so it waits for ShipStation to do the pick up.
- **Order Posted** – Indicates that ShipStation has picked up the order information and NetSuite Connector is waiting for ShipStation to send the fulfillment information back. NetSuite Connector cannot pull orders from ShipStation.
- **Complete** – Indicates that NetSuite Connector has received the fulfillment information from ShipStation and has sent the information back to NetSuite. If the order originated in a marketplace or cart, this status does not indicate that the fulfillment has been successfully sent to the marketplace.

or cart. The **Order Complete** status in the connector determines that the fulfillment was sent to the marketplace or cart.

To check the order status in ShipStation connector:

1. Log in to app.farapp.com.
2. Select ShipStation connector
3. Select the relevant account.
4. Go to Data Flows > Orders.

The **Status** column on the order dashboard displays the status of the order. The order number that you see on the dashboard is usually the NetSuite sales order number but may vary on your configuration.

After ShipStation picks the order, you can log into ShipStation to view the order status. Note that ShipStation pulls and pushes orders to NetSuite Connector on its own schedule.

ShipStation Connector FAQ

Review the questions and answers below to learn more about ShipStation connector.

How long does it take for NetSuite Connector to bring my ShipStation order back to NetSuite?

ShipStation is not a 3PL, but a shipping software. Therefore, ShipStation picks up and drops off orders to and from NetSuite Connector rather than having NetSuite Connector push or pull orders from ShipStation. The frequency at which the data exchange occurs is determined by ShipStation. In the context of the custom store NetSuite Connector syncs to in ShipStation, NetSuite Connector is the marketplace. Therefore, when ShipStation notifies the marketplace of the shipment, it is notifying NetSuite Connector and providing the fulfillment data. Typically, the ShipStation custom store is configured to do that when you generate your shipping label but you can adjust that configuration in ShipStation. For more details, check the ShipStation documentation.

Can NetSuite Connector fulfill orders that were manually imported to ShipStation using a CSV file?

No. NetSuite Connector can only sync order fulfillments that used NetSuite Connector to sync to ShipStation.

For more information, read [Q:](#)

Instead of importing orders using CSV file to ShipStation, you can import the orders to NetSuite using CSV files. After the orders are in NetSuite, NetSuite Connector can sync those orders to ShipStation for fulfilling. Later, you can sync the fulfillment from ShipStation back to NetSuite.

Can I sync fulfillments for my other stores in ShipStation with NetSuite Connector?

No. NetSuite Connector can only sync order fulfillments that used NetSuite Connector to sync to ShipStation.

Can I sync the weight from ShipStation to my item fulfillments in NetSuite?

You cannot sync the weight from ShipStation to your item fulfillments in NetSuite. You will see a weight of 1 pound on the item fulfillment by default, unless overwritten by NetSuite. For more information, read [Fields that can be Mapped to and from ShipStation](#).

Can I use one ShipStation custom store in ShipStation to connect to multiple ShipStation connectors in NetSuite Connector?

No. You must use one ShipStation connector account for each custom store.

Can NetSuite Connector sync the Hold Until field to ShipStation with the orders?

NetSuite Connector integrates to ShipStation through the ShipStation custom store. Custom Store integration does not support the **Hold Until** field through API and NetSuite Connector is therefore unable to sync that field. For more information, read [Fields that can be Mapped to and from ShipStation](#).

Why is the order number in ShipStation different than the order number in NetSuite Connector?

If your NetSuite is configured to sync pick tickets, then you are using the item fulfillment record as the source of the ShipStation data. Therefore, the transaction ID from the item fulfillment in NetSuite is used as the order number in ShipStation.

Because multiple item fulfillments can be generated for a single sales order, using the same order number as that on the sales order could result in duplicate order numbers in ShipStation. Therefore, to avoid duplicate order numbers, NetSuite Connector uses the transaction ID for the order number in ShipStation.

Why are fulfillments not syncing from ShipStation back to NetSuite?

If an order is shipped in ShipStation but the fulfillment is not synced back to NetSuite, verify that ShipStation has sent a fulfillment notification to NetSuite Connector. Otherwise, resend the notification. ShipStation does this when it notifies the marketplace of the shipment. Refer to the ShipStation documentation to know how to configure ShipStation to notify the marketplace when orders are shipped, or to resend a marketplace notification.

Why do I have old or fulfilled orders in my ShipStation or ShipWorks connector?

NetSuite Connector pulls orders or pick tickets to the ShipStation Connector when those orders are in a certain state. When NetSuite Connector looks for those records in matching states, it checks for orders with a modified date that is more recent than the last check.

Following are the possible causes and their resolution for old or fulfilled orders in your connector:

- In a default configuration, many older orders in your connector means many of those orders are either of the following:
 - Waiting to be picked up by ShipStation
 - Picked up and not yet returned.

To fix this issue, verify the NetSuite Connector credentials in ShipStation configuration. You can validate the credentials by logging into NetSuite Connector with those credentials. Make sure you actually type the credentials and not use your browser's cached credentials.

- If you are using the NetSuite Connector corrections feature, then make sure you do not have an automation in NetSuite that is changing the older orders. When many orders appear with corrections, it usually means that some automation is updating those orders, causing the modified date on those orders to change. These changes to modified date cause NetSuite Connector to pull the orders back into ShipStation connector.

These orders are often already shipped and need not go to ShipStation. Also, ShipStation knows that these orders are already shipped.

As the number of orders for ShipStation grow, the speed at which ShipStation retrieves the orders slows. When this speed significantly slows can lead to other issues.

Review your NetSuite automations and make sure you do not unnecessarily save old orders when you are not actually changing them. For example, you write a workflow to change field X to value Y, but do not build in a condition to first see if Y is already set. Due to this issue, each time the workflow runs, all records in which the workflow looks are saved. If you must make some change to your orders, make sure you save the order only when the change is actually made.

If you either fulfill the old orders or do not want to fulfill those orders, you can cancel those orders in NetSuite Connector. For more information about cancelling orders, read [Deleting and Canceling Orders in NetSuite Connector](#).

How do I view the activity logs in ShipStation?

ShipStation lets you view the activity log of your NetSuite Connector custom store and individual orders. For more details, refer to the following links:

- To view the activity of a store in ShipStation, read the ShipStation support article – <https://help.shipstation.com/hc/en-us/articles/360026158431-Store-Activity-Log>
- To view the activity of an order in ShipStation, read the ShipStation support article – <https://help.shipstation.com/hc/en-us/articles/360025869072-Edit-Order-Details>

Troubleshooting Common Connector Issues

This section covers the troubleshooting of common connector issues.

Troubleshooting Storefront Connection Issues with NetSuite Connector

If your marketplace or cart disconnects from NetSuite Connector, no data will sync to or from the storefront.

For information about accessing your storefront, read [Integrating NetSuite Connector with NetSuite](#).

If your marketplace, cart, or server is behind a firewall, you may need to add NetSuite Connector's IP addresses to safelist. For more information, read [NetSuite Connector IP Addresses to Safelist \(Allowlist\)](#).

Troubleshooting Error Email About Invalid Credentials in NetSuite Connector

If you receive this error email, test your connector connection. If the test connection is successful, ignore the email. If the test connection is not successful, update the credentials for the connector. If you still continue to experience the issue, contact NetSuite Customer Support.

Testing Connector Connection

For connectors that have the option to test the connection, use the following procedure:

To test connector connection:

1. Log in to app.farapp.com.

2. Select the connector for which you got the error email.
3. Go to Settings > Credentials.
4. Click **Save and Test Connection**.

NetSuite Connector FAQ

Following are the links to FAQ sections of NetSuite Connector:

- [General NetSuite Connector FAQ](#)
- [NetSuite Connector Order Sync FAQ](#)
- [NetSuite Connector Refund Sync FAQ](#)
- [NetSuite Connector Product Sync FAQ](#)
- [Amazon Connector FAQ](#)
- [BigCommerce Connector FAQ](#)
- [eBay Connector FAQ](#)
- [Shopify Connector FAQ](#)
- [Walmart Connector FAQ](#)
- [3PL Connectors FAQ](#)
- [ShipStation Connector FAQ](#)

General NetSuite Connector FAQ

The following are some frequently asked questions for NetSuite Connector.

Do I need a NetSuite OneWorld account if I have international transactions?

NetSuite OneWorld is not required for international transactions. NetSuite Connector does not require OneWorld to sync order data but you need to set up the following in NetSuite for international orders:

- A tax nexus for each country or locality to which you ship orders.
- If you operate in other countries as different legal entities, you may need a subsidiary to represent each legal entity in NetSuite.

To check if you need OneWorld, you can conduct a series of manual transactions in NetSuite for a test accounting period. If you pass all the processes, procedures, and accounting in your test without OneWorld, then you probably do not need OneWorld.

Do you integrate with payment gateways such as Stripe, Square, Authorize.net, and PayPal?

Direct integration with a payment gateway is rarely required, so NetSuite Connector does not have any direct integrations. Almost all eCommerce systems such as Shopify, Magento, and eBay already integrate with payment gateways directly. Some marketplaces or carts let you only authorize payments through the payment gateway and do the payment capture on your own. Whereas, other marketplaces or carts require you to do both authorization and capture payment through their system. Such systems that let you to do your own payment capture provide the payment gateway transaction IDs as part of the order data. NetSuite Connector can import this data so that you can do the payment capture yourself. This task does not require direct integration from NetSuite Connector to payment gateway if your business system, like NetSuite, integrates with your payment gateway.

Does enabling SuiteTax in NetSuite require any changes to NetSuite Connector?

NetSuite Connector currently does not support SuiteTax. This feature is in development, and you will be notified when it becomes available. Until that time, do not enable SuiteTax in NetSuite. Enabling SuiteTax on NetSuite accounts connected with NetSuite Connector can result in irreversible configuration issues.

Does NetSuite Connector send email alerts when inventory is low?

NetSuite Connector does not send email alerts when your item quantities are low. The reason is, NetSuite Connector does not keep a record of your inventory and does not monitor your inventory levels. NetSuite Connector does not know if your inventory is considered low or out of stock. You should configure NetSuite or your marketplace or cart to get such email notifications.

Does NetSuite Connector support subscription orders?

No, NetSuite Connector does not support subscription orders, and storefronts must handle the subscription functionality. The order must be processed as a single order and placed at the set interval. NetSuite Connector only imports order data from the storefront. If the imported subscription order is one aggregated order instead of a single interval, NetSuite Connector does not parse the order to be placed at the set interval. The storefront should separate the order.

For example, assume that the subscription is \$10 monthly for a period of 12 months. If the storefront aggregates that subscription order into a single order of \$120, NetSuite Connector cannot separate that to be charged for \$10 every month. A separate order of \$10 must come every month in NetSuite Connector.

Many storefronts recommend to use a third-party app that handles the subscription functionality.

How does NetSuite Connector determine when to post a new customer or use an existing customer?

NetSuite Connector searches customers in NetSuite based on a combination of email and either name or company name of the customer on the order. The search criteria may vary depending on your account configuration.

If NetSuite Connector does not find a match using the customer information on the order, then it creates a new customer. If NetSuite Connector finds multiple customer records that match the customer information on the order, it matches to the oldest customer record by created date.

When a customer matches a name or company name, but not the email address, a new customer record is created. If the customer ID (entity ID) in NetSuite is composed of the first name and last name, NetSuite Connector appends the customer ID with the order number. The reason is that NetSuite requires all customer or entity IDs be unique.

For example, assume the customer name **Bob Smith** on order **17363** matches to an existing customer **Bob Smith** in NetSuite. However, the email address for the existing customer differs from the email address on the order. In this case, a new customer record is created with the name **Bob Smith [17363]**. This makes the customer or entity ID on the record unique.

How can I find my NetSuite role ID?

For NetSuite Connector, the default role name in NetSuite is NetSuite Connector Web Services. To know the role ID of the role, read the help topic [Internal IDs Associated with Roles](#).

I changed the design of my site. How will this affect NetSuite Connector?

Cosmetic changes to the website layout should not cause any issues with your integration. However, other changes such as server, operating system, database, or network may impact your sync depending on the change and specific settings and mappings. You should sync a test order to determine if the change had an impact.

Can I send files or data from files to the storefront?

No. There is no functionality to send unrelated or random data during the sync.

NetSuite Connector only sends the data that appears on the records used as the source record for the associated sync.

[When do I turn off my existing sync service so I can use NetSuite Connector?](#)

This is common when you have another service that is syncing items or orders and you want to integrate NetSuite Connector service.

You should keep your existing service running during the NetSuite Connector integration. For more information, read [Switching to NetSuite Connector from an Existing Integration](#).

[When I log in to NetSuite Connector, the Profile page in my account settings shows inaccurate details.](#)

The primary user is the only user to which the profile applies. Other users do not have profiles. Therefore, each user will see the same information in the Profile page irrespective of the user ID you used to log in.

Usually, the person who opens the NetSuite Connector account provides the information in the Profile page. If your user ID has the required permissions, you can edit the information for accuracy. For more details, read [Changing the Primary User Information in NetSuite Connector](#).

[Why can I not change the account name of a connector after creating it?](#)

The account name is closely integrated with each connector you create in NetSuite Connector. You cannot change the account name to avoid breaking the integration.

NetSuite Connector creates a **default** account for each storefront.

You can edit the **Nickname** field to create a custom name for your connectors. Changing the nickname does not have any adverse effects on your integration. For more information, read [Updating the Connector Account Nickname in NetSuite Connector](#).

[Why NetSuite Connector does not post credit card numbers?](#)

NetSuite Connector does not import credit card numbers. Instead, NetSuite Connector imports credit card tokenization details by default to maintain Payment Card Industry (PCI) compliance. Most storefronts do not support capturing full payment details. However, if a storefront supports capturing full payment details and NetSuite Connector imports this data to NetSuite, that would be a breach of PCI compliance.

[Can I map to a custom record in NetSuite?](#)

Currently, NetSuite Connector does not support mapping to a custom record type in NetSuite.

[Can I manually change the items on an order in NetSuite or in 3PL?](#)

You can change items on your orders in NetSuite, but that will disable partial fulfillments. The reason is that NetSuite does not know for which item you are making the substitution. When the order is completely fulfilled, NetSuite Connector communicates to the storefront that the items it has on the order were fulfilled, regardless of what you actually fulfilled. The reason is that if NetSuite Connector tries to indicate other items were fulfilled, the storefront will throw an error.

If you do not need partial fulfillments to sync, change the items on the synced orders in NetSuite. However, make sure those changed items will not reflect in the storefront.

[Can I make syncs run at a specific time?](#)

Consider the scenario where the sync runs every 60 minutes and you want to run it at a specific time during that interval. For example, at the beginning of the hour. In this case, first time you should manually run the sync at the start of the hour.

After that, the sync should run 60 minutes from when you triggered the manual sync. For example, if you manually start the sync at 12:01, the next sync should run roughly at 1:01, and so on.

For more information, read [Manually Triggering a Scheduled Sync in NetSuite Connector](#).

There may be small variations that occur between runs, which can add up and push back or delay the sync. Eventually, the sync may not run at the desired time. Therefore, you may need to revisit the sync periodically and manually trigger the sync at the desired time again.

[Can I disable user credentials on the NetSuite Connector integration record?](#)

If NetSuite Connector is using token-based authentication to connect to NetSuite, you can safely disable user credentials on the NetSuite Connector integration record.

[Can I connect my NetSuite Connector account to more than one NetSuite accounts or change the NetSuite account to which the NetSuite Connector is connected?](#)

A single NetSuite Connector account can only connect to a single NetSuite account.

Sometimes, you may need to test new workflows in your sandbox or release preview accounts. However, after your NetSuite Connector integration goes live, you should not change the NetSuite account to which your NetSuite Connector account is connected.

To use your nonproduction NetSuite account to test with NetSuite Connector, contact NetSuite Sales.

[Can I connect both NetSuite production and sandbox accounts to the same NetSuite Connector account?](#)

No, you cannot connect both NetSuite accounts to the same NetSuite Connector account. You can connect only one NetSuite account to a NetSuite Connector account at a time. The NetSuite sandbox account requires you to create a NetSuite Connector sandbox account.

[Can NetSuite Connector upload or download CSV files for products, orders, or mappings?](#)

NetSuite Connector does not provide the CSV file upload functionality, but NetSuite does. Upload data directly into NetSuite.

NetSuite Connector only exports sync errors to a CSV file. You can export the errors from the order sync or product sync dashboards.

[Can NetSuite Connector sync customers to NetSuite without an order?](#)

No, NetSuite Connector does not sync customers to NetSuite without an order.

[Why is the tracking link not working?](#)

This problem usually occurs when the carrier on the item fulfillment is different from the carrier that actually shipped the order and provided the tracking number. To fix this problem, make sure the carrier on the item fulfillment is the carrier that provided the tracking number.

The tracking link is automatically generated by placing the tracking number into a link template determined by the carrier. When you place a FedEx tracking number into a link template for a FedEx carrier, the link works. However, if you place a UPS tracking number into a link template for a FedEx carrier, the link is invalid.