SuiteTalk REST Web Services Records Guide

2025.1

September 3, 2025

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.

Table of Contents

REST Web Services Supported Records

The following records are supported in REST web services.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Communication Records

- Event
- Issue
- Message
- Note Type
- Phone Call
- Task

Entity Records

- Competitor
- Contact
- Contact Category
- Contact Role
- Customer
- Employee
- Partner
- Vendor Category

HCM records

List records

- Bin
- Merchandise Hierarchy Level
- Merchandise Hierarchy Node
- Merchandise Hierarchy Version
- Unit of Measure

ERP Records

- Account
- Accounting Period
- Advanced Intercompany Journal Entry
- Assembly Build
- Assembly Unbuild
- Billing Revenue Event
- Bin Transfer

- Blanket Purchase Order
- BOM
- BOM Revision
- Class
- Cost Category
- Customer Category
- Customer Deposit
- Customer Message
- Customer Payment
- Customer Refund
- Customer Status
- Customer Subsidiary Relationship
- Department
- Deposit
- Deposit Application
- Description Item
- Discount Item
- Download Item
- Expense Category
- Expense Report
- Fair Value Price
- Fulfillment Request
- Gift Certificate Item
- Inbound Shipment
- Intercompany Journal Entry
- Intercompany Transfer Order
- Inventory Adjustment
- Inventory Cost Revaluation
- Inventory Count
- Inventory Item
- Inventory Number
- Inventory Transfer
- Item Fulfillment
- Item Group
- Item Receipt
- Item Revision
- Job
- Job Status
- Job Type
- Journal Entry
- Kit Item

- Location
- Manufacturing Cost Template
- Manufacturing Operation Task
- Manufacturing Routing
- Nexus
- Non-Inventory Purchase Item
- Non-Inventory Sale Item
- Non-Inventory Resale Item
- Other Charge for Purchase Item
- Other Charge for Resale Item
- Other Charge for Sale Item
- Other Name
- Other Name Category
- Paycheck
- Payment Item
- Payment Method
- Project Task
- Purchase Contract
- Purchase Order
- Requisition
- Revenue Recognition Schedule
- Revenue Recognition Template
- Sales Role
- Sales Tax Item
- Service Purchase Item
- Service Resale Item
- Service Sale Item
- Ship Item
- Statistical Journal Entry
- Subsidiary
- Subtotal Item
- Support Case
- Tax Type
- Time Bill (Track Time)
- Topic
- Transfer Order
- Vendor
- Vendor Bill
- Vendor Credit
- Vendor Payment

- Vendor Prepayment
- Vendor Prepayment Application
- Vendor Return Authorization
- Vendor Subsidiary Relationship
- Weekly Timesheet
- Work Order
- Work Order Close
- Work Order Completion
- Work Order Issue

Marketing Records

- Campaign
- Campaign Response
- Coupon Code
- Email Template
- Promotion Code

SuiteBilling Records

- Assembly Item
- Billing Account
- Billing Schedule
- Change Order
- Charge
- Price Book
- Price Plan
- Pricing Group
- Subscription
- Subscription Line
- Subscription Plan
- Subscription Term
- Usage

Pricing Records

- Cash Refund
- Cash Sale
- Check
- Consolidated Exchange Rate
- Credit Card Charge
- Credit Card Refund
- Credit Memo
- Currency
- Currency Rate

- Estimate
- Invoice
- Markup Item
- Opportunity
- Price Level
- Return Authorization
- Sales Order

Website Records

- Commerce Category
- Website

Account

An account is a record in your general ledger that is used to store transactions. Examples of accounts include accounts payable, cash, or cost of goods sold. Accounts are collected in your Chart of Accounts, which provides a set of destinations for posting transactions, and categorizes these transactions for tracking and reporting purposes.

Accounts have account types, which are used to organize data in account registers and other financial reports.

- Income statement account types include Cost of Goods Sold, Expense, Income, Other Expense, and Other Income.
- Balance sheet account types include Accounts Payable, Accounts Receivable, Bank, Credit Card, Deferred Expense, Deferred Revenue, Equity, Fixed Asset, Long Term Liability, Other Asset, Other Current Asset, Other Current Liability, and Unbilled Receivable.
- Cash flow statement account types include the net of Income and Other Income, Accounts Payable, Accounts Receivable, Equity, Fixed Asset, Long Term Liability, Other Asset, Other Current Asset, Other Current Liability, and Unbilled Receivable.
- Statistical accounts, part of the Advanced Financial module, enable your financial team to track non-monetary data and then use that information about reports and income statements. Financial users can examine the non-monetary data to view its relationship with the financial activity of your organization. For more information, see the help topic Using Statistical Accounts.

For more information, see the help topic Chart of Accounts Management.

The REST API Browser includes information about the field names and field types of the account record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's account reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an account REST record is **account**.

The account record has the following subrecords:

- accountcontextsearch

- localizations

Inaccessible Elements

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "acctName": "Test Account333 1692257016", "acctType": { "id": "OthCurrAsset" }, "cashFlowRate": { "id": "AVERAGE" }, "descrip
tion": "description text", "eliminate": false, "generalRate": { "id": "CURRENT" }, "includeChildren": false, "inventory":
 true, "isInactive": true, "isSummary": true, "revalue": true, "subsidiary": { "items": [ { "id": 2 }, { "id": 4 } ] } }
```

Accounting Period

An accounting period record exposes an accounting period to REST web services. This record has the following subrecords:

- fiscalcalendars
- perbookperiodclosing

Accounting periods support general ledger management. When accounting periods are enabled, users can select a past or future posting period for each transaction, and accounting personnel can more effectively monitor and control the status of accounts.

To access this record in NetSuite, go to Setup > Accounting > Manage G/L > Manage Accounting Periods.

For more information about managing accounting periods in NetSuite, see the help topic Manage Accounting Periods Page.

The REST API Browser includes information about the field names and field types of the accounting period record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's accountingPeriod reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an accounting period REST record is **accountingperiod**.

Inaccessible Elements

The following sublist is not accessible through REST web services:

- User Notes

Elements with Different Functionality

The following fields only appear in the base accounting period type:

- allowNonGLChanges
- isAdjust

The following sublist only appears in the base accounting period type:

- fiscalCalendars

Note: The accounting period is a base period if isYear = "false" and isQuarter = "false".

The following field only appears in the quarter and base accounting period types:

- parent

Note: The accounting period is a quarter period if isQuarter = "True" and isYear = "false".

Additional Details

You must have a parent period to set up a new quarter or base period.

Code Sample

This sample shows how to create a new base accounting period:

```
POST /services/rest/record/v1/accountingperiod { "allLocked": false, "allowNonGLChanges": false, "apLocked": false, "arLocked": false, "closed": false, "endDate": "2051-01-31", "parent": "546", "isAdjust": false, "isInactive": false, "isPosting": true, "isQuarter": false, "isYear": false, "payrollLocked": false, "periodName": "January 2051 1692277896", "startDate": "2051-01-01"
}
```

Advanced Intercompany Journal Entry

An advanced intercompany journal entry subrecord exposes an advanced intercompany journal entry to REST web services.

The advanced intercompany journal entry is a subrecord.

An advanced intercompany journal entry records debits and credits to be posted to ledger accounts for transactions between an originating subsidiary and multiple receiving subsidiaries. In an account that has the Multi-Book Accounting feature enabled, you can also use this record type to create book specific advanced intercompany journal entries.

To access this record in NetSuite, go to Transactions > Financial > Make Advanced Intercompany Journal Entries.

For more information about the advanced intercompany journal entry record, see the help topic Making Advanced Intercompany Journal Entries.

The REST API Browser includes information about the field names and field types of the advanced intercompany journal entry record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's advIntercompanyJournalEntry reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an advanced intercompany journal entry REST record is **AdvIntercompanyJournalEntry**.

Inaccessible Elements

The following fields are not accessible through REST web services:

- book
- details
- detailsUrl
- line
- message
- messageType
- expensePlanMessage
- expensePlanMessageExists
- expensePlanStatus

Elements with Different Functionality

The following fields have different functionality through REST web services:

- expensePlanStatus (Read only)
- tosubsidiaries (Read only)

Usage Notes

The performautobalance field is only available in metadata and schema.

Code Sample

This sample shows how to retrieve an advanced intercompany journal entry record by ID:

```
GET {{REST_SERVICES}}/record/v1/AdvIntercompanyJournalEntry/1107?expandSubResources=true
```

Assembly Build

An assembly build record exposes an assembly build to REST web services.

This record:

- is not a subrecord
- has one subrecords: component

The REST API Browser includes information about the field names and field types of the assembly build record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's assemblyBuild reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an assembly build REST record is **assemblyBuild**.

Code Samples

The following samples show common use cases for assembly builds. The example ID is 4.

Creating an Assembly Build Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyBuild
{ "subsidiary": { "id": "5" }, "location": { "id": "5" }, "item": { "id": "67" }, "quantity": 10.0
}
```

Retrieving an Assembly Build Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyBuild/4
```

The following code sample shows how to retrieve an assembly build component. The component ID is 7.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyBuild/4/component/lineNumber=7
```

Updating an Assembly Build Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyBuild/4
{ "quantity": "20"
}
```

The following code sample shows how to update an assembly build component. The component ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyBuild/4/component/lineNumber=7/
{ "quantity": "21"
}
```

The following code sample shows how to convert an assembly build to an assembly unbuild.

```
POST {{COMPANY_URL}}/services/rest/record/v1/assemblyBuild/4/!transform/assemblyUnbuild
```

Assembly Item

An assembly item is an inventory item made up of several components but identified as a single item. Assemblies are manufactured by combining raw materials that you stock.

For more information, see the help topic Assembly Items.

This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the assembly item record, and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's assembly item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the assembly item REST record is **assemblyItem**.

Prerequisites

You must enable the Assemblies feature before you can use this record through REST web services.

Inaccessible Elements

The following sublists are not accessible through REST web services:

- Quantity Pricing Discount/Levels (quantitypricingdiscount, quantitypricinglevel)
- Item Translations subtab (translations)

Elements with Different Functionality

The following sublists have different functionality through REST web services:

Manufacturing > Bills of Materials > Master Default field is not controllable through REST.

Code Samples

Get Record Metadata

```
GET https://<HOST>/services/rest/record/v1/metadata-catalog/assemblyItemHeader: Accept: application/swagger+json
```

Get Record

```
GET https://<HOST>/services/rest/record/v1/assemblyItem/<ID>?expandSubResources=true
```

Update Record

```
PATCH
https://<HOST>/services/rest/record/v1/assemblyItem/<ID>
```

```
Header: Content-Type: application/json
Body:
 {
       "itemId": "Updated item name",
      "upcCode: "123456"
}
```

Update Item Price when all Related Features are Enabled

```
PATCH https://<HOST>/services/rest/record/v1/assemblyItem/<ID>/price/currencypage=1,pricelevel=1,quantity=10
Header: Content-Type: application/json
Body:
{
    "price": 10
}
```

Update Bill of Materials/Default for Location

```
PATCH https://<HOST>/services/rest/record/v1/assemblyItem/<ID>
Header: Content-Type: application/json
Body:
{ "billOfMaterials": { "items": [ { "billOfMaterials": { "id": 14 } "defaultForLocation": { "items": [ { "id": 5 }, { "id": 6
 } ] ] ] ] ] ]
```

Assembly Unbuild

An assembly unbuild record exposes an assembly unbuild to REST web services.

This record:

- is not a subrecord
- has one subrecord: component

The REST API Browser includes information about the field names and field types of the assembly unbuild record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's assemblyUnbuild reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an assembly unbuild REST record is **assemblyUnbuild**.

Code Samples

The following samples show common use cases for assembly unbuilds. The example ID is 4.

Creating an Assembly Unbuild Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyUnbuild
{ "subsidiary": { "id": "5" }, "location": { "id": "5" }, "item": { "id": "67" }, "quantity": 10.0
```

```
}
```

Retrieving an Assembly Unbuild Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyUnbuild/4
```

The following code sample shows how to retrieve an assembly unbuild component. The component ID is 7.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyUnbuild/4/component/lineNumber=7
```

Updating an Assembly Unbuild Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyUnbuild/4
{ "quantity": "20"
}
```

The following code sample shows how to update an assembly unbuild component. The component ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/assemblyUnbuild/4/component/lineNumber=7/
{ "quantity": "21"
}
```

Billing Account

A billing account record exposes a SuiteBilling account to REST web services. You can enable Billing Accounts without enabling SuiteBilling.

This record:

- is not a subrecord
- has no subrecords

All elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the billing account record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's billing account reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the billing account REST record is **billingaccount**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Limitations

A billing account REST record does not show related records.

Code Sample

```
{ "startDate": "2019-2-20", "customer": {"id" : 1}, "billingSchedule": {"id": 1} "currency": {"id" : 1}
}
```

Billing Revenue Event

The billing revenue event record is available only when the Advanced Revenue Management (Essentials) feature is enabled. In the UI, you access this record at Lists > Accounting > Revenue Recognition Events > New (Administrator).

Billing revenue events trigger the creation of revenue recognition plans. Each event must have an event type. Event types are not scriptable. To create a revenue recognition event type, go to Lists > Accounting > Revenue Recognition Events > New (Administrator). You must create at least one event type before you can create a billing revenue event.

For help working with this record in the UI, see the help topic Creating a Custom Revenue Recognition Event.

The REST API Browser includes information about the field names and field types of the Billing Revenue Event record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's billingRevenueEvent reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a billing revenue event REST record is **BillingRevenueEvent**.

Prerequisites

This record is available only when the Product Management and Subscription Billing features are enabled. For more information about how to enable these features, see the help topics Enabling Project Features and Enabling SuiteBilling Features.

Limitations

The following fields are read-only for this record in REST:

- eventPurpose (required field)
- eventType (required field)
- projectRevenueRule
- subscriptionLine
- transactionLine

The eventDate field is also a required field.

Additional Information

The **quantity** and **amount** fields appear when the **eventType** field is not set as percent complete. The **cumulativePercentComplete** field appears when **eventType** is set to percent complete.

Code Samples

The following example shows how to read a billing revenue event:

Read:

```
GET {{REST_SERVICES}}/record/v1/BillingRevenueEvent/102?expandSubResources=true
```

Billing Schedule

The billing schedule record exposes a billing schedule to REST web services. This record:

- is not a subrecord
- has no subrecords

This record enables you to create a billing schedule that can be applied to a sales order, a line item on a sales order, or a project. For details about this type of transaction, see the help topic Billing Schedules.

You must enable the Advanced Billing feature before you can use this record through REST web services.

The REST API Browser includes information about the field names and field types of the billing schedule record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's billing schedule reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a billing schedule REST record is **BillingSchedule**.

Code Samples

These samples show common use cases for billing schedule.

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/BillingSchedule/
{ "frequency": { "id": "QUARTERLY", "refName": "Quarterly" }, "inArrears": true, "initialAmount": 100.0, "name": "NetSuite
 Test", "numberRemaining": 4, "repeatEvery": { "id": "1", "refName": "1" }, "scheduleType": { "id": "STD", "refName": "Standard" }
}
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/BillingSchedule/106
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/BillingSchedule/106
{ "frequency": { "id": "QUARTERLY", "refName": "Quarterly" }, "inArrears": true, "initialAmount":50.0, "name": "NetSuite
 Test", "numberRemaining": 2, "repeatEvery": { "id": "1", "refName": "1" }, "scheduleType": { "id": "STD", "refName": "Standard" }
}
```

```
DELETE: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/billingSchedule/106
```

Bin

A bin records enable you to organize and track items and their on-hand quantities within a location. This record is available when the Bin Management feature is enabled. For more information about bins, see the help topic Bin Management

The REST API Browser includes information about the field names and field types of the bin record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's bin reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a bin REST record is **bin**.

Bin Transfer

A bin transfer transaction records changes in bin details of inventory items transferred between bins within a location. This record is available when the Bin Management feature is enabled. This record has the following subrecord: inventory detail. For more details about this type of transaction, see the help topic Bin Transfers.

The REST API Browser includes information about the field names and field types of the bin transfer record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's bin transfer reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a bin transfer REST record is **bintransfer**.

The record ID for the inventory detail REST subrecord is **inventorydetail**.

Prerequisites

To transfer lot and serialized items between bins, you must enable the Advanced Bin/Numbered Inventory Management feature.

Code Samples

The following example shows how to create a bin transfer:

```
POST {{REST_SERVICES}}/record/v1/bintransfer
{ "location": { "refName": "WMS Warehouse 1" }, "memo": "newdmemo", "tranId": "BT-1234-new", "inventory": { "items": [ { "descrip
tion": "BinAssembly", "inventoryDetail": { "customForm": { "id": "-10820", "refName": "Standard Inventory Detail Form"
```

```
 }, "ignoreQtyValidation": false, "inventoryAssignment": { "items": [ { "binNumber": { "id": "14", "refName": "W1-InStage5"
 }, "internalId": 2061, "inventoryDetail": 114, "inventoryStatus": { "id": "1", "refName": "Good" }, "quantity": 5.0, "toBin
Number": { "id": "13", "refName": "W1-IN-InStage4" }, "toInventoryStatus": { "id": "1", "refName": "Good" } } ] }, "item":
 { "id": "40", "refName": "BinAssembly" }, "itemDescription": "BinAssembly", "location": { "id": "1", "refName": "WMS Warehouse
 1" }, "quantity": 5.0, "toLocation": { "id": "-1", "refName": "-1" } }, "item": { "id": "40", "refName": "BinAssembly"
 }, "line": 1, "quantity": 5.0 } ] }
}
```


Blanket Purchase Order

With this record, you can take advantage of fixed pricing for a preset number of items that you will buy during a specific time period. This approach lets you avoid sporadic pricing negotiations with vendors.

This record is available only when the Blanket Purchase Order feature is enabled at Setup > Company > Setup Tasks > Enable Features (Administrator), on the Transactions subtab.

In the UI, you access this record at Vendor Dashboard > Transactions > Enter Blanket Purchase Order (Administrator). For help working with this record in the UI, see the help topic Blanket Purchase Order.

The **item** element on this record is not accessible through REST web services.

The blanket purchase order record type includes the following actions from the Actions Framework:

- New
- Make Copy
- Make Copy without Schedules
- Email

The REST API Browser includes information about the field names and field types of the blanket purchase order record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's blanketPurchaseOrder reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a blanket purchase order REST record is **blanketPurchaseOrder**.

Prerequisites

This record is available only when the Blanket Purchase Order feature is enabled at Setup > Company > Setup Tasks > Enable Features (Administrator), on the Transactions subtab.

Code Samples

The following examples show how to read, create, or update a blanket purchase order:

Read:

```
GET {{REST_SERVICES}}/record/v1/blanketPurchaseOrder/${transaction-id}
```

Create:

```
POST {{REST_SERVICES}}/record/v1/ blanketPurchaseOrder { "entity": { "id": ${entity-id} }, "item": { "items": [{ "item": { "id":
 ${item-id} }, "rate": 10, "quantity": 4 }] }, "memo": "Memo text" }
```

Update:

```
PATCH {{REST_SERVICES}}/record/v1/ blanketPurchaseOrder /${transaction-id} { "memo": "Memo text updated", "item": { "items":
 [{ "item": { "id": ${item-id} }, "rate": 10 }] }
```

BOM

A Bill of Materials, or BOM, record exposes a Bill of Materials, or BOM, to REST web services.

This record:

- is not a subrecord
- has one subrecord: assembly

The REST API Browser includes information about the field names and field types of the BOM record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's bom reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a BOM REST record is **bom**.

Prerequisites

You must enable the Advanced Bill of Materials feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for BOMs. The example ID is 4.

Creating a BOM Using a POST Request

```
POST
https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bom
{
   "name": "bom rest 1",
   "availableForAllAssemblies": true,
   "availableForAllLocations": true,
   "subsidiary": {
       "items": [
           {
               "id": "5"
           }
       ]
   }
}
```

Retrieving a BOM Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bom/4
```

Updating a BOM Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bom/4
{
   "assembly": {
       "items": [
           {
              "assembly": {"id": 115},
              "masterDefault": true
           },
           {
                "assembly": {"id": 67},
                "defaultforlocation":  {
                    "items": [
                        {
                            "id": "5"
                        },
                        {
                            "id": 8
                        }
                    ]
                }
           }
       ]
   }
}
```

BOM Revision

A BOM revision record exposes a BOM revision to REST web services.

This record:

- is not a subrecord
- has one subrecord: component

The REST API Browser includes information about the field names and field types of the BOM revision record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's bomRevision reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a BOM revision REST record is **bomRevision**.

Prerequisites

You must enable the Advanced Bill of Materials feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for BOM revisions. The example ID is 4.

Creating a BOM Revision with a Component Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bomRevision
{
   "name": "Revision rest 1",
   "billOfMaterials": {"id": 78},
   "component": {
       "items": [
           {
               "item": {"id": 53},
               "bomQuantity": 1
           }
       ]
   }
}
```

Retrieving a BOM Revision Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bomRevision
/4
```

Updating a BOM Revision Using a PATCH Request

The following code sample shows how to edit the quantity on a component line. The component line ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/bomRevision/4/component/7
{
    "bomQuantity": 5
}
```

Campaign

The campaign record exposes a campaign to REST web services.

This record:

- is not a subrecord
- contains the following subrecords:
  - campaignEmail
  - campaignDrip
  - campaignDirectMail
  - campaignEvent
  - defaultEvent (read-only)

For more information about working with campaigns in the UI, see the help topic Campaign Overview.

The REST API Browser includes information about the field names and field types of the campaign record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's campaign reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a campaign REST record is **campaign**.

Prerequisites

You must enable the Marketing Automation feature before you can use this record through REST web services. For more information, see the help topic Marketing Automation Overview.

Limitations

Consider the following limitations when using the campaign record in REST.

- The following field is not accessible through REST web services: Template Category.
- The following supporting records are not exposed to REST:
  - Category
  - Audience
  - Vertical
  - Family
  - Search Engine
  - Channel
  - Offer

  You can set related fields on the campaign record, but the supporting records cannot be accessed directly or modified.

Code Samples

The following samples show common actions for a campaign.

Get Campaign

```
GET <domain>/record/v1/campaign/<ID>
```

Update the Title Field on an Existing Campaign

```
PATCH <domain>/record/v1/campaign/127
{
   "title": "New Campaign Title"
}
```

Add an Unscheduled Email Event to an Existing Campaign

```
PATCH <domain>/record/v1/campaign/<ID>
{ "campaignEmail": { "items": [{ "campaignGroup": "<Group ID>", "channel": "<Channel ID>", "cost": 11.0, "datescheduled":
 "2023-10-30", "timescheduled": "10:20", "description": "From REST", "promoCode": "<Protocode ID>", "status": "<Status ID>",
 "subscription": "<Subscription ID>", "template": "<Template ID>"
}] }
}
```

Campaign Response

The campaign response record exposes a campaign response to REST web services.

This record:

- is not a subrecord
- contains one subrecord: responses

For more information about working with campaign response records in the UI, see the help topic Tracking Campaign Responses.

The REST API Browser includes information about the field names and field types of the campaign respone record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's campaign response reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a campaign response REST record is **campaignResponse**.

Prerequisites

You must enable the Marketing Automation feature before you can use this record through REST web services. For more information, see the help topic Marketing Automation Overview.

Code Sample

The following sample shows a common action for a campaign response.

Get Campaign Response

```
GET <domain>/record/v1/campaignResponse/<ID>
```

Create a Campaign Response

```
POST: /record/v1/campaignresponse
{
            "entity": {"id": 291},
```

```
            "leadSource": {"id": 25},
            "campaignEvent": {"id": 17},
            "response": {"id": "RESPONDED"}
}
```

Cash Refund

The cash refund record exposes a cash refund to REST web services. This record:

- is not a subrecord
- has no subrecords

A cash refund transaction records the return of money to a customer who immediately paid for goods or services using cash, a check or a credit card. For details about this type of transaction, see the help topic Refunding a Cash Sale.

There are no prerequisites for using this record through REST web services.

Using this record requires you to specify an item line number that is 1-indexed.

The REST API Browser includes information about the field names and field types of the cash refund record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's cash refund reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

This record has fields related to taxation features. To view the complete list of fields, go to Cash Refund in SuiteScript Records Browser.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record ID

The record ID for a cash refund REST record is **CashRefund**.

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show common use cases for cash refund.

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashrefund
{ "account": { "id": "155", "refName": "660 US Bank USD" }, "currency": { "id": "1", "refName": "USA" }, "entity":
 { "id": "110", "refName": "Anonymous Customer" }, "item": { "items": [ { "item": { "id": "98" }, "quantity": 1 } ] }, "location":
 { "id": "6", "refName": "US ONLY LOCATION" }
}
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashrefund/318
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashrefund/318
{ "account": { "id": "155", "refName": "660 US Bank USD" }, "currency": { "id": "1", "refName": "USA" }, "entity":
 { "id": "110", "refName": "Anonymous Customer" }, "item": { "items": [ { "line": 1, "refName": "Custom", "rate": 49.95, "item":
 { "id": "98" }, "quantity": 2 } ] }, "location": { "id": "6", "refName": "US ONLY LOCATION" }
}
DELETE: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashrefund/318
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Cash Sale

The cash sale record exposes a cash sale to REST web services.

To access this record in NetSuite, go to Transactions > Sales > Enter Cash Sales.

For information about the cash sale record user interface, see the help topic Cash Sales.

There are no prerequisites for using this record through REST web services.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the cash sale record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's cash sale reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a cash sale REST record is **cashsale**.

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show common use cases for cash sales.

In the following examples,

- <accountID> represents your account ID.
- <recordKey> represents the database key for the invoice.

Creating a New Cash Sale With an Item

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale { "asofdate": "2021-03-15", "end
date": "2021-06-15", "entity": { "id": "217" }, "item": { "items": [ { "amount": 1000, "item": { "id": "143" } } ] }, "start
date": "2021-03-15"
}
```

Creating a New Cash Sale With a Billable Item

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale { "asofdate": "2021-03-15", "entity":
 { "id": "218" }, "itemcost": { "items": [ { "apply": true, "doc": { "id": "305" }, "line": 1 } ] }
}
```

Creating a New Cash Sale With a Billable Expense

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale { "entity": { "id": "217" }, "expcost":
 { "items": [ { "apply": true, "doc": { "id": "303" }, "line": 1 } ] }
}
```

Updating an Existing Cash Sale

Note: The start date, end date, and quantity are in line item 1.

```
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale/<recordKey> { "item": { "items":
 [ { "line": 1, "quantity": 5 } ] }, "startDate": "2021-03-05", "endDate": "2021-04-05"
}
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Change Order

A change order record exposes a SuiteBilling change order to REST web services. This record has the following subrecords:

- subLine
- newSubLine
- renewalSteps

The REST API Browser includes information about the field names and field types of the change order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's change order reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the change order REST record is `subscriptionChangeOrder`.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Code Sample

```
POST 'https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionchangeorder' \ "action": "MODI
FY_PRICING", "subscription": 208, "customer": 5, "billingAccount": 2, "subline": { "items": [ { "apply": true, "sequence": 1, "pri
ceplannew": 541 } ] }
}'
```

Charge

A charge record exposes a SuiteBilling charge to REST web services. This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the charge record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's charge reference page.

Record ID

The record ID for the charge REST record is **charge**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Additional Details

You must specify the stage field as an enumeration string. Possible values can be obtained through the Metadata Catalog. Charge type must be specified as an internal ID. You can obtain possible values through SuiteScript while creating a charge through the User Interface.

Code Sample

This sample shows a common billing usage case.

```
POST https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/charge
{ "billTo": 5, "stage": "READY_FOR_BILLING", "chargeDate": "2020-07-14", "chargeType": -19, "billingItem": 5, "rate": 100.0, "quan
tity": 1
}
```

Check

A check record exposes a check transaction to REST web services.

This record is not a subrecord but has two subrecords: check item and check expense.

All elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the check record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's check reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the check REST record is **check**.

The record IDs for the check REST subrecords are:

- **item**
- **expense**

Limitations

Voiding of checks is not supported in REST web services. You can only void a check in the UI, which creates a voided journal entry and the original check remains and has a **Voided** status.

Code Samples

These samples show common use cases for check REST methods.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://test.netsuite.com/services/rest/record/v1/metadata-catalog/check
```

Fetching a Check Record

```
GET https://test.netsuite.com/services/rest/record/v1/check/<id>
```

Creating a Check Record

```
POST https://test.netsuite.com/services/rest/record/v1/check
{ "account": "1", "balance": -171.5, "cleared": false, "createdDate": "2023-06-13T20:32:00Z", "currency": "1", "customFor
m": { "id": "49", "refName": "Standard Check" }, "entity": "40", "exchangeRate": 1.0, "expense": { "items": [ { "account":
 { "id": "64" }, "amount": 333.00, "taxCode": { "id": "99" } } ] }, "oldaccount": "151", "postingPeriod": { "id": "553", "ref
Name": "Jun 2023" }, "prevDate": "2023-06-13", "toBePrinted": false, "tranDate": "2023-06-13", "tranId": "1"
}
```

Updating a Check Record

```
PATCH https://test.netsuite.com/services/rest/record/v1/check/<id>
{ "account": "1", "memo": "new memo"
}
```

Deleting a Check Record

```
DELETE https://test.netsuite.com/services/rest/record/v1/check/<id>
```

Class

Classes (or classifications) can be used to identify and categorize records in your NetSuite account. You can also define classifications for your organization's needs by creating custom segments. Classifications enable you to better organize data and preserve accuracy.

For information about working with classes in the UI, see the help topic Classifications in NetSuite.

The REST API Browser includes information about the field names and field types of the class record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's classification reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the classification REST record is **classification**.

The classification record has the following subrecord:

- classTranslation

Limitations

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "isInactive": false, "name": "REST test 1693820062", "subsidiary": { "items": [ { "id": 2 }, { "id": 4 } ] }, "parent": { "id": 1
 }, "includeChildren": true
}
```

Commerce Category

A commerce category record exposes a commerce category to REST web services.

This record:

- is not a subrecord
- has no subrecords
- has only sublists

The REST API Browser includes information about the field names and field types of the commerce category record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's commerce category reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a commerce category REST record is **commercecategory**.

Prerequisites

You must Enable Commerce Categories Feature for any interaction with the record. The account must also contain a Commerce Catalog record to reference.

Inaccessible Elements

The following sublists are not accessible through REST web services:

- URL Fragment Aliases sublist is not exposed
- "isCurrentlyEffective is internal only

Elements with Different Functionality

Updating the tags collection using REST does not delete removed tags, it only adds new tags and updates existing tags.

Additional Details

The "displayinSite" field values must be "T" or "F", not standard boolean.

The "sitemapPriority" field values must be a string with possible values between "0.0" and "1.0" incrementing by tenths. For example, ("0.1", "0.2", "0.3", etc.). This behavior is the same across all channels, not specific to REST web services.

Code Sample

Updating a Commerce Category Record

This example shows how to update a commerce category record.

```
HTTP Method: PATCH
URL: {NETSUITE_DOMAIN}/services/rest/record/v1/commercecategory/{id}
Request Parameters: {id} - Commerce Category ID to update
HTTP Request Headers: Content-Type: application/json
{ "name": "REST Category", "catalog": 1, "urlFragment": "rest-category", "isInactive": false, "addToHead": "string", "metaDe
scription": "string", "metaKeywords": "string", "pageTitle": "string", "pageHeading": "string", "externalId": "string", "de
scription": "string", "startDate": "2023-07-13T16:37:57.000Z", "endDate": "2024-07-13T16:37:57.000Z", "sequenceNum
ber": 0, "displayInSite": "T", "sitemapPriority": "0.0", "thumbnail": 5034, "pageBanner": 5034, "subcategories": { "items":
 [ { "subcategory": "2", "pageHeadingOverride": "string", "addToHeadOverride": "string", "displayInSiteOverride": "F", "se
quenceNumber": 0, "nameOverride": "Child REST Category", "urlFragmentOverride": "child-rest-category", "pageTitleOver
ride": "string", "metaDescriptionOverride": "string", "metaKeywordsOverride": "string", "sitemapPriorityOverride": "1.0", "de
scriptionOverride": "string", "pageBannerOverride": 3323, "thumbnailOverride": 3323 }, { "subcategory": "3", "sequen
ceNumber": 1, "nameOverride": "Another Child REST Category", "urlFragmentOverride": "another-child-rest-category"
 } ] }, "items": { "items": [ { "primaryCategory": true, "sequenceNumber": 0, "item": 206 }, { "primaryCategory": true, "se
quenceNumber": 1, "item": 227 } ] }, "translations": { "items": [ { "metaDescription": "string", "name": "string", "de
scription": "string", "pageTitle": "string", "pageHeading": "string", "locale": "fr_CA", "addToHead": "string", "metaKey
words": "string", "thumbnail": 3323, "pageBanner": 3323 }, { "metaDescription": "string", "name": "string", "pageTi
tle": "string", "pageHeading": "string", "locale": "es_ES", "metaKeywords": "string" } ] }, "tags": { "items":
 [ { "name": "tag1", "id": 1 }, { "name": "tag2", "id": 2 } ] }
}
```

Competitor

You can create competitor records to track how other businesses in your industry impact your sales. In the UI, this is a user defined list at Lists > Relationships > Competitors > New.

Use the competitor record to create new competitors and expose them to REST web services.

To learn more, see the help topic Competitor.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the competitor record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's competitor reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a competitor REST record is **competitor**.

Prerequisites

You must enable Permissions > Lists > Competitors before you can use this record through REST web services.

Elements Not Accessible Through REST Web Services

The following elements are not accessible through REST web services:

- opportunities

Usage Notes

Review the following notes on working with the competitor REST record:

- `name` is required, all other fields are optional

Code Sample

The following samples show common use cases for competitor.

Creating a Competitor

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/competitor
Body: { "name": "A Competitor", "description": "Description of competitor", "strengths": "Competitor's strengths", "weaknesses": "Competitor's weaknesses", "strategy": "Strategy for winning against competitor", "productService": "Competitor's products and services", "isInactive": false
}
```

Retrieving a Competitor

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/competitor/2
```

Updating a Competitor

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/competitor/2
Body:
{ "name": "Updated Competitor", "description": "Updated description of competitor", "strengths": "Updated Competitor's strengths", "weaknesses": "Updated competitor's weaknesses", "strategy": "Updated strategy for winning against competitor", "productService": "Updated competitor's products and services", "isInactive": true
}
```

Deleting a Competitor

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/ competitor/2
```

Consolidated Exchange Rate

A consolidated exchange rate record exposes a consolidated exchange rate to REST web services.

Consolidated exchange rate records ensure that currency amounts translate properly from child to parent subsidiaries for consolidated reports. NetSuite creates initial consolidated exchange rates when associated subsidiaries and accounting periods are created.

To access this record, go to Lists > Accounting > Consolidated Exchange Rates.

For more information about consolidated exchange rates in NetSuite, see the help topic Consolidated Exchange Rates.

The REST API Browser includes information about the field names and field types of the consolidated exchange rate record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's consolidatedExchangeRate reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a consolidated exchange rate REST record is **ConsolidatedExchangeRate**.

Prerequisites

You must enable the Multiple Currencies feature before you can use this record through REST web services.

Elements with Different Functionality

The following fields have different functionality through REST web services:

- accountingBook (Read only)
- fromCurrency (Read only)
- fromSubsidiary (Read only)

- isDerived (Read only)
- isEliminationSubsidiary (Read only)
- isPeriodClosed (Read only)
- periodStartDate (Read only)
- postingPeriod (Read only)
- toCurrency (Read only)
- toSubsidiary (Read only)

Usage Notes

The Full Multi-Book Accounting feature must be enabled if you want to view or edit the following field:

- accountingBook

Code Sample

This sample shows how to update the historical exchange rate using a PATCH request:

```
PATCH {{REST_SERVICES}}/record/v1/consolidatedExchangeRate/10
{
"historicalRate": 0.8655
}
```

Contact

A contact record exposes a contact to REST web services.

This record:

- is not a subrecord
- has no subrecords

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the contact record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's contact reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Tax-related fields are not supported in REST.

Record ID

The record ID for a contact REST record is **contact**.

Code Sample

In the following example, <*accountID*> represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/contact/973`

```
{ "firstName": "Peter", "contactSource": "99994", "comments": "Works directly with all key decision-makers."
}
```

Contact Category

A contact category record exposes a contact category to REST web services.

This record:

- is not a subrecord
- has no subrecords

All elements on this record are accessible through REST web services.

Record actions supported for this record type include delete, submitnew, and submitter. For more information, see the help topic Supported Record Actions.

The REST API Browser includes information about the field names and field types of the contact category record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's contactCategory reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a contact category REST record is **contactcategory**.

Code Sample

In the following example, <*accountID*> represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/contactcategory/4`

```
{ "name": "ContactCategoryFour", "externalid": "3929345", "private": true
}
```

Contact Role

A contact role record exposes a contact role to REST web services.

This record:

- is not a subrecord

- has no subrecords

All elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the contact role record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's contactRole reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a contact role REST record is **contactrole**.

Code Sample

In the following example, *<accountID>* represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/contactrole/1`

```
{ "name": "ContactRoleOne", "description": "Key Contact", "isinactive": true
}
```

Cost Category

A cost category record exposes a cost category to REST web services. This record is not a subrecord. This record does not have any subrecords.

For help working with this record in the UI, see the help topic Creating Cost Categories.

The REST API Browser includes information about the field names and field types of the cost category record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's costCategory reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a cost category REST record is **CostCategory**.

Prerequisites

You must enable either the Standard Costing or Landed Cost feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for cost categories. The example ID is 4.

Creating a Cost Category Using a POST Request

This sample shows how to create a cost category when you are using the Standard Costing feature only.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory
{ "itemCostType": { "id": "MATERIAL" }, "name": "Test Name"
}
```

This sample shows how to create a cost category when you are using the Landed Cost feature only.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory
{ "itemCostType": { "id": "LANDED" }, "name": "Air Freight Charges", "account": { "id": 69 }
}
```

Retrieving a Cost Category Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory/4
```

Updating a Cost Category Using a PATCH Request

This sample shows how to update a cost category when you are using the Standard Costing feature only.

You can update the name only. You cannot change the cost type after you create the cost category.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory/4
{ "name": "New Name",
}
```

This sample shows how to update a cost category when you are using the Landed Cost feature only.

You can update the name and expense account ID only. You cannot change the cost type after you create the cost category.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory/4
{ "name": "FedEx Charges", "account": { "id": 65 }
}
```

Deleting a Cost Category Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/costCategory/4
```

Coupon Code

Promotions enable you to track the source of revenue and to offer discounts in the form of coupons. Each promotion has a promotion code that can be applied to transactions and campaigns.

This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the coupon code record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's couponCode reference page.

For information about coupon codes, see the help topic Coupon Code.

For information about working with coupon codes in the UI, see the help topic Coupon Codes.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the coupon code REST record is **couponcode**.

Prerequisites

You must enable the SuitePromotions feature before you can use this record through REST web services.

Elements with Different Functionality

The following fields have different functionality through REST web services:

- used
- useCount

Additional Details

For multi-use coupon code type promotions:

- The promotion must have at least one coupon code associated to it so that it is usable.

  If you remove all the coupon codes, you will see an empty coupon code field in the UI, and from the REST point of view, the code field will not be returned with the promotioncode record.

  However, you can add the coupon codes through the user interface or through REST by updating the code field in the promotioncode record.
- You can update the coupon code field using a PATCH request to the promotioncode record.

For single-use coupon code type promotions:

- The promotion must have at least one coupon code associated to it so that it is usable.

  If you remove all the coupon codes, you will have an empty list of coupon codes.

  However, you can add the coupon codes through the user interface or through REST.
- You cannot update the coupon code field using a PATCH request to the promotioncode record.

Code Sample

In the following example, 123 represents the promotion ID.

The following code shows how to create a coupon code:

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/couponCode
{ "promotion": {"id": 123}, "code": "ABC-ABC-100"
```

```
}
```

In the following example, 206 represents the promotion ID and 15 represents the recipient ID.

The following code shows how to update a coupon code:

```
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/couponCode/206
{ "promotion": {"id": 123}, "code": "ABC-CDE-200", "dateSent": "2024-06-18", "recipient": {"id": "15"},
}
```

Note: When updating the coupon code records, consider the difference between single-use and multi-use coupon code type promotions.

The following code shows how to obtain a coupon code using a GET Request:

```
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/couponCode/206
```

The following code shows how to delete a coupon code:

```
DELETE: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/couponCode/206
```

Note: If a coupon code is already being used in a transaction, you won't be able to remove the promotions record.

Credit Card Charge

A credit card charge record exposes outgoing credit card transactions to REST web services.

This record is not a subrecord but has two subrecords: credit card charge expense and credit card charge item.

The REST API Browser includes information about the field names and field types of the credit card charge record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's credit card charge reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a credit card charge REST record is **creditCardCharge**.

The record IDs for the credit card charge REST subrecords are:

- **item**
- **expense**

Prerequisites

Before you can use this record through REST web services, you must enable and set up the SuiteTax feature as legacy tax fields are not available over REST. See the help topic Enabling the SuiteTax Feature.

Limitations

Updating legacy tax fields is not supported.

Code Samples

These samples show common use cases for credit card charge REST methods.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://test.netsuite.com/services/rest/record/v1/metadata-catalog/creditcardcharge
```

Fetching a Credit Card Charge Record

```
GET https://test.netsuite.com/services/rest/record/v1/creditcardcharge/<id>
```

Creating a Credit Card Charge Record

```
POST https://test.netsuite.com/services/rest/record/v1/creditcardcharge
{
"account": "123",
"entity": "26", "expense": { "items": [ { "account": "58", "amount": 50.00 } ] }
}
```

Updating a Credit Card Charge Record

```
PATCH https://test.netsuite.com/services/rest/record/v1/creditcardcharge/<id>
{ "account": "204", "entity": "10"
}
```

Deleting a Credit Card Charge Record

```
DELETE https://test.netsuite.com/services/rest/record/v1/creditcardcharge/<id>
```

Credit Card Refund

A credit card refund record exposes incoming credit card transactions to REST web services.

This record is not a subrecord but has two subrecords: credit card refund expense and credit card refund item.

With the exception of tax-related fields, all elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the credit card refund record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's credit card refund reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a credit card refund REST record is **creditCardRefund**.

The record IDs for the credit card refund REST subrecords are:

- **item**
- **expense**

Prerequisites

Before you can use this record through REST web services, you must enable and set up the SuiteTax feature as legacy tax fields are not available over REST. See the help topic Enabling the SuiteTax Feature.

Limitations

Updating legacy tax fields is not supported.

Code Samples

These samples show common use cases for credit card refund REST methods.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://test.netsuite.com/services/rest/record/v1/metadata-catalog/creditcardrefund
```

Fetching a Credit Card Refund Record

```
GET https://test.netsuite.com/services/rest/record/v1/creditcardrefund/<id>
```

Creating a Credit Card Refund Record

```
POST https://test.netsuite.com/services/rest/record/v1/creditcardrefund
{ "account": "123", "entity": "26", "expense": { "items": [ { "account": "58", "amount": 50.00 } ] }
}
```

Updating a Credit Card Refund Record

```
PATCH https://test.netsuite.com/services/rest/record/v1/creditcardrefund/<id>
{ "account": "204", "entity": "10"
}
```

Deleting a Credit Card Refund Record

```
DELETE https://test.netsuite.com/services/rest/record/v1/creditcardrefund/<id>
```

Credit Memo

A credit memo record exposes a credit memo to REST web services. A credit memo is used to refund or credit a customer account. For more information about credit memos and how they function in the user interface, see the help topic Customer Credit Memos.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the credit memo record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's credit memo reference page.

For information about using the REST API browser, see the help topic The REST API Browser.

This record:

- is not a subrecord
- does not have any subrecords

Record ID

The record ID for the credit memo record is **creditmemo**.

Prerequisites

Prerequisites include:

- REST Web Services feature
- Two-factor authentication (for REST)
- Accounts receivable (A/R) for the credit memo record

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Sample

The following example is for creating a credit memo record.

```
Create
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo { "entity": { "id": "4" }, "item": { "items":
 [ { "amount": 1000.0, "item": { "id": "5" } } ] }, "subsidiary": { "id": "1" } "location": "5"
} Get
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/101 Update
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/101
{"memo": 'test memo'}
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Currency

A currency record exposes a currency to REST web services.

Currency records are created for the different currencies you want to use in your transactions and for your subsidiaries if you use NetSuite OneWorld.

To access this record in NetSuite, go to Lists > Accounting > Currencies > New.

For more information about currencies in NetSuite, see the help topic Currency Management.

The REST API Browser includes information about the field names and field types of the currency record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's currency reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a currency REST record is **Currency**.

Prerequisites

You must have the Multiple Currencies feature enabled before you can use this record through REST web services. For more information about this feature, see the help topic Enabling the Multiple Currencies Feature.

Elements with Different Functionality

The following fields have different functionality through REST web services:

- formatSample (Read only)
- isAnchorCurrency (Read only)
- isBaseCurrency (Read only)
- lastModifiedDate (Read only)

Usage Notes

The Currency Exchange Rate Integration feature must be enabled to view and edit the following fields:

- includeInFxRateUpdates
- fxRateUpdateTimeZone

The Use Triangulation Calculation By NetSuite accounting preference must be enabled to view and edit the following field:

- isAnchorCurrency

Code Sample

This sample shows how to update the Override Currency Format box from false to true:

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/currency/2
{
"overrideCurrencyFormat": true
}
```

Currency Rate

A currency rate record exposes a currency exchange rate to REST web services.

Currency exchange rates are used to convert foreign currencies to base currencies, providing default rates for transactions in currencies other than the base currency. Exchange rates are expressed in terms of base currency units per foreign currency (source) unit.

To access this record in NetSuite, go to Lists > Accounting > Currency Exchange Rates > New.

For more information about currency exchange rates in NetSuite, see the help topic Currency Exchange Rates.

The REST API Browser includes information about the field names and field types of the currency exchange rate record and about the HTTP methods, request parameters, and operations available to this record. For details, see the Rest API Browser's currency rate reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a currency rate REST record is **currencyrate**.

Prerequisites

You must enable the Multiple Currencies feature before you can use this record through REST web services. For more information about this feature, see the help topic Enabling the Multiple Currencies Feature.

Accessible Elements

The following fields are not accessible through REST web services:

- Previous Effective Date
- Previous Exchange Rate

Code Sample

The following code sample shows how to create a currency exchange rate record:

```
POST: {{COMPANY_URL}}/services/rest/record/v1/currencyrate
{
    "baseCurrency": "1",
    "transactionCurrency": "3",
    "exchangeRate": 1.9,
    "effectiveDate": "2023-11-26",
    "externalId": "1"
}
```

Customer

A customer record exposes a customer to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - addressbook
  - campaigns
  - contactroles
  - currencylist
  - grouppricing
  - itempricing
  - partners

- salesteam
- subscriptionmessagehistory
- subscriptions
- taxregistration

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the customer record, and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's customer reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the customer REST record is **customer**.

Inaccessible Elements

The following sublists are not accessible through REST web services:

- bulkmerge
- creditcards
- download
- eftacct
- itemorders
- paymentinstruments
- referrerlist
- submachine
- upsell

Elements With Limited Functionality

The create operation is not supported for the campaigns and contactroles elements.

Code Sample

The following code sample shows how to create a customer record:

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customer
{ "companyName": "Company 1615554322", "email": "customer@example.com"
}
```

Customer Category

NetSuite exposes the customer category record to REST web services. Customer category defines a list of values that are used by the customer record to set the type of customer.

Record ID

The record ID for the customer category REST record is **customerCategory**.

Prerequisites

There are no prerequisites required for using this record through REST web services.

The customer category record is not a subrecord and is used on the customer record and other CRM records.

There are no elements on this record that are not accessible through REST web services.

The REST API Browser includes information about the field names and field types of the estimate record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customer category reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Code Samples

The code samples in this section show common use cases for customer category items.

Creating a Customer Category

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customercategory
{ "name": "Customer Category 1"
}
```

Updating a Customer Category

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customercategory/{id}
{ "name": "Customer Category 2"
}
```

Customer Deposit

NetSuite exposes the customer deposit record to REST web services. A customer deposit transaction records the funds received when a customer makes an advance payment for an order. This payment is recorded in the general ledger as a liability until the goods or services are delivered, and does not affect the customer's accounts receivable balance. After the order is filled, the deposit is applied against the invoice.

For more information about customer deposits, see the help topic Customer Deposits.

There are no elements on this record that are not accessible through REST web services.

The REST API Browser includes information about the field names and field types of the customer deposit record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customerDeposit reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the customer deposit REST record is **customerDeposit**.

Sublist

The customer deposit record has a sublist of the **accountingBookDetail** subrecord.

Prerequisite

Before you work with customer deposits, enable the A/R feature.

Code Samples

The code samples in this section show common use cases for customer deposits.

Creating a Customer Deposit

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customerdeposit
{ "customer":{"id": 6}, "currency":{"id": 1}, "exchangerate": 1.00, "payment": 10.00, "paymentOption": {"id": 4}
}
```

Updating a Customer Deposit

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customerdeposit/{id}
{ "payment": 12.0, }
```

Customer Message

A customer message record exposes a customer message to REST web services.

For more information about customer messages and how they function in the user interface, see the help topic Setting Up Accounting Lists.

To access this record in NetSuite, go to Setup > Accounting > Accounting Lists.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the customer message record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customer message reference page.

For information about using the REST API Browser, see the help topic The REST API Browser

Record ID

The record ID for the customer message REST record is **customermessage**.

Code Sample

The following samples show common use cases for a customer message record. The example ID is 6.

Retrieving Customer Message Metadata

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/metadata-catalog/customermessage
```

Creating a Customer Message Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/customermessage
{ "description": "example message description", "isInactive": false, "name": "ExampleCustomerMessage", "preferred": false
}
```

Deleting a Customer Message Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/customermessage/6
```

Customer Payment

NetSuite exposes the customer payment record to REST web services. A customer payment transaction records a customer payment and applies it to the appropriate invoice or cash sale, decreasing the amount due, and tracking revenue.

The customer payment record has no subrecords.

There are no prerequisites for using this record through REST web services.

For more information about the customer payment record, see the help topic Customer Payments.

The REST API Browser includes information about the field names and field types of the customer payment record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customerPayment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the customer payment REST record is **customerpayment.**

Code Samples

The following code samples show how to work with the customer payment record.

Creating a Customer Payment Record

```
{
    "account": {
        "id": "1"
    },
    "apply": {
        "items": [
            {
                "amount": 200.0,
                "apply": true,
                "doc": {
                    "id": "104"
                },
                "line": 0
            }
        ]
    },
    "currency": {
        "id": "1"
    },
    "customer": {
        "id": "4"
    },
    "memo": "10",
    "payment": 250,
    "toBeEmailed": false
}
```

Getting an Existing Customer Payment by ID

```
HTTP method: GET
URL: {{COMPANY_URL}}/services/rest/record/v1/customerpayment/{{customerPaymentId}}
```

Changing the Applied Invoice Amount by Updating an Existing Customer Payment by ID

```
HTTP method: PATCH
URL: {{COMPANY_URL}}/services/rest/record/v1/customerpayment/{{customerPaymentId}}
Request body: { "apply": { "items": [ { "doc": { "id": "4" }, "amount": 200 } ] }, "customer": { "id": "4" }
}
```

Adding an Invoice to Apply by Updating an Existing Customer Payment by ID

```
HTTP method: PATCH
URL: {{COMPANY_URL}}/services/rest/record/v1/customerpayment/{{customerPaymentId}}
Request body: { "apply": { "items": [ { "doc": { "id": "104" }, "amount": 10 } ] }, "customer": { "id": "4" }
```

```
}
```

Unapply an Invoice by Updating an Existing Customer Payment by ID

```
{ "apply": { "items": [ { "doc": { "id": "104" }, "apply": false } ] }, "customer": { "id": "4" }
}
```

```
{ "apply": { "items": [ { "doc": { "id": "104" }, "amount": 0 } ] }, "customer": { "id": "4" }
}
```

Customer Refund

NetSuite exposes the customer refund record to REST web services. A customer refund transaction records the return of funds to a customer who paid for goods or services using cash, a check, or a payment card. The refund is generally made in cash or by check.

For more information about customer refunds, see the help topic Customer Refunds.

The REST API Browser includes information about the field names and field types of the customer refund record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customerRefund reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the customer refund REST record is **customerRefund**.

Subrecords and Sublists

The customer refund record has the following sublists and subrecord:

- The apply sublist of the CustomerRefundApplyLine subrecord.
- The deposit sublist of the CustomerRefundDepositLine subrecord.
- The TransactionAccountingBookDetail subrecord.

Prerequisite

Before you work with customer refunds, enable the A/R feature.

Limitations

Certain elements on the customer refund record are not accessible through REST web services: the deposit sublist in the GET request is empty. The deposit sublist is hidden in edit mode in UI, and the applied deposit is represented by Deposit Application in apply sublist. This is how sublists are displayed in GET response in REST.

Certain elements on the customer refund record have a different level of functionality than the rest of the record:

- The deposit sublist is always empty in response for GET request and cannot be updated.
- Lines on the deposit sublist are not identified by sublist line number, but by the doc field as the key. See the Creating a Customer Refund code sample.
- Lines on the apply sublist are not identified by the line number, but by the doc field as the key with the line as a secondary key. The line represents the transaction line ID, not the line number on the sublist. See the Creating a Customer Refund code sample.

Code Samples

The code samples in this section show common use cases for customer refunds.

Creating a Customer Refund

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customerrefund
{ "customer": {"id": "6"}, "deposit": { "items": [ { "doc": {"id": "60"}, "apply": true, "amount": 25.00 } ] }, "apply": { "items":
 [ { "doc": {"id": "59"}, "line": 1, "apply": true, "amount": 4.20 }, { "doc": {"id": "56"}, "line": 1, "apply": true, "amoun
t": 5.00 } ] }
}
```

Updating a Customer Refund

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customerrefund/{id}
{ "memo": "Updated refund"
}
```

Customer Status

A customer status record exposes a customer status to REST web services. It describes a lead, prospect, or a customer's stage in the sales cycle.

For more information about customer statuses and how they function in the user interface, see the help topic Customer Statuses.

To access this record in NetSuite, go to Setup > Sales > Setup Tasks > Customer Statuses > New.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the customer status record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customer status reference page.

For information about using the REST API Browser, see the help topic The REST API Browser

Record ID

The record ID for the customer status REST record is **customerstatus**.

Code Sample

The following samples show common use cases for a customer status record. The example ID is 6.

Retrieving Customer Status Metadata

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/metadata-catalog/customerstatus
```

Updating a Customer Status Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/customerstatus/6
{ "probability": 99.0
}
```

Deleting a Customer Status Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/customerstatus/6
```

Customer Subsidiary Relationship

NetSuite exposes the customer subsidiary relationship record to REST web services. This record enables you to manage a specific customerSubsidiaryRelationship record. You can also get a list of customer subsidiary relationship records. This record is not accessible through the user interface.

The REST API Browser includes information about the field names and field types of the customer subsidiary relationship record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's customerSubsidiaryRelationship reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the customer subsidiary relationship REST record is **customerSubsidiaryRelationship**.

Prerequisites

You must either use NetSuite OneWorld or have the Subsidiaries hidden feature enabled before you can use this record through REST web services.

Updatable Fields on the customerSubsidiaryRelationship Record

You can update the following fields on the customerSubsidiaryRelationship record:

- custom fields (available on the customer record, Subsidiary subtab)

- (externalid) - external id

Supported Operations

The following operations are supported for REST web services:

- GET (Read, Search for a list of customer subsidiary relationship records)
- POST (insert a record)
- DELETE (a specific record)
- GET (Read, Search a specific record)
- PATCH (update a specific record)
- PUT (insert or update a specific record)

Tax-related fields are not supported in REST.

Code Samples

The following code sample shows how to get a specific customerSubsidiaryRelationship record:

```
GET https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customersubsidiaryrelationship?q=entity
EQUAL 13 AND subsidiary EQUAL 5
```

The following code sample shows how to get the ID of a specific customerSubsidiaryRelationship record:

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5
```

```
Header:
'prefer: transient'
'Content-Type: application/json'
Body:
'{ "q": "SELECT id FROM customerSubsidiaryRelationship WHERE entity = 13 AND subsidiary = 5"
}'
```

The following code sample shows how to set the externalId for a specific customerSubsidiaryRelationship record:

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customersubsidiaryrelationship/1
```

```
Header:
'Content-Type: application/json'
Body:
'{ "externalId": "VSR1" }'
```

Department

Departments can be used to identify and categorize records in your NetSuite account. You can also define departments for your organization's needs by creating custom segments. Departments enable you to better organize data and preserve accuracy.

For information about working with departments in the UI, see the help topic Classifications in NetSuite.

The REST API Browser includes information about the field names and field types of the department record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's department reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the department REST record is **department**.

The department record has the following subrecord:

- classTranslation

Limitations

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "isInactive": false, "name": "REST test 1693820309", "subsidiary": { "items": [ { "id": 2 }, { "id": 4 } ] }, "parent": { "id": 1
 }, "includeChildren": true
}
```

Deposit

A deposit record exposes a deposit to REST web services.

In NetSuite, record deposits to your bank accounts to capture customer payments and other monies received in the course of doing business. By recording funds deposited, you can accurately track income. For information about working with this record in the UI, see the help topic Deposits.

This record:

- is not a subrecord
- has the following three sublists:
  - Deposit Cashback (ID: `cashback`)
  - Deposit Other (ID: `other`)
  - Deposit Payment (ID: `payment`)

The REST API Browser includes information about the field names and field types of the deposit record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's deposit reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Usage Notes

The **To Be Printed** field is not exposed to REST web services.

In both the UI and in REST web services, the **Exchange Rate** field is a create-time-only field that is used to specify the initial exchange rate with respect to the base currency. After the initial exchange rate value is set, this value does not get updated to reflect real-time exchange rates.

Record ID

The record ID for the deposit REST record is **deposit**.

Code Samples

The following code samples show how to work with the deposit record.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/metadata-catalog?select=deposit
```

Creating a Deposit with Other Deposit and Cash Back Lines

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit
{ "account": { "id": "1" }, "cashback": { "items": [ { "account": { "id": "10" }, "amount": 18.46 } ] }, "other": { "items":
 [ { "account": { "id": "97" }, "amount": 178.24, "entity": { "id": "107" }, "paymentMethod": { "id": "1" } } ] }
}
```

Adding a Deposit Other Line

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit/{{depositId}} { "other": { "items": [ { "account":
 { "id": "10" }, "amount": 78.5, "entity": { "id": "69" }, "memo": "Deposit Other MEMO", "paymentMethod": { "id": "5" } } ] }
}
```

Replacing a Deposit Cashback Sublist

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit/{{depositId}}?replace=cashback { "cashback":
 { "items": [ { "account": { "id": "97" }, "amount": 12.0, "memo": "Cashback item of new sublist" } ] }
}
```

Creating a Deposit with Payment Lines

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit { "account": { "id": "1" }, "payment": { "items":
 [ { "deposit": true, "id": 379 }, { "deposit": true, "id": 380 } ] }
}
```

Adding a Deposit Payment Line

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit/{{depositId}} { "payment": { "items":
 [ { "deposit": true, "id": 381 } ] }
}
```

Getting a Deposit by ID

```
GET https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit/{{depositId}}
```

Removing a Deposit by ID

```
DELETE https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/deposit/{{depositId}}
```

Deposit Application

NetSuite exposes the deposit application record to REST web services. A deposit application transaction applies a customer deposit against an invoice after the order is complete. For details, see the help topic Applying a Customer Deposit.

The REST API Browser includes information about the field names and field types of the deposit application record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's depositApplication reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the deposit application REST record is **depositApplication**.

Subrecords

The deposit application record has a list of depositApplicationApplyLine and accountingBookDetail subrecords.

Prerequisite

Before you work with deposit applications, enable the A/R feature.

Note: Lines on the apply sublist are not identified by line numbers, but by the doc field as a key with the line as a secondary key. The line represents the transaction line ID, not the number of line on the sublist.

Also, note that you can create a deposit application only by transformation. See Creating a Deposit Application.

Code Samples

Creating a Deposit Application

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/customerDeposit/<depositId>/!transform/depositApplication
```

```
{ "apply": { "items": [ { "doc": {"id": 261}, "apply": true } ] }
}
```

Updating a Deposit Application

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/depositapplication/{id}
{ "memo": "UPDATED TEST"
}
```

Description Item

Description line items let you place sentence or paragraph long descriptions on items you are not selling. For example, you can enter special shipping instructions or a disclaimer.

To learn more, see the help topic Description Items.

This record is not a subrecord, nor does it have any subrecords.

The REST API Browser includes information about the field names and field types of the description item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's descriptionItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the description item REST record is **descriptionitem**.

Code Sample

Creating a Description Item

```
POST {{COMPANY_URL}}/services/rest/record/v1/descriptionitem { "itemId": "MY-CUSTOM-UNIQUE-ITEM-ID", "description": "Special
 shipping instructions..."
}
```

Getting a Description Item by ID

```
GET {{COMPANY_URL}}/services/rest/record/v1/descriptionitem/{{descriptionItemId}}


{{descriptionItemId}} - existing description items record's ID
```

Finding a Description Item by itemId

```
GET {{COMPANY_URL}}/services/rest/record/v1/descriptionitem?q=itemId IS MY-CUSTOM-UNIQUE-ITEM-ID

Query parameters: q={{expression}}
```

```

possible expression: field {{operator}} field's value

example: itemId IS MY-CUSTOM-UNIQUE-ITEM-ID
```

Updating an Existing Description Item

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/descriptionitem/{{descriptionItemId}} { "description": "Updated descrip
tion", "itemId": "updatedDescriptionItemId"
}
```

Removing a Description Item

```
DELETE {{COMPANY_URL}}/services/rest/record/v1/descriptionitem/{{descriptionItemId}}
```

Discount Item

A discount item record exposes a discount item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the discount item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's discountItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a discount item REST record is **discountItem**.

Taxation

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The discount item includes the Apply Before Tax field related to taxation features.

Code Samples

The following samples show common use cases for discount items. The example ID is 4.

Creating a Discount Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/discountItem
{ "isInactive": false, "itemId": "031036c6-1815-4b95-87d6-d5f9367a1cf5", "rate": 8.0, "displayName": "Test nonposting", "account":
 { "id": "105", "refName": "Discount 500$ After Tax Non Post", "externalId": "2" }, "nonPosting": false
}
```

Retrieving a Discount Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/discountItem/4
```

Updating a Discount Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/discountItem/4
{ "isInactive": false, "description": "Discount 10$ After Tax Non Post", "itemId": "03", "rate": "", "displayName": "Test", "accoun
t": { "id": "105", "refName": "Discount 5$ After Tax Non Post", "externalId": "2" }
}
```

Deleting a Discount Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/discountItem/4
```

Download Item

A download item record exposes a download item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the download item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's DownloadItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a download item REST record is **downloadItem**.

Prerequisites

You must first enable the feature.

Code Samples

The following samples show common use cases for download items. The example ID is 4.

Creating a Download Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/downloadItem
{ "itemId": "download item", "subsidiary": { "items": [ { "id": "5" } ] }
}
```

Retrieving a Download Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/downloadItem/4
```

Updating a Download Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/downloadItem/4
{ "parent": {"id": 315}
}
```

Email Template

An email template record exposes an email template to REST web services.

This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the email template record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's emailTemplate reference page.

For information about using email templates, including adding fields to an email template, see the help topic Email Template.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the email template REST record is **emailtemplate**.

The record ID for the email template REST subrecord is **CrmTemplateCategorySubsidiary**.

Prerequisites

You must enable the CRM feature before you can use this record through REST web services.

Exposed Elements

The following email template record elements are exposed to REST web services:

- {addcompanyaddress} – add company address
- {addunsubscribelink} – add unsubscribe link
- {content} – content
- {description} – description
- {id} – ID
- {isautoconverted} – is automatically converted
- {isinactive} – is inactive
- {isprivate} – is private
- {mediaitem} – media item
- {name} – name
- {recordtype} – record type
- {subject} – subject

- {subscription} – subscription

Code Sample

In the following example, <*accountID*> represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/emailtemplate/973`

```
{ "name": "EmailTemplateNine", "addunsubscribelink": true, "description": "Marketing Campaign Template"
}
```

Employee

An employee record exposes an employee to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - AccruedTime
  - AddressBook
  - Certificate
  - CompanyContribution
  - CorporateCards
  - Deduction
  - DirectDepositList
  - Earning
  - EmpPerms
  - EmployeeEmergencyContact
  - EmployeeCurrency
  - EmployeeDirectDeposit
  - EmployeeHCMPosition
  - JurisdictionHist
  - Rates
  - TaxOptions

The REST API Browser includes information about the field names and field types of the employee record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's employee reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the employee REST record is **employee**.

Prerequisites

You must enable the Advanced Employee Permissions feature, the Employee Record, the Perform Search, the SuiteAnalytics Workbook, and SuiteScript before you can use this record through REST web services.

Inaccessible Elements

The following sublists are not accessible through REST web services:

- hcmposition
- usernotes

The following fields are not accessible through REST web services:

- adpwarining
- billpay
- directdeposit
- hasshippingaddress
- jurisdictionschool
- origbinactive
- oriqgiveaccess
- origsubstatus
- roleforsearch
- shipaddr1
- shipaddr2
- shipaddr3
- shipaddressee
- shipattention
- shipcity
- shipcountry
- shipstate
- shipzip
- unsubscribe

Elements with Different Functionality

The following sublists have different functionality through REST web services:

- directdepositlist
- jurisdictionhist
- roles

The following fields have different functionality through REST web services:

- adipid
- altname

- autoname
- changedetails
- conflictresults
- customform
- defaytkaddress
- defaultbillingaddress
- defaultshippingaddress
- effectivedatemode
- empcenterqty
- emplcenterqtymax
- employeechangereason
- entitynumber
- fulluserqty
- fulluserqtymax
- image
- isempcenterqtyenforced
- isfulluserqtyenforced
- isretailuserqtyenforced
- jurisdictioncounty
- jurisdictionfederal
- jurisdictionhist
- jurisdictionstate
- password
- password2
- phoneticname
- requiredpwdchange
- retailuserqty
- retailuserqtymax
- sendemail
- socialsecuritynumber
- terminationbydeath
- wasempcenterhasaccess
- wasfulluserhasaccess
- wasinactive
- wasretailuserhasaccess

Actions

The employee record has the following actions through REST web services:

- Delete

- Save As
- Save & New
- Save & Next
- Save
- Reset
- Customize Form
- New Field

Elements Related to Taxation Features

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The edition field is related to the ADVTAXENGINE feature.

This following fields are related to the ANYPAYROLL feature:

- jurisdictioncounty
- jusdictionfederal
- jursidictionhist
- jursidictionlocal
- jurisdictionstate
- lastpaiddate
- payfrequency
- useperquest
- usetimedate
- workplace
- adpwarning
- accruedtime
- companycontribution
- deduction
- earning
- emptaxoptions

Additional Details

The employee record is complex because it is integrated with many features.

Some elements of the employee form behave differently when accessed through REST web services.

Code Sample

The following code shows how to create a new employee record:

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/employee
{ "firstName": "TEST FIRSTNAME", "lastName": "TEST LASTNAME", "subsidiary": {"id":"1"}
}
```

The following code shows how to update an employee's last name:

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/employee/{id}
{ "lastName": "NEW LASTNAME"
}
```

Estimate

An estimate record exposes an estimate to REST web services. This record is not a subrecord. This record has one subrecord: record ID item.

The REST API Browser includes information about the field names and field types of the estimate record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's estimate reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the estimate REST record is **estimate**.

Prerequisites

You must enable estimates before you can use this record through REST web services. To do so, go to Setup > Company > Enable Features. On the **Transactions** subtab under Basic Features, check the **Estimates** box, and then click **Save**.

Actions

The estimates record has the following actions through REST web services:

- New
- Make Copy
- Email
- Show Activity
- Go To Register
- GL Impact

Field Exposure

Fields exposed through SuiteScript should all be exposed through REST web services. Some fields may be hidden on the UI.

Taxation

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The following fields on the estimate REST record are related to taxation:

- tax code
- tax rate
- shipping tax code
- shipping tax rate

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show common use cases for estimates. The example ID is 6.

Creating an Estimate Using a POST Request

```
POST {{REST_SERVICES}}/record/v1/estimate
{ "altSalesTotal": 0.0, "canHaveStackable": false, "currency": { "id": "1", "refName": "USA" }, "discountItem": { "id": "-6", "ref
Name": "Partner Discount" }, "discountRate": -5.0, "discountTotal": -5.0, "dueDate": "2023-08-09", "entity": { "id": "110", "ref
Name": "Anonymous Customer" }, "entityStatus": { "id": "10", "refName": "Proposal" }, "exchangeRate": 1.0, "expectedClose
Date": "2023-07-10", "forecastType": { "id": "2", "refName": "Most Likely" }, "item": { "items": [{ "line": 1, "item":
 { "id": 98}, "rate": 39.95, "quantity": 1 }] }, "subsidiary": { "id": "1", "refName": "Parent Company" }, "subtotal": 39.95, "to
BeEmailed": false, "toBeFaxed": false, "toBePrinted": false, "total": 34.95, "tranDate": "2023-07-10", "tranId": "1", "visibleToCus
tomer": true
}
```

Retrieving an Estimate Using a GET Request

```
GET {{REST_SERVICES}}/record/v1/estimate/6
```

Updating an Estimate Using a PATCH Request

```
PATCH {{REST_SERVICES}}/record/v1/estimate/6
{ "altSalesTotal": 0.0, "canHaveStackable": false, "currency": { "id": "1", "refName": "USA" }, "discountItem": { "id": "-6", "ref
Name": "Partner Discount" }, "discountRate": -5.0, "discountTotal": -5.0, "dueDate": "2023-08-09", "entity": { "id": "110", "ref
Name": "Anonymous Customer" }, "entityStatus": { "id": "10", "refName": "Proposal" }, "exchangeRate": 1.0, "expectedClose
Date": "2023-07-10", "forecastType": { "id": "2", "refName": "Most Likely" }, "item": { "items": [{ "line": 1, "item":
```

```
 { "id": 98}, "rate": 39.95, "quantity": 1 }, { "line": 2, "item": { "id": 5}, "rate": 10, "quantity": 2 }] }, "subsidiary":
 { "id": "1", "refName": "Parent Company" }, "subtotal": 39.95, "toBeEmailed": false, "toBeFaxed": false, "toBePrinted": false, "to
tal": 34.95, "tranDate": "2023-07-10", "tranId": "1", "visibleToCustomer": true
}
```

Deleting an Estimate Using a DELETE Request

```
DELETE {{REST_SERVICES}}/record/v1/estimate/6
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Event

An event record exposes an event to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - Attendee
  - CalendarEventRresourceMap
  - EventFile

The REST API Browser includes information about the field names and field types of the event record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's event reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the event REST record is **calendarevent**.

Exposed Elements

The following event record elements are exposed to REST web services:

- accesslevel
- alldayevent
- company
- contact
- createddate (Read only – set to current date)
- customform
- endtime (For alldayevent, set to 6pm)
- id
- location
- message
- organizer
- owner (read only)
- relateditem
- resource
- response
- startdate
- starttime (For alldayevent, set to 8am)
- status
- supportcase
- timedevent
- title
- transaction
- usernotes

Code Sample

In the following example, <accountID> represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/calendarevent`

Request Body

```
{ "title": "Meeting with Tom", "status": "CONFIRMED", "message": "Bring Monthly Results", "location": "Caffe West", "start
date": "2022-11-02", "starttime": "14:30", "timedevent": true
}
```

To Obtain Previously-Created Event Record (where ID is 100425)

HTTP Method: GET

URL: https://<accountID>/services/rest/record/v1/calendarevent/100425

Response Body


```
{ "links": [ { "rel": "self", "href": http://<accountID>/services/rest/record/v1/calendarevent/100425 } ], "accessLevel":
 { "links": [], "id": "PUBLIC", "refName": "Public" }, "allDayEvent": false, "attendee": { "links": [ { "rel": "self", "href":
 http://<accountID>/services/rest/record/v1/calendarevent/100425/attendee } ] }, "createdDate": "2020-11-25T16:40:00Z", "cus
tomForm": { "links": [], "id": "-110", "refName": "Standard Event Form" }, "endTime": "15:30", "id": "100425", "lastModi
fiedDate": "2020-11-25T16:40:00Z", "location": "Caffe West", "message": "Bring Monthly Results", "organizer": { "links":
 [ { "rel": "self", "href": http://<accountID>/services/rest/record/v1/customer/-5 } ], "id": "-5", "refName": "A Wolfe-admin"
 }, "owner": { "links": [ { "rel": "self", "href": http://<accountID>/services/rest/record/v1/customer/-5 } ], "id": "-5", "ref
Name": "-5" } "resource": { "links": [ { "rel": "self", "href": "http://<accountID>.eng.netsuite.com/services/rest/record/v1/
calendarevent/100425/resource" } ] }, "startDate": "2022-11-02", "startTime": "14:30", "status": { "links": [], "id": "CONFIRMED",
 "refName": "Confirmed" }, "timedEvent": true, "timezone": "America/Los_Angeles", "title": "Meeting with Tom"
}
```


Expense Category

An expense category request record exposes an expense category to REST web services. Expense categories are used to group expenses. Each expense category is linked to an account. When an employee enters an expense report, they select a category for each expense, and the expense automatically posts to the associated expense account.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the expense category record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's expense category reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an expense category REST record is **expensecategory**.

Prerequisites

You must enable the Expense Reports feature before you can use this record through REST web services.

Limitations

Tax-related fields are not supported in REST.

Code Sample

This sample shows how to create the record:

```
POST /services/rest/record/v1/expenseCategory HTTP/1.1
Host: runbox.
```

```
Content-Type: application/json
Authorization: .
Content-Length: 47
{ "name": "5", "subsidiaries": "3"
}
```

This sample shows how to update the record:

```
PATCH /services/rest/record/v1/expenseCategory/2 HTTP/1.1
"name": "updated name"
}
```

Expense Report

An expense report record exposes an expense report to REST web services. An expense report transaction records an employee's expenses for approval and conversion into a bill. The expense total remains in an unapproved expense account and has no accounting impact until the expense is approved by someone with accounting authority. After an expense report is approved, a bill is created and the expense amount is reflected on the books.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the expense report record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's expense report reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an expense report REST record is **expensereport**.

Prerequisites

You must enable the Expense Reports feature before you can use this record through REST web services.

Code Sample

This sample shows how to update the record:

```
PATCH /services/rest/record/v1/expenseReport/4 HTTP/1.1
"complete": true, "tranDate": "2023-08-18", "supervisorapproval": true
}
```

Fair Value Price

Fair Value Price is a record that defines the fair value for items. Fair value price is used to allocate revenue in revenue arrangements.

For help working with this record in the UI, see the help topic Fair Value Setup.

The REST API Browser includes information about the field names and field types of the fair value price record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's fairValuePrice reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a fair value price REST record is **fairvalueprice**.

Prerequisites

This record is part of the Advanced Revenue Management (Revenue Allocation) feature. You must enable the Accounting Periods feature, Advanced Revenue Management (Essentials) feature, and Advanced Revenue Management (Revenue Allocation) feature to use advanced revenue management (revenue allocation). Before you begin working with advanced revenue management programmatically, see the help topic Advanced Revenue Management (Essentials) and (Revenue Allocation).

Limitations

The following fields on this record are required:

- currency
- fairValue
- fairValueFormula

The following fields are disabled when the **FairValueRangePolicy** field is null:

- highValue
- highValuePercent
- lowValue
- lowValuePercent

The following fields on this record are not available in REST:

- units
- unitsType

Additional Information

The **quantity** and **amount** fields appear when the **eventType** field is not set as percent complete. The **cumulativePercentComplete** field appears when **eventType** is set to percent complete.

Code Samples

The following example shows how to read a fair value price record:

Read:

```
GET {{REST_SERVICES}}/record/v1/fairvalueprice/2
```

Fulfillment Request

A fulfillment request record exposes a fulfillment request to REST web services.

A fulfillment request is created by doing a transformation on a sales order.

The status of fulfillment request can be changed. The fulfillment request exception can be added.

This record:

- is not a subrecord
- has the following subrecords:
  - FulfillmentRequestException
  - FulfillmentRequestIitem
  - FulfillmentRequestItemDetail
  - FufillmentRequestItemDetailInventoryAssignment
  - FulfillmentRequestItemException

The REST API Browser includes information about the field names and field types of the fulfillment request record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's fulfillment request reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a fulfillment request REST record is fulfillmentrequest.

Prerequisites

You must enable Fulfillment Request before you can use this record through REST web services.

Optionally, you must enable Store Pickup to use store pickup on the fulfillment request record through REST web services.

Elements Not Accessible Through REST Web Services

The following subtabs are not accessible using REST web services:

- deletionreason
- transactionnumber
- lastmodifiedby
- createdby

Code Sample

This sample shows how to transform a sales order to a fulfillment request:

```
POST -> {{COMPANY_URL}}/services/rest /record/v1/salesOrder/989?transform/fulfillmentRequest
With body: { "items": { "items": [ { "orderLine": 1, "quantity": 1 } ] }
}
Update transaction status to "In progress":
PATCH -> {{COMPANY_URL}}/services/rest/record/v1/fulfillmentrequest/992
Body: {
"tranStatus": { "id": "B" }
}
```

Gift Certificate Item

NetSuite exposes the gift certificate item record to REST web services.

You can create gift certificate items that allow customers to purchase store credit they can send to someone as a gift. The recipient uses the gift certificate code when placing an order through your Web store or entering a transaction with a sales representative. You can set a preference in NetSuite for how you want to generate the gift certificate codes: you can create them yourself, or use a random hash code automatically generated by the system.

Gift certificate codes are not active until the order used to purchase the gift certificate is billed.

For more information, see the help topic Gift Certificates.

Record ID

The record ID for the gift certificate item REST record is **giftCertificateItem**.

Prerequisite

Before you work with gift certificate items, enable the Gift Certificates feature.

Note: The gift certificate item record contains certain tax fields. If the SuiteTax feature is not enabled, you cannot create a gift certificate because of the required taxSchedule field that is annotated as a legacy tax field.

The REST API Browser includes information about the field names and field types of the estimate record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's gift certificate item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Code Samples

Creating a Gift Certificate Item

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/giftcertificateitem
{ "itemId": "TEST", "liabilityaccount": {"id": 25}
}
```

Updating a Gift Certificate Item

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/giftcertificateitem/{id}
{ "itemId": "TEST UPDATED", "upcCode": "123456"
}
```

Inbound Shipment

An inbound shipment record exposes an inbound shipment to REST web services. This record is not a subrecord. If the Inventory Detail feature is enabled, this record has one subrecord: inventorydetail.

For help working with this record in the UI, see the help topic Inbound Shipment Management.

The REST API Browser includes information about the field names and field types of the inbound shipment record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inboundShipment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an inbound shipment REST record is **inboundshipment**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Code Samples

The following samples show common use cases for inbound shipments. The example ID is 4.

Creating an Inbound Shipment From a Purchase Order

This sample shows how to create an inbound shipment from a purchase order. The shipmentitem field value should be the lineuniquekey attribute of the purchase order and not the internal id of the item.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inboundshipment
{ "shipmentstatus": "inTransit", "shipmentmemo": "Test Memo1", "items": { "items": [ { "purchaseorder": 2172, "shipmen
titem": "4827", "quantityexpected": 1 }, { "purchaseorder": 2172, "shipmentitem": "4828", "quantityexpected": 1 } ] }
}
```

Retrieving an Inbound Shipment Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inboundshipment/4
```

Updating an Inbound Shipment Using a PATCH Request

This sample shows how to update an inbound shipment to add a line.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inboundshipment/4
{ "items": { "items": [ { "purchaseorder": 2172, "shipmentitem": "4829", "quantityexpected": 1 } ] }
}
```

Deleting an Inbound Shipment Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inboundshipment/4
```

Intercompany Journal Entry

An intercompany journal entry record is a specialized type of record available only in OneWorld accounts. An intercompany journal entry records debits and credits to be posted to ledger accounts for transactions between two subsidiaries. In an account that has the Multi-Book Accounting feature enabled, you can also use this record type to create book specific intercompany journal entries.

In the user interface, you access this record as follows:

- **Intercompany journal entries** – Go to Transactions > Financial > Make Journal Entries (Administrator). For help working with this record in the user interface, see the help topic Making Intercompany Journal Entries.
- **Book specific intercompany journal entries** – Go to Transactions > Financial > Make Journal Entries (Administrator). Note that this form is available only to accounts that use Multi-Book Accounting.

For information about working with intercompany journal entries in the UI, see the help topic Making Intercompany Journal Entries.

The REST API Browser includes information about the field names and field types of the intercompany journal entry record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's intercompanyJournalEntry reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the intercompany journal entry REST record is **intercompanyjournalentry**.

The intercompany journal entry record has the following subrecords:

- accountingbookdetail
- appliedrules
- expenseplanmessage

Limitations

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "approved": true, "createdDate": "2023-08-03T06:27:00Z", "currency": { "id": "4" }, "customForm": { "id": "30" }, "exchangeR
ate": 0.01241975, "lastModifiedDate": "2023-08-03T06:27:00Z", "line": { "items": [ { "lineSubsidiary": { "id": "4" }, "account":
 { "id": "6" }, "credit": 10 }, { "lineSubsidiary": { "id": "4" }, "account": { "id": "9" }, "debit": 10 }, { "lineSubsidiary":
 { "id": "5" }, "account": { "id": "6" }, "credit": 10 }, { "lineSubsidiary": { "id": "5" }, "account": { "id": "9" }, "debit": 10
 } ] }, "memo": "memo test", "postingPeriod": { "id": "267" }, "prevDate": "2023-08-02", "reversalDefer": false, "subsidiary":
 { "id": "4" }, "toSubsidiary": { "id": "5" }, "tranDate": "2023-08-02", "tranId": "1"
}
```

Intercompany Transfer Order

In NetSuite OneWorld accounts, an inventory transfer order transaction records the change in location of inventory items from a subsidiary to a location in another subsidiary. It also records the quantity and shipping details.

This transaction is available when the Locations feature and the Multi-Location Inventory (MLI) feature are enabled. It has the following subrecord: inventory detail. For information about how intercompany transfer orders are used, see the help topic Intercompany Inventory Transfers - Non-Arm's Length.

The REST API Browser includes information about the field names and field types of the intercompany transfer order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's intercompany transfer order reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for the intercompany transfer order REST record is **intercompanytransferorder**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Usage Notes

You should read the usage notes for Intercompany Transfer Order because these notes also apply to the Intercompany Transfer Order in REST web services.

Code Samples

The following sample shows you how to add an intercompany transfer order record:

```
{{REST_SERVICES}}/record/v1/intercompanytransferorder
{ "currency": { "id": "1", "refName": "British pound" }, "incoTerm": { "links": [], "id": "1", "refName": "DAP" }, "item":
 { "items": [ { "amount": 0.0, "inventoryDetail": { "inventoryAssignment": { "items": [], "totalResults": 0 } }, "isClosed":
 false, "item": { "id": "22" }, "itemType": { "id": "InvtPart" }, "line": 1, "quantity": 25.0, "rate": 2.0 } ] }, "location":
 { "id": "1", "refName": "WMS Warehouse 1" }, "subsidiary": { "id": "1", "refName": "WMS" }, "tosubsidiary": { "id": "3", "ref
Name": "Canada" }, "transferlocation": { "id": "18", "refName": "Toronto" }
}
```

Inventory Adjustment

An inventory adjustment transaction records the change in quantity and resulting value of inventory items within a location. For example, this transaction can be used to account for clerical errors, changes in cost, thefts, or miscounts. If you use the LIFO or FIFO costing methods, this transaction preserves the quantity and value of an inventory item and preserves the costing history of the item.

This transaction is available when the Inventory feature is enabled. This record has one subrecord: inventory detail. For more details about this type of transaction, see the help topic Inventory Adjustments.

The REST API Browser includes information about the field names and field types of the inventory adjustment record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventory adjustment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an inventory adjustment REST record is **inventoryadjustment**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Sample Codes

The following example shows how to create an inventory adjustment:

```
POST {{REST_SERVICES}}/record/v1/inventoryadjustment{ "account": { "id": "1", "refName": "Checking" }, "adjLocation":
 { "id": "1", "refName": "WMS Warehouse 1" }, "customer": { "id": "19" }, "inventory": { "items": [ { "adjustQtyBy": 5.0, "item":
 { "id": "20", "refName": "Item" }, "line": 1, "location": { "id": "1", "refName": "WMS Warehouse 1" } }, { "adjustQty
By": 10.0, "item": { "id": "21", "refName": "Item2" }, "line": 2, "location": { "id": "2", "refName": "WMS Warehouse 2"
 } } ] }, "postingPeriod": { "id": "194", "refName": "Jun 2023" }, "subsidiary": { "id": "1", "refName": "WMS" }
}
```

Inventory Cost Revaluation

An inventory cost revaluation record exposes an inventory cost revaluation to REST web services. This record is not a subrecord. This record has one subrecord: inventory cost revaluation cost component.

For help working with this record in the UI, see the help topics Manually Entering an Inventory Cost Revaluation and Revalue Standard Cost Inventory.

The REST API Browser includes information about the field names and field types of the inventory cost revaluation record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventoryCostRevaluation reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an inventory cost revaluation REST record is **inventorycostrevaluation**.

The record ID for an inventory cost revaluation cost component REST subrecord is **InventoryCostRevaluationCostComponent**.

Prerequisites

You must enable the Standard Costing feature before you can use this record through REST web services.

When the feature is enabled, you can access the inventory cost revaluation record in the UI at Transactions > Inventory > Revalue Inventory Cost (Administrator).

Code Samples

The following samples show common use cases for inventory cost revaluations. The example ID is 4.

Creating an Inventory Cost Revaluation for a Non-Assembly Item

This example shows how to create an inventory cost revaluation for a non-assembly item (itemType = "InvtPart") using a POST request.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inventorycostrevaluation
{ "account": { "id": "94" }, "costComponent": { "items": [{ "cost": 12.0, "costCategory": { "id": 1 }, "items":
 [{ "cost": 10.0, "costCategory": { "id": 1 } }] }, "item": { "id": "69" }, "location": { "id": "6" }, "subsidiary": { "id": "1"
 }, "tranDate": "2023-05-01"
}
```

Creating an Inventory Cost Revaluation for an Assembly Item

This examples shows how to create an inventory cost revaluation for an assembly item (itemType = "Assembly") using a POST request. It defines additional fields for component items.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inventorycostrevaluation
{ "account": { "id": "94" }, "costComponent": { "items": [{ "cost": 12.0, "costCategory": { "id": 1 }, "componentItem": { "id": 60
 }, "quantity": 1 }, { "cost": 10.0, "costCategory": { "id": 1 }, "componentItem": { "id": 61 }, "quantity": 1 }] }, "item":
 { "id": "169" }, "location": { "id": "6" }, "subsidiary": { "id": "1" }, "tranDate": "2023-05-01"
}
```

Retrieving an Inventory Cost Revaluation Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inventorycostrevaluation/4
```

Updating an Inventory Cost Revaluation Using a PATCH Request

This sample shows how to update cost component values for an inventory cost revaluation record with a transaction id of 4. It sets the cost category ID to 3 and the cost value to 14.0.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inventorycostrevaluation/4
{ "costComponent": { "items": [{ "cost": 14.0, "costCategory": { "id": 3 } }] }
}
```

Deleting an Inventory Cost Revaluation Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/inventorycostrevaluation/4
```

Inventory Count

An inventory count transaction records the results of a physical count of items in a location. Inventory counts can initiate an inventory adjustment to update on-hand quantities based on the results.

This transaction is available when Inventory Count feature is enabled. Inventory counts for lot and serialized items require the Advanced Bin/Numbered Inventory Management feature. Inventory count records have one subrecord: count detail. For more information, see the help topic Inventory Count.

The REST API Browser includes information about the field names and field types of the inventory count record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventory count reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an inventory count REST record is **inventorycount**.

The record ID for a count detail REST subrecord is **countdetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Usage Notes

Review the following notes on working with the inventory count REST record:

- Snapshot inventory detail and adjustment inventory detail are not accessible through REST but available in NetSuite UI.
- Start, Approve, and Complete actions are not accessible through REST. Perform any of these actions on the NetSuite UI before you can perform the corresponding updates through REST.
- When an inventory count is in Started status, items cannot be added or deleted from the list.

Sample Codes

The sample codes show common use cases for inventory counts.

Creating an Inventory Count

The following example shows how to create an inventory count:

```
POST {{REST_SERVICES}}/record/v1/inventorycount { "subsidiary": {"id": 1}, "location": {"id": 1}, "item": { "items": [{ "item":
 { "id": "206" }, "binNumber": { "id": "20", "refName": "W1-Z3-Storage" } }] }
}
```

Updating an Inventory Count

The following example shows how to enter the count quantity and details of an inventory count in Started status:

```
PATCH {{REST_SERVICES}}/record/v1/inventorycount/2177 { "item": { "items": [ { "countLine": 1, "countQuantity": 4.0, "countDetail":
 { "inventoryDetail": { "items": [ { "inventoryStatus": {"id": "1" }, "quantity": 4.0 } ] } } } ] }
}
```

Inventory Item

Inventory item records are used to track information about items for which you maintain a stock.

For more information, see the help topic Items.

This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the inventory item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventory item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the inventory item REST record is **inventoryItem**.

Prerequisites

There are no prerequisites for using this record in REST.

Limitations

An inventory item REST record does not show Quantity Pricing Discount/Levels (quantitypricingdiscount, quantitypricinglevel) or the Item Translations subtab (translations).

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Code Sample

The following example shows how to get record metadata for an inventory item.

Get Record Metadata:

```
GET https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/metadata-catalog/inventoryItemHeader: Accept: applica
tion/swagger+json
```

Get Record:

```
GET https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryItem/<ID>?expandSubResources=true
```

Update Record:

```
PATCH https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryItem/<ID>
Header: Content-Type: application/json
Body:
{
   "itemId": "Updated item name",
   "upcCode: "123456"
}
```

Update Item Price When Related Features are Enabled:

```
PATCH
https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryItem/<ID>/price/currencypage=1,pricelevel=1,quantity=10
Header: Content-Type: application/json
Body:
{
   "price": 10
}
```

Inventory Number

An inventory number record uniquely identifies either an item in physical inventory with a serial number or a group of items with a lot number. This record is available when the Serialized Inventory or Lot Tracking feature is enabled. For details about these types of items, see the help topic Serial Numbered Items or Lot Numbered Items.

The REST API Browser includes information about the field names and field types of the inventory number record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventory number reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an inventory number REST record is **inventorynumber**.

Prerequisites

There are no prerequisites for using the record in REST.

Limitations

Review the following limitations when working with the inventory number REST record:

- You cannot create standalone inventory number records. The system generates these records when a serialized item or lot numbered item is created, based on the serial number or lot number values entered for the item record. See the help topic Add New Serial Numbers to Inventory or Creating Lot Numbered Items.
- You cannot delete inventory number records.
- The Inventory Number record includes a sublist that has the following fields for serial or lot numbers:
  - Issue Inventory Number (ID: issueinventorynumber) – In this field, you can select lot or serial numbers that exist in your inventory. Be sure to reference this field to select numbers for items that you include in a sales order, item fulfillment, and negative adjustments, among others.

- Receipt Inventory Number (ID: receiptinventorynumber) – In this field, you can enter lot or serial numbers for items that you want to add to your inventory. Be sure to use this field to assign numbers to items within item receipts and positive adjustments, among others.

Sample Codes

The following sample shows you how to update an inventory number record:

```
{{REST_SERVICES}}/record/v1/inventorynumber/
{ "memo": "test lot num inventory number", "expirationDate": "2023-06-19"
}
```

Inventory Transfer

An inventory transfer transaction records the change in quantity when items are transferred between locations. It also decreases the on-hand quantity in the source location and increases it in the new location.

This transaction is available when the Locations feature and the Multi-Location Inventory (MLI) feature are enabled. For more details about this type of transaction, see the help topic Transferring Inventory.

The REST API Browser includes information about the field names and field types of the inventory transfer record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's inventory transfer reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an inventory transfer REST record is **inventorytransfer**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Sample Codes

The sample codes show common use cases for inventory transfers.

Creating an Inventory Transfer

The following example shows how to create an inventory transfer:

```
POST {{REST_SERVICES}}/record/v1/inventorytransfer
{ "subsidiary": {"id": 1}, "location": {"id": 1}, "transferLocation": {"id": 2}, "inventory": { "items": [{ "item": { "id": 20
 }, "adjustQtyBy": 10 }] }
```

```
}
```

Updating an inventory transfer

The following example shows how to update an inventory transfer:

```
PATCH {{REST_SERVICES}}/record/v1/inventorytransfer/2072 { "total": 10.0, "inventory": { "items": [{ "line": 1, "adjustQty
By": 2, "inventoryDetail": { "inventoryAssignment": { "items": [ { "internalId": 7207, "inventoryDetail": 1476, "inventoryStatus":
 {"id": 1}, "toInventoryStatus": {"id": 1}, "quantity": 2.0 } ] } } }] }
}
```

Invoice

The invoice record exposes an invoice to REST web services.

To access this record in NetSuite, go to Transactions > Sales > Create Invoices.

For information about the invoice record user interface, see the help topic Invoices.

There are no prerequisites for using this record through REST web services.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the invoice record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's invoice reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an invoice REST record is **invoice**.

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show common use cases for invoices.

In the following examples,

- <accountID> represents your account ID.
- <recordKey> represents the database key for the invoice.

Creating a New Invoice With an Item

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/invoice { "asofdate": "2021-03-15", "end
date": "2021-06-15", "entity": { "id": "220" }, "item": { "items": [ { "amount": 1000.0, "item": { "id": "144" } } ] }, "start
date": "2021-03-15", "subsidiary": { "id": "1" }, "terms": { "id": "1" }
```

Creating a New Invoice With a Billable Item

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/invoice { "asofdate": "2021-03-15", "entity":
 { "id": "221" }, "itemcost": { "items": [ { "apply": true, "doc": { "id": "310" }, "line": 1 } ] }
}
```

Creating a New Invoice With Billable Time

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/invoice { "entity": { "id": "220" }, "time": { "items":
 [ { "apply": true, "doc": { "id": "1" }, "rate": 10 } ] }
}
```

Updating an Existing Invoice

Note: The transaction date and rate are in line item 1.

```
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/<recordKey> { "item": { "items":
 [ { "line": 1, "rate": 50.00 } ] }, "tranDate": "2021-03-05"
}
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Issue

Issue is used to document and manage issues that occur with the products you manufacture and sell. In the UI, you can log issues at Issues > Issues > Issues > New.

Use the issue record to create new issues and expose them to REST web services.

To learn more, see the help topic Logging Issues.

This record:

- is not a subrecord
- has no subrecords

There are no prerequisites for using this record through REST web services.

The REST API Browser includes information about the field names and field types of the issue record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's issue reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an issue REST record is **issue**.

Code Sample

The following samples show common use cases for issue.

Creating an Issue

```
POST https:// {{REST_SERVICES}} /services/rest/record/v1/issue
Body: { "duplicateStatus": "F", "emailAssignee": false, "isOwner": true, "isShowstopper": false, "issueAbstract": "Issue
 abstract 1701112365", "issueNumber": "issue_1701112365", "issueStatus": { "id": "1", "refName": "Submitted" }, "issueTags":
 { "count": 0, "hasMore": false, "items": [], "offset": 0, "totalResults": 0 }, "issueType": { "id": "1", "refName": "Problem"
 }, "mediaItem": { "count": 0, "hasMore": false, "items": [], "offset": 0, "totalResults": 0 }, "priority": { "id": "1", "ref
Name": "P1" }, "product": { "id": "103" }, "relatedIssues": { "items": [ ], "totalResults": 0 }, "reportedBy": { "id": "-5", "ref
Name": "Mark Wheel" }, "reproduce": { "id": "1", "refName": "Test Environment" }, "severity": { "id": "1", "refName": "S1 - Block
ing" }, "source": { "id": "1", "refName": "Internal" }, "statusType": "OPEN", "targetVersion": { "items": [], "totalResults": 0
 }, "trackCode": { "id": "1", "refName": "Never" }
}'
```

Retrieving an Issue

```
GET
https:// {{REST_SERVICES}} /services/rest/record/v1/issue
```

Updating an issue

```
PATCH https:// {{REST_SERVICES}} /services/rest/record/v1/issue/{{ISSUE_NUMBER}}
Body:
{ "duplicateStatus": "F", "emailAssignee": false, "isOwner": true, "isShowstopper": false, "issueAbstract": "Issue abstract
 1701112438", "issueNumber": "issue_1701112438", "issueStatus": { "id": "1", "refName": "Submitted" }, "issueTags":
```

```
 { "count": 0, "hasMore": false, "items": [], "offset": 0, "totalResults": 0 }, "issueType": { "id": "1", "refName": "Problem"
 }, "mediaItem": { "count": 0, "hasMore": false, "items": [], "offset": 0, "totalResults": 0 }, "priority": { "id": "1", "ref
Name": "P1" }, "product": { "id": "103" }, "relatedIssues": { "items": [ { "issuenumber": "303", "issuestatus": "Submitted", "re
lationship": { "id": "M3", "refName": "Blocks" } } ], "totalResults": 1 }, "reportedBy": { "id": "-5", "refName": "Mark Wheel"
 }, "reproduce": { "id": "1", "refName": "Test Environment" }, "severity": { "id": "1", "refName": "S1 - Blocking" }, "source":
 { "id": "1", "refName": "Internal" }, "statusType": "OPEN", "targetVersion": { "items": [], "totalResults": 0 }, "trackCode":
 { "id": "1", "refName": "Never" }
}'
```

Item Fulfillment

An item fulfillment transaction records the shipment of some or all items on an order to the customer. The processes for item fulfillment transactions depend on whether the Advanced Shipping feature is enabled.

- If Advanced Shipping is not enabled, the fulfillment and invoicing processes are combined. When an item fulfillment is created, a related invoice is created at the same time.
- If Advanced Shipping is enabled, fulfillment and invoicing are two independent processes, and shipments can be recorded separately from billing.

For more information about item fulfillment entries, see the help topic Order Fulfillment.

This record:

- ItemFulfillment is not a subrecord.
- InventoryDetail is the subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the item fulfillment record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's item fulfillment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the item fulfillment REST record is **ItemFulfillment**.

Prerequisites

There are no prerequisites for using this record in REST.

Code Sample

The following example shows how to create an item fulfillment from a sales order record.

```
POST https://<accountID>/services/rest/record/v1/salesorder/{{SALES_ORDER_ID}}/!transform/itemFulfillment
{ "item":{ "items":[ { "orderLine":1, "location":6, "itemreceive": true } { "orderLine": 3 "location": 6, "itemreceive":
 false } ] }
}
```

Item Group

An item group is sold as a single unit but made up of several individual items. An item group is a group of items on a sales or purchase order that need to be sold or purchased together. The item group is not fulfilled, received, or stocked. However, the item group is available in the item list on a sales or purchase order and can be added to those transactions.

For more information, see the help topic Item Groups.

The item group record has one subrecord: **member**.

The REST API Browser includes information about the field names and field types of the item group record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's itemgroup reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the item group REST record is **itemgroup**.

Inaccessible Elements

The **translations** sublist is not exposed to REST web services.

Code Sample

Creating an Item Group

```
POST {{COMPANY_URL}}/services/rest/record/v1/itemGroup { "itemId": "ItemGroup REST TEST", "member": { "items": [ { "item":
 { "id":"233" }, "quantity":1 } ] }
}
```

Getting an Existing Item Group by ID

```
GET {{COMPANY_URL}}/services/rest/record/v1/itemGroup/{{itemGroupId}}
```

Updating an Existing Item Group by ID

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/itemGroup/{{itemGroupId}} { "itemId": "ItemGroup REST TEST updated", "member":
 { "items": [ { "lineNumber": 1, "quantity":2 } ] }
}
```

Item Receipt

An item receipt transaction records the receipt of items from orders or returns that arrive at your location. How you receive items depends on whether you use the Advanced Receiving feature, as described in the following:

- If you do not use Advanced Receiving, the receiving and billing processes are combined. When you receive an item, you create a vendor bill for it simultaneously.
- If you prefer to have separate processes to receive items and create vendor bills, you can use the Advanced Receiving feature.

  With Advanced Receiving, you can use separate processes to receive items separately from billing items. Then, you can receive parts of an order before creating a bill for the whole order.
- If you use return authorizations, you can also receive authorized returns in parts using Advanced Receiving and the Item Receipt page.

This transaction is available when the Locations feature and the Multi-Location Inventory (MLI) feature are enabled. The item receipt record has one subrecord: inventory detail. For more information about item receipts, see the help topic Receiving Orders.

The REST API Browser includes information about the field names and field types of the item receipt record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's item receipt reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an item receipt REST record is **itemreceipt**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Usage Notes

You should read the usage notes on initializing item receipts from the supported transactions: Item Receipt.

Sample Codes

The following sample codes show you how to create an item receipt from a purchase order record. It uses the following details for a sample purchase order:

- Purchase order internal ID – 2173
- Purchase order item quantity – 4

```
POST {{REST_SERVICES}}/record/v1/purchaseorder/2173/!transform/itemreceipt { "item": { "items": [ { "line": 1, "inventoryde
tail":{ "inventoryassignment": { "items": [{ "receiptInventoryNumber": "L1", "inventorystatus": 1, "binnumber": 20, "quantity":4
 } ] } } } ] }
}
```

Item Revision

An item revision record exposes an item revision item to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - component
  - operation

The REST API Browser includes information about the field names and field types of the item revision record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's itemRevision reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an item revision REST record is **ItemRevision**.

Code Samples

The following samples show common use cases for item revisions. The example ID is 4.

Creating an Item Revision Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/itemRevision
{ "item": {"id":410}, "name": "REV_2022", "effectiveDate": "2022-01-01", "memo": "memo"
}
```

Retrieving an Item Revision Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/itemRevision/4
```

Updating an Item Revision Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/itemRevision/4
{ "memo": "new memo"
}
```

Job

A job record exposes a job (project) status to REST web services.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the job record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's job reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a job REST record is **job**.

Prerequisites

You must enable the Projects feature before you can use this record through REST web services.

Limitations

Tax-related fields are not supported in REST.

The following fields are not supported in REST:

- currencyPrecision
- bBudgetShowCalculatedLines
- cBudgetShowCalculatedLines
- customer
- originatingDoc
- originatingLine
- originatingActualLine
- originatingItem
- projectTasks_count
- isBudgetRebuilding
- revenueArrangement
- primaryContact
- alternateContact
- baselineBudget
- estimateAtCompletionBudget

The following sublists are not supported in REST:

- ProjectResource
- ProjectBillingBudget
- ProjectCostBudget
- Work Assignments

Code Sample

This sample shows how to update the record:

```
POST /services/rest/record/v1/job
{ "subsidiary": "1", "entityId": "title", "projectExpenseType": "-2"
```

```
}
```

Job Status

A job status record exposes a job (project) status to REST web services. The job status record indicates the progress of the project.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the job status record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's job status reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a job status REST record is **jobstatus**.

Prerequisites

You must enable the Projects feature before you can use this record through REST web services.

Limitations

Tax-related fields are not supported in REST.

Code Sample

This sample shows how to update the record:

```
PATCH /services/rest/record/v1/jobstatus/17 {
"name": "Updated name", "description": "desc"
}
```

Job Type

A job type record exposes a job (project) type to REST web services. Project types are used to define basic project records if the Projects feature is enabled, and the Project Management feature is not enabled. You can use the Projects feature to track information about projects you are working on for customers, including time, contacts, and transactions.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the job type record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's job type reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an job type REST record is **jobtype**.

Prerequisites

You must enable the Projects feature before you can use this record through REST web services.

Limitations

Tax-related fields are not supported in REST.

Code Sample

This sample shows how to update the record:

```
PATCH /services/rest/record/v1/jobtype/1 {
"name": "updated name", "isInactive": true
}
```

Journal Entry

You use the journal entry record to adjust balances in accounts. Journal entries let you change the value of any set of accounts without having to enter a posting transaction. In an account that has the Multi-Book Accounting feature enabled, you can also use this record to create book specific journal entries.

For more information about journal entries in NetSuite, see the help topic Journal Entries.

This record:

- is not a subrecord
- has the following subrecords:
  - journalentryline
  - accountingbookdetail

The REST API Browser includes information about the field names and field types of the journal entry record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's journal entry reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the journal entry REST record is **journalentry**.

Prerequisites

There are no prerequisites for using the record in REST.

Limitations

- Tax-related fields are not supported in REST.
- The accounting book detail sublist is part of the Multi-Book Accounting feature (and its related features). For more information, see the help topic Multi-Book Accounting.
- Read-only sublists, including system notes and GL impact, are not supported in REST.

Code Sample

```
{
"subsidiary": { "id": "1" }, "line": { "items": [ { "account": { "id": "58" }, "debit": 187, "memo": "Journal Example" },
 { "account": { "id": "4" }, "credit": 187 }} ]
}
```

Kit Item

A kit item record exposes a kit item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the kit item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's kitItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record IDs

The record ID for a kit item REST record is **kitItem**.

Elements with Different Functionality

The record has a matrix pricing machine. Otherwise, it behaves in the same way as the inventory item.

Code Samples

The following samples show common use cases for kit items. The example ID is 4.

Creating a Kit Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/kitItem
```

```
{ "itemId": "Kit A", "member": { "items": [ { "item": { "id": "71", "refName": "Invt_Item_1" }, "quantity": 1.0 } ] }
}
```

Retrieving a Kit Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/kitItem/4
```

Updating a Kit Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/kitItem/4
{ "itemId": "Kit B", " description": "Kit Item description",
}
```

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/kitItem/4
{ "item": { "id": "71" }, "quantity": 5.0
}
```

Deleting a Kit Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/kitItem/4
```

Location

Locations can be used to identify and categorize records in your NetSuite account. You can also define locations for your organization's needs by creating custom segments. Locations enable you to better organize data and preserve accuracy.

For information about working with locations in the UI, see the help topic Classifications in NetSuite.

The REST API Browser includes information about the field names and field types of the location record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's location reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the location REST record is **location**.

The location record has the following subrecords:

- classTranslation
- businessHours (as part of the Store Pickup feature)
- mainAddress
- inventoryBalance
- returnAddress
- tranNumbering
- docNumbering

- includeLocationRegions (as part of the Automatic Location Assignment feature)
- excludeLocationRegions (as part of the Automatic Location Assignment feature)

Limitations

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "isInactive": false, "name": "REST test 1692272155", "subsidiary": { "items": [ { "id": 2 }] } , "makeInventoryAvailableStore":
 true, "latitude": 90, "tranPrefix": "tp", "defaultAllocationPriority": 112, "locationType": 1, "longitude": 180, "makeInvento
ryAvailable": true, "timeZone": { "id": "Asia/Dacca"}, "logo": { "id": 14 } }
```

Manufacturing Cost Template

A manufacturing cost template record exposes a manufacturing cost template to REST web services.

This record:

- is not a subrecord
- has one subrecord: costDetail

The REST API Browser includes information about the field names and field types of the manufacturing cost template record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's manufacturingCostTemplate reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a manufacturing cost template REST record is **manufacturingCostTemplate**.

Prerequisites

You must enable the Manufacturing Routing and Work Center features before you can use this record through REST web services.

Code Samples

The following samples show common use cases for manufacturing cost templates. The example ID is 4.

Creating a Manufacturing Cost Template Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingCostTemplate
{
    "subsidiary": { "id": "5"},
    "name": "rest mfg cost template",
    "costDetail": {
```

```
        "items": [
            {
                "costCategory": {"id": 5},
                "item": {"id": 71}
            },
            {
                "costCategory": {"id": 3},
                "item": {"id": 69}
            }
        ]
    }
}
```

Retrieving a Manufacturing Cost Template Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingCostTemplate/4
```

Updating a Manufacturing Cost Template Using a PATCH Request

The following code sample shows how to edit an item on the cost detail line. The component line ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRoutingCostTeamplate/4/costDetail/7
}
```

Manufacturing Operation Task

A manufacturing operation task record exposes a manufacturing operation task to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - costDetail
  - predecessor

The REST API Browser includes information about the field names and field types of the manufacturing operation task record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's manufacturingOperationTask reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a manufacturing operation task REST record is **manufacturingOperationTask**.

Prerequisites

You must enable the Manufacturing Routing and Work Center features before you can use this record through REST web services.

Code Samples

The following samples show common use cases for manufacturing operation tasks. The example ID is 4.

Creating a Manufacturing Operation Task Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingOperationTask
{
   "manufacturingWorkCenter" : {"id":"39"},
   "manufacturingCostTemplate" : {"id": 4},
   "title": "My operation",
   "operationSequence": 60,
   "setupTime": 6,
   "runRate": 4,
   "machineResources": 1,
   "laborResources": 1,
   "workOrder": {
       "id": 159
   }
}
```

Retrieving a Manufacturing Operation Task Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingOperationTask/4
```

Retrieving a Predecessor Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingOperationTask/238/predecessor
```

Updating a Manufacturing Operation Task Using a PATCH Request

The following code sample shows how to add an entry to the cost detail sublist.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingOperationTask/4
{
    "costDetail": {
        "items": [
            {
                "costCategory" : {"id": 1},
                "item" : {"id": 72}
            }
        ]
    }
}
```

Manufacturing Routing

A manufacturing routing record exposes a manufacturing routing to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - routingComponent
  - routingStep

The REST API Browser includes information about the field names and field types of the manufacturing routing record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's manufacturingRouting reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a manufacturing routing REST record is **manufacturingRouting**.

Prerequisites

You must enable the Manufacturing Routing and Work Center features before you can use this record through REST web services.

Additional Details

When you use a POST request to create a Manufacturing Routing, the routingComponent machine is indexed by the "lineNumber" fieldID. When you update the fieldID, the routingComponent machine is indexed by the "component" fieldID.

Code Samples

The following samples show common use cases for manufacturing routings. The example ID is 4.

Creating a Manufacturing Routing Using a POST Request

The following code sample shows how to create a manufacturing routing with routing steps and set a value for the operation on components using the operation sublist.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRouting
{ "name": "routing via rest", "billOfMaterials": {"id": 12}, "subsidiary": {"id": 5}, "location": { "items": [ {"id": 5} ] }, "routingstep": { "items": [ { "operationsequence": 10, "operationName": "10", "manufacturingWorkCenter": {"id": 40}, "manufacturingCostTemplate": {"id": 4}, "setupTime": 5, "runRate": 10 }, { "operationsequence": 20, "operationName": "20", "manufacturingWorkCenter": {"id": 40}, "manufacturingCostTemplate": {"id": 4}, "setupTime": 5, "runRate": 10, "lagType": {"id": "time"}, "lagAmount": 9 } ]}, "routingComponent": { "items": [ { "lineNumber": 1, "operationSequenceNumber": 10 }, { "lineNumber": 2, "operationSequenceNumber": 20 } ] } }
{ "itemId": " Other Charge for Purchase", "displayName": "displayName", "upcCode": "code" }
```

Retrieving a Manufacturing Routing Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRouting/4
```

Updating a Manufacturing Routing Using a PATCH Request

The following code sample shows how to edit a component line using the operation sublist. The component line ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRouting/4/routingComponent/7
{ "operationsequenceNumber": 10
}
```

The following code sample shows how to edit the setup time on the routing step sublist.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRouting/4/routingStep/10
{ "setupTime": 25
}
```

Deleting a Manufacturing Routing Using a DELETE Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/manufacturingRouting/4
```

Markup Item

A markup item record exposes a markup item to REST web services. To access this record in NetSuite, go to Lists > Accounting > Items > New > Markup.

The REST API Browser includes information about the field names and field types of the markup item record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's markupItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the markup item REST record is **markupitem**.

Code Samples

These samples show common use cases for markup items. The example ID is 6.

Creating a Markup Item Using a POST Request

```
{
   "itemId": "New Markup",
   "itemType": {
      "id": "Markup",
      "refName": "Markup"
   },
   "nonPosting": true,
   "rate": 60
}
```

Retrieving a Markup Item Using a GET Request

```
GET: {{REST_SERVICES}}/record/v1/markupitem/6
```

Updating a Markup Item Using a PATCH Request

```
PATCH: {{REST_SERVICES}}/record/v1/markupitem/6

{
   "itemId": "Newer Markup Item",
   "rate": 80
}
```

Deleting a Markup Item Using a DELETE Request

```
DELETE: {{REST_SERVICES}}/record/v1/markupitem/6
```

Message

A message record exposes a message to REST web services.

This record has the following subrecords:

- MessageFile
- CcBccRecipient

The REST API Browser includes information about the field names and field types of the message record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's message reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the message REST record is **Message**.

Exposed Elements

The following message record elements are exposed to REST web services:

- activity
- author
- authoremail
- bcc
- cc
- emailed
- entity

- externalid
- hasattachment
- htmlmessage
- id
- incoming
- lastmodifieddate
- message
- messagedate
- recipient
- recipientemail
- record
- recordtype
- subject
- template
- time
- transaction

Usage Notes

All fields are read-only.

Code Sample

In the following example, <accountID> represents your account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/message`

Request Body

```
{ "subject": "Meeting agenda", "message": "Please send the agenda for tomorrow's meeting.", "author": "-5", "authorE
mail": "sender@your.example.com", "recipientEmail": "recipient@your.example.com", "cc": "cc_recipient@your.example.com", "bcc": "bc
c_recipient@your.example.com"
}
```

To Obtain Previously-Created Message Record (where ID is 230)

HTTP Method: GET

URL: `https://<accountID>/services/rest/record/v1/message/230`

Response Body

```
{ "links": [ { "rel": "self", "href": http://<accountID>/services/rest/record/v1/message/230 } ], "author": { "links":
 [ { "rel": "self", "href": http://<accountID>/services/rest/record/v1/customer/-5 } ], "id": "-5", "refName": "A Wolfe-admin"
```

```
 }, "authorEmail": "sender@your.example.com", "bcc": "bcc_recipient@your.example.com", "cc": "cc_recipient@your.example.com", "e
mailed": true, "hasAttachment": false, "htmlMessage": true, "htmlMessage": true, "id": "230", "incoming": true, "lastModified
Date": "2020-11-25T16:32:00Z", "message": "Please send the agenda for tomorrow's meeting.", "messageDate": "2020-11-25", "recipien
t": { "links": [ { "rel": "self", "href": "<accountID>/services/rest/record/v1/customer/-5" } ], "id": "-5", "refName": "A Wolfe-
admin" }, "recipientEmail": "recipient@your.example.com", "subject": "Meeting agenda", "time": "08:32"
}
```


Merchandise Hierarchy Level

A merchandise hierarchy level record exposes a merchandise hierarchy level to REST web services. This record is not a subrecord and does not contain a subrecord.

The REST API Browser includes information about the field names and field types of the merchandise hierarchy level record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's merchandise hierarchy level reference page.

For information on using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the merchandise hierarchy level REST record is **merchandisehierarchylevel**.

Prerequisites

You must enable the Merchandise Hierarchy feature before you can use this record through REST web services. This feature depends on the Custom Segments feature.

Code Sample

Creating a Merchandise Hierarchy Level

```
POST {{REST_SERVICES}}/record/v1/merchandisehierarchylevel
{
    "name": "Level 2",
    "description": "L2",
    "insertbefore": {
        "id": 1
    }
}
```

Updating a Merchandise Hierarchy Level

```
PATCH {{REST_SERVICES}}/record/v1/merchandisehierarchylevel/{{id}}
{
    "name": "Level 2 updated",
    "description": "L2 updated",
    "insertbefore": {
        "id": 1
    }
}
```

Merchandise Hierarchy Node

A merchandise hierarchy node record exposes a merchandise hierarchy node to REST web services. This record is not a subrecord. This record contains one subrecord with the ID **hierarchyversions**.

The REST API Browser includes information about the field names and field types of the merchandise hierarchy node record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's merchandise hierarchy node reference page.

For information on using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the merchandise hierarchy node REST record is **merchandisehierarchynode**.

Prerequisites

You must enable the Merchandise Hierarchy feature before you can use this record through REST web services. This feature depends on the Custom Segments feature.

Code Sample

Creating a Merchandise Hierarchy Node

```
POST {{REST_SERVICES}}/record/v1/merchandisehierarchynode
{
    "name": "Node 2",
    "description": "N2"
}
```

Updating a Merchandise Hierarchy Node

```
PATCH {{REST_SERVICES}}/record/v1/merchandisehierarchynode/{{id}}
{
    "name": "Node 2 updated",
    "description": "N2 updated",
    "hierarchyversions":{
        "items":
        [
            {
                "hierarchylevel":{
                    "id": 3
                },
                "hierarchyversion":{
                    "id": 2
                },
                "parentnode": 1,
                "isincluded": true
            }
        ]
    }
}
```

Merchandise Hierarchy Version

A merchandise hierarchy version record exposes a merchandise hierarchy version to REST web services. This record is not a subrecord. This record contains one subrecord with the ID **hierarchylevels**.

The REST API Browser includes information about the field names and field types of the merchandise hierarchy version record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's merchandise hierarchy version reference page.

For information on using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the merchandise hierarchy node REST record is **merchandisehierarchyversion**.

Prerequisites

You must enable the Merchandise Hierarchy feature before you can use this record through REST web services. This feature depends on the Custom Segments feature.

Code Sample

Creating a Merchandise Hierarchy Version

```
POST {{REST_SERVICES}}/record/v1/merchandisehierarchyversion
{
    "name": "Version 1",
    "description": "V1",
    "startdate": "2024-09-01"
}
```

Updating a Merchandise Hierarchy Version

```
PATCH {{REST_SERVICES}}/record/v1/merchandisehierarchyversion/{{id}}
{
    "name": "Version 1 updated",
    "description": "V1 updated",
    "startdate": "2024-09-02"
}
```

Nexus

NetSuite uses the term nexus to describe a tax jurisdiction or geographic area where you do business and which has its own tax regulations. For example, a nexus can be a country, municipality, city, state, province, county, or economic zone.

For more information, see the help topic Understanding Nexuses in SuiteTax.

This record is not a subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the nexus record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's nexus reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the nexus REST record is **nexus**.

Prerequisites

The SuiteTax feature must be enabled.

Actions

The nexus record has the following actions through REST web services:

- Submit new
- Submit
- Reset
- Delete

Code Samples

This code sample shows how to post the nexus record:

```
 POST /services/rest/record/v1/nexus { "country": "US", "state": "CA", "description": "California", "parentNexus": "1", "taxA
gency": "2"
}
```

This code sample shows how to get the nexus record:

```
GET /services/rest/record/v1/nexus/13 { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-deuf1-001.eng.net
suite.com/services/rest/record/v1/nexus/13" } ], "country": { "id": "US", "refName": "United States" }, "description": "Califor
nia", "id": "13", "isInactive": false, "parentNexus": { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-
deuf1-001.eng.netsuite.com/services/rest/record/v1/nexus/1" } ], "id": "1", "refName": "US California" }, "state": { "links":
 [], "id": "CA", "refName": "California" }, "taxAgency": { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-
deuf1-001.eng.netsuite.com/services/rest/record/v1/vendor/2" } ], "id": "2", "refName": "Tax Agency CA" }, "taxDateFromFulfill
ment": false
}
```

Non-Inventory Purchase Item

Non-inventory items for purchase can only be bought and entered on vendor-facing transactions such as purchase orders and vendor bills. These items are purchased by your organization but not resold, such as office supplies.

To learn more, see the help topic Non-Inventory Items.

This record is not a subrecord and it does not have a subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the non-inventory purchase items record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's non-inventory purchase item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser

Record ID

The record ID for the non-inventory purchase item record is **noninventorypurchaseitem**.

Actions

You can convert a non-inventory purchase item to an inventory item.

Code Sample

Creating a Non-Inventory Purchase Item

```
POST {{COMPANY_URL}}/services/rest/record/v1/noninventorypurchaseitem { "itemId": "NONINVENTORYPURCHASEITEM_ID"
}
```

Getting a Non-Inventory Purchase Item by ID

```
GET {{COMPANY_URL}}/services/rest/record/v1/noninventorypurchaseitem/{{nonInvtPurchaseItemId}} { "itemId": "NEW_NONINVENTORYPURCHASEITEM_ID"
}
```

Deleting a Non-Inventory Purchase Item

```
DELETE {{COMPANY_URL}}/services/rest/record/v1/noninventorypurchaseitem/{{nonInvtPurchaseItemId}}
```

Non-Inventory Resale Item

Non-inventory resale item records are used to track something you buy and then sell for a profit, but do not stock. Non-inventory items for resale can be bought and sold and appear on all applicable transaction types. This includes drop-ship items that you do not store but sell directly from the vendor.

To learn more, see the help topic Non-Inventory Items.

This record does not have a subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the non-inventory resale item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's non-inventory resale item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the non-inventory resale item REST record is **noninventoryresaleitem**.

Inaccessible Elements

ItemImages is not accessible in REST web services.

Elements with Different Functionality

The record has a matrix pricing machine. Otherwise, it behaves in the same way as the non-inventory purchase item and non-inventory resale item records.

Actions

You can convert a non-inventory resale item to an inventory item.

Code Sample

Creating a Non-Inventory Resale Item

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/itemGroup/ noninventoryresaleitem { "itemId": "52", "price": { "items":
 [{ "priceLevel": { "id": "1" }, "quantity": { "value": "5" }, "currencypage": { "id": "1" }, "price": 100 }] }, "costEsti
mate": 64200.0
}
```

Getting a Non-Inventory Resale Item

```
GET {{COMPANY_URL}}/services/rest/record/v1/noninventoryresaleitem/{{nonInventoryResaleItemId}}
```

Finding a Non-Inventory Resale Item by itemid

```
GET {{COMPANY_URL}}/services/rest/record/v1/noninventoryresaleitem?q=itemId IS MY-CUSTOM-UNIQUE-ITEM-ID

Query parameters

q={{expression}}
```

```

possible expression: field {{operator}} field's value

example: itemId IS MY-CUSTOM-UNIQUE-ITEM-ID
```

Updating an Existing Non-Inventory Resale Item

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/ noninventoryresaleitem/{{nonInventoryResaleItemId}} { "price": { "items":
 [{ "priceLevel": { "id": "1" }, "quantity": { "value": "5" }, "currencypage": { "id": "1" }, "price": 200 }] }, "costEsti
mate": 379,
}
```

Deleting a Non-Inventory Resale Item

```
DELETE {{COMPANY_URL}}/services/rest/record/v1/ noninventoryresaleitem/{{nonInventoryResaleItemId}}
```

Non-Inventory Sale Item

You can record and track items that you always drop ship as non-inventory items. You can also record and track other items that you sell or purchase but do not stock as non-inventory items.

For more information, see the help topic Non-Inventory Items.

This record is not a subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the non-inventory sale item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's non-inventory sale item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the non-inventory sale item REST record is **noninventorySaleItem**.

Prerequisites

There are no prerequisites for using this record in REST.

Code Samples

The following example shows how to get record metadata for a non-inventory sale item.

Get Record Metadata:

```
GET https://<accountID>/services/rest/record/v1/metadata-catalog/nonInventorySaleItem
```

```
Header: Accept: application/swagger+json
```

Get Record:

```
GET https://<accountID>/services/rest/record/v1/nonInventorySaleItem/<ID>
```

Update Record:

```
PATCH https://<accountID>/services/rest/record/v1/nonInventorySaleItem/<ID>
Header: Content-Type: application/json
Body:
{
      "itemId": "Updated item name",
      "upcCode: "123456"
}
```

Update Item Price When Related Features are Enabled:

```
PATCH
https://<accountID>/services/rest/record/v1/nonInventorySaleItem/<ID>/price/currencypage=1,pricelevel=1,quantity=10
Header: Content-Type: application/json
Body:
{
   "price": 10
}
```

Note Type

Note type defines a list of values that are used by the note record to set the type of note. In the UI, this is a user defined list at Setup > Sales > Setup Tasks > CRM Lists.

Use the note type record to create new note types and expose them to REST web services.

To learn more, see the help topic Note Type.

This record:

- is not a subrecord
- has no subrecords

There are no prerequisites for using this record through REST web services.

The REST API Browser includes information about the field names and field types of the note type record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's note type reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a note type REST record is **notetype**.

Code Sample

The following samples show common use cases for notetype.

Creating a Note Type

```
POST https://{{REST_SERVICE}}/services/rest/record/v1/notetype
Body: {
"description": "NoteType description 1696527953", "isInactive": false, "name": "NoteType 1696527953", "externalId": "ex
t_1696527953",
}'
```

Retrieving a Note Type

```
GET
https://{{REST_SERVICE}}/services/rest/record/v1/notetype/209?expandSubResources=true
```

Updating a Note Type

```
PATCH https://REST_SERVICE/services/rest/record/v1/notetype/209
Body:
{ "description": "Updated NoteType description 1696528478", "isInactive": true, "name": "Update NoteType 1696528478"
}'
```

Deleting a Note Type

```
DELETE https://REST_SERVICE/services/rest/record/v1/notetype/209
```

Opportunity

An opportunity record exposes an opportunity to REST web services. This record is not a subrecord. This record has one subrecord: record ID item.

The REST API Browser includes information about the field names and field types of the opportunity record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's opportunity reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for an opportunity REST record is **opportunity**.

Limitations

When updating an existing Opportunity record through REST, do not include the "line" element when adding an item line.

Incorrect

```
"item": { "items": [{ "item": { "id": 5 }, "line": 1, "quantity": 2.0, "rate": 12.78 }]
```

```
}
```

Correct

```
"item": { "items": [{ "item": { "id": 5 }, "quantity": 2.0, "rate": 12.78 }]
}
```

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show typical use cases for opportunity. The sample ID is 132.

Creating an Opportunity Using a POST Request

```
https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/opportunity
{ "altSalesRangeHigh": 0.0, "altSalesRangeLow": 0.0, "altSalesTotal": 0, "canHaveStackable": false, "competitors": { "items":
 [ { "competitor": { "id": "1" } } ] }, "currency": { "id": "1" }, "entity": { "id": "110" }, "entityStatus": { "id": "17"
 }, "exchangeRate": 1.0, "expectedCloseDate": "2023-07-13", "forecastType": { "id": "2" }, "isBudgetApproved": false, "item":
 { "items": [ { "fromJob": false, "isEstimate": false, "item": { "id": "98" }, "marginal": false, "printItems": false, "quan
tity": 4.0, "rate": 39.95 }, { "fromJob": false, "isEstimate": false, "item": { "id": "5" }, "marginal": false, "printItem
s": false, "rate": 954.21 } ] }, "leadSource": { "id": "-2" }, "prevDate": "2023-07-13", "probability": 10.0, "projAltSale
sAmt": 0.0, "projectedTotal": 100.0, "rangeHigh": 100.0, "rangeLow": 100.0, "salesTeam": { "items": [ { "contribution": 100.00, "em
ployee": { "id": "40" }, "isPrimary": false, "salesRole": { "id": "-2" } }, { "contribution": 100.00, "employee": { "id": "134"
 }, "isPrimary": true, "salesRole": { "id": "3" } } ] }, "shipIsResidential": false, "shipOverride": false, "subsidiary":
 {"id": "1" }, "tranDate": "2023-07-13", "weightedTotal": 10.0
}
```

Retrieving an Opportunity Using a GET Request

```
GET https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/opportunity/132
```

Updating an Opportunity Using a PATCH Request

```
https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/opportunity/132
{ "altSalesRangeHigh": 0.0, "altSalesRangeLow": 0.0, "altSalesTotal": 0, "canHaveStackable": false, "competitors":
 { "items": [ { "competitor": { "id": "1" } } ] }, "currency": { "id": "1" }, "entity": { "id": "110" }, "entityStatus":
 { "id": "17" }, "exchangeRate": 1.0, "expectedCloseDate": "2023-07-13", "forecastType": { "id": "2" }, "isBudgetAp
proved": false, "item": { "items": [ { "description": "Ergonomic Keyboard", "fromJob": false, "isEstimate": false, "item":
 { "id": "98" }, "line": 1, "marginal": false, "printItems": false, "quantity": 4.0, "rate": 39.95 }, { "description": "NonIn
vItem", "fromJob": false, "isEstimate": false, "item": { "id": "5" }, "marginal": false, "printItems": false, "rate": 954.21
 } ] }, "leadSource": { "id": "-2" }, "prevDate": "2023-07-13", "probability": 10.0, "projAltSalesAmt": 0.0, "projected
```

```
Total": 100.0, "rangeHigh": 100.0, "rangeLow": 100.0, "salesTeam": { "items": [ { "contribution": 100.00, "employee":
 { "id": "40" }, "isPrimary": false, "salesRole": { "id": "-2" } }, { "contribution": 100.00, "employee": { "id": "134" }, "isPri
mary": true, "salesRole": { "id": "3" } } ] }, "shipIsResidential": false, "shipOverride": false, "subsidiary": {"id": "1"
 }, "tranDate": "2023-07-13", "weightedTotal": 10.0
}
```

Deleting an Opportunity Using a DELETE Request

```
DELETE https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/opportunity/132
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Other Charge for Purchase Item

An other charge for purchase item record exposes an other charge for purchase item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the other charge for purchase item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's otherChargePurchaseItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record IDs

The record ID for an other charge for purchase item REST record is **otherChargePurchaseItem**.

Code Samples

The following samples show common use cases for other charge for purchase items. The example ID is 4.

Creating an Other Charge for Purchase Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargePurchaseItem
{ "itemId": " Other Charge for Purchase", "displayName": "displayName", "upcCode": "code" }
```

Retrieving an Other Charge for Purchase Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargePurchaseItem/4
```

Updating an Other Charge for Purchase Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargePurchaseItem/4
{ "itemId": "Payment Item B", "displayName": "Payment Item B DisplayName", "description": "description"
}
```

Deleting an Other Charge for Purchase Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargePurchaseItem/4
```

Other Charge for Resale Item

An other charge for resale item record exposes an other charge resale item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the other charge for resale item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's otherChargeResaleItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record IDs

The record ID for an other charge for resale item REST record is **otherChargeResaleItem**.

Inaccessible Elements

ItemImages is not accessible in REST web services.

Elements with Different Functionality

The record has a matrix pricing machine. Otherwise, it behaves in the same way as the non-inventory purchase item and non-inventory resale item records.

Code Samples

The following samples show common use cases for other charge resale items. The example ID is 4.

Creating an Other Charge for Resale Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeResaleItem
{ "itemId": " Other Charge Resale Item", "cost": "1", "incomeAccount": { "id": "54", "refName": "Sales" }
}
```

Retrieving an Other Charge for Resale Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeResaleItem/4
```

Updating an Other Charge for Resale Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeResaleItem/4
{ "cost": "10",
}
```

Deleting an Other Charge for Resale Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeResaleItem/4
```

Other Charge for Sale Item

An other charge for sale item record exposes an other charge for sale item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the other charge for sale record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's otherChargeSaleItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for an other charge for sale item REST record is **otherChargeSaleItem**.

Code Samples

The following samples show common use cases for other charge for sale items. The example ID is 4.

Creating an Other Charge for Sale Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeSaleItem
{ "itemId": " Other Charge for Sale", "displayName": "displayName", "upcCode": "code" }
```

Retrieving an Other Charge for Sale Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeSaleItem/4
```

Updating an Other Charge for Sale Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/otherChargeSaleItem/4
{ "itemId": " Other Charge for Sale A", "displayName": "displayName A", "upcCode": "code A"
}
```

Other Name

An other name record exposes a collection of records for people or companies who are not vendors, customers or employees to REST web services.

This record is not a subrecord and has no subrecords.

With the exception of tax-related fields, all elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the other name record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's other name reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the other name REST record is **otherName**.

Limitations

Updating legacy tax fields is not supported.

Code Samples

These samples show common use cases for other name REST methods.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://test.netsuite.com/services/rest/record/v1/metadata-catalog/othername
```

Fetching an Other Name Record

```
GET https://test.netsuite.com/services/rest/record/v1/othername/<id>
```

Creating an Other Name Record

```
POST https://test.netsuite.com/services/rest/record/v1/othername
{ "entityId": "Sophia Company", "subsidiary": { "refName": "Parent Company" }, "companyName": "test company name", "phone": "123
 123 1234", "fax": "123 123 1234", "email": "test-email@netsuite.com", "printOnCheckAs": "123", "emailPreference":
 { "refName": "PDF" }, "phoneticName": "so-FEE-uh", "comments": "Test company comment", "url": "https://test-url.com/", "image":
 { "refName": "testimage.gif" }, "creditLimit": 200, "terms": { "refName": "Net 60" }, "category": { "refName": "A" }, "accountNum
ber": 123, "isInactive": false
}
```

Updating an Other Name Record

```
PATCH https://test.netsuite.com/services/rest/record/v1/othername/<id>
{ "email": "updated-test-email@netsuite.com", "emailPreference": { "refName": "HTML" }, "comments": "Updated test company comment"
}
```

Deleting an Other Name Record

```
DELETE https://test.netsuite.com/services/rest/record/v1/othername/<id>
```

Other Name Category

An other name category record exposes a user-defined mapping to help categorize other name records to REST web services. Other name category records are used to help group other name records by user-created categories.

This record is not a subrecord and has no subrecords.

All elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the other name category record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's other name category reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the other name category REST record is **otherNameCategory**.

Code Samples

These samples show common use cases for other name category REST methods.

Getting OpenAPI Metadata from the Metadata Catalog

```
GET https://test.netsuite.com/services/rest/record/v1/metadata-catalog/othernamecategory
```

Fetching an Other Name Category Record

```
GET https://test.netsuite.com/services/rest/record/v1/othernamecategory/<id>
```

Creating an Other Name Category Record

```
POST https://test.netsuite.com/services/rest/record/v1/othernamecategory
{ "name": "Charities", "isInactive": true
}
```

Updating an Other Name Category Record

```
PATCH https://test.netsuite.com/services/rest/record/v1/othernamecategory/<id>
{ "name": "Nonprofits", "isInactive": false
}
```

Deleting an Other Name Category Record

```
DELETE https://test.netsuite.com/services/rest/record/v1/othernamecategory/<id>
```

Partner

A partner record exposes a partner to REST web services.

For more information about partners and how they function in the user interface, see the help topic Partners.

To access this record in NetSuite, go to Lists > Relationships > Partners.

This record:

- is not a subrecord
- has no subrecords

The following sublists are not accessible through REST web services:

- bulkmerge
- subscriptionmessagehistory
- subscriptions
- taxregistration

The REST API Browser includes information about the field names and field types of the partner record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's partner reference page.

For information about using the REST API Browser, see the help topic The REST API Browser

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record ID

The record ID for the partner REST record is **partner**.

Prerequisites

You must enable the CRM feature before you can use this record through REST web services.

Code Sample

The following samples show common use cases for a partner record. The example ID is 5.

Retrieving Partner Metadata

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/metadata-catalog/partner
```

Retrieving a Partner Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/partner/5
```

Creating a Partner Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/partner
{ "partnerCode": "EXAMPLE COMPANY", "companyName": "Example Company", "subsidiary": "3"
}
```

Updating a Partner Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/partner/5
{ "companyName": "Updated Company Name"
```

```
}
```

Deleting a Partner Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/partner/5
```

Paycheck

A paycheck record exposes a paycheck to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - Account
  - Contact
  - payBonuses
  - payComm
  - payContrib
  - PayDeduct
  - payDisburse
  - payrollitem
  - payEarn
  - payExp
  - payPto
  - paySummary
  - payTax
  - payTime
  - accountingperiod

The REST API Browser includes information about the field names and field types of the paycheck record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's paycheck reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the paycheck REST record is **paycheck**.

Prerequisites

Before you can use this record with REST web services, you must enable the following features:

- REST Web Services

- Token-Based Authentication
- Client SuiteScript
- Server SuiteScript
- Payroll

Inaccessible Elements

Reversed paychecks and paycheck adjustments do not appear in the REST web services paycheck list. However they can be accessed by an administrator by appending the paycheck ID directly at the end of the URL request.

Note: A known issue exists in which an employee with REST permissions can see a paycheck with a future check date in the paycheck list when the **Paychecks Visible to Employees** setting is set to **On Check Date**.

The following subrecords are not exposed in REST:

- payComm
- payExp

Elements with Different Functionality

The following sublists are editable:

- payContrib
- payDeduct
- payEarn

Other sublists are not editable.

The editing of some of this record's subrecords and sublists works differently.

- payExp: When you edit a line item in the UI, the "Apply" checkbox can be unchecked. If the line is unchecked, the line is removed when you save. Through REST web services, this functionality is not available.
- payPto: This sublist is not returned when querying /services/rest/record/v1/paycheck/{id}. To retrieve this record, the user needs to query /services/rest/record/v1/paycheck{id}/payPto.

  payTime: When you edit a line item in the UI, the "Apply" checkbox can be unchecked. If the line is unchecked, the line is removed when you save. Through REST web services, this functionality is not available.

Actions

The paycheck record accessed through REST web services can be edited and deleted. Accessing paycheck records through REST web services will give you information in a different format and include more detailed information about the paycheck entries. Examples include:

- The inclusion internal IDs of the payitems shown on the paycheck record
- The display of residential, editability and exemption status of taxes
- The display of REST reference URLs of subresources returned in the request

- A check date in the following format: “2023-10-23T19:12:00Z”
- Number of total results in the requested record

Elements Related to Taxation Features

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Fields on the paycheck record are related to payrolltaxes and to the ANYPAYROLL feature.

the payTax sublist represents a list of taxes on the paycheck.

Additional Details

Appending "?expandSubResources=True" to the end of an individual paycheck request URL expands the obtained subresources' details.

Code Sample

The following code shows how to access the list of paycheck records:

```
https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/paycheck
```

The following code shows how to access a specific paycheck by ID (123):

```
https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/paycheck/123
```

The following code shows how to access a specific paycheck by ID (123), expanding the subresources returned in the response:

```
https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/paycheck/123?expandSubResources=True
```

The following code shows how to get a paycheck by ID (304):

```
{
   "name": "Example 3.6: Get Paycheck",
   "request": {
      "method": "GET",
      "header": [
         {
            "key": "Content-Type",
            "name": "Content-Type",
            "value": "application/json",
            "type": "text"
         },
         {
            "key": "Accept",
            "value": "application/swagger+json",
            "type": "default",
            "disabled": true
         }
      ],
      "body": {
         "mode": "raw",
```

```
        "raw": ""
      },
      "url": {
        "raw": "{{COMPANY_URL}}/services/rest/record/v1/paycheck/304/?expandSubResources=true",
        "host": [
          "{{COMPANY_URL}}"
        ],
        "path": [
          "services",
          "rest",
          "record",
          "v1",
          "paycheck",
          "304",
          ""
        ],
        "query": [
          {
            "key": "expandSubResources",
            "value": "true"
          }
        ]
      }
    }
}
```

The following code shows how to update a paycheck by ID (606):

```
{
   "name": "Example 3.5: Update paycheck memo",
   "request": {
      "method": "PATCH",
      "header": [
         {
            "key": "Content-Type",
            "name": "Content-Type",
            "value": "application/json",
            "type": "text"
         }
      ],
      "body": {
         "mode": "raw",
         "raw": "{\"memo\": \"new memo\"}"
      },
      "url": {
         "raw": "{{COMPANY_URL}}/services/rest/record/v1/paycheck/606",
         "host": [
            "{{COMPANY_URL}}"
         ],
         "path": [
            "services",
            "rest",
            "record",
            "v1",
            "paycheck",
            "606"
         ]
      }
   }
}
```

The following code shows how to delete a paycheck by ID (705):

```
{
    "name": "Example 2.4: Remove Paycheck",
    "request": {
        "method": "DELETE",
        "header": [],
        "body": {
            "mode": "raw",
```

```
            "raw": ""
        },
        "url": {
            "raw": "{{COMPANY_URL}}/services/rest/record/v1/paycheck/705",
            "host": [
                "{{COMPANY_URL}}"
            ],
            "path": [
                "services",
                "rest",
                "record",
                "v1",
                "paycheck",
                "705"
            ]
        }
    }
}
```

Payment Item

A payment item record exposes a payment item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the payment item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's paymentItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a payment item REST record is **paymentItem**.

Code Samples

The following samples show common use cases for payment items. The example ID is 4.

Creating a Payment Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/paymentItem
{ "itemId": "Payment Item A", "displayName": "Payment Item A DisplayName"
}
```

Retrieving a Payment Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/paymentItem/4
```

Updating a Payment Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/paymentItem/4
{ "itemId": "Payment Item B", "displayName": "Payment Item B DisplayName", "description": "description"
}
```

Deleting a Payment Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/paymentItem/4
```

Payment Method

NetSuite exposes the payment method record to REST web services. A payment method represents an account that transactions write into. You can create payment methods to add selections to the Payment Method field on transactions and in your Web store.

For information about working with the payment method record in the UI, see the help topic Creating a Payment Method.

There are not any prerequisites required for using the payment method record through REST web services.

Note that when you create a new payment method, you have to provide the **methodtype** ID, for example "1" for the Payment Card payment method type. For a list of available payment method type IDs, see the help topic Creating a Payment Method.

The REST API Browser includes information about the field names and field types of the payment method record and about the HTTP methods, request parameters, and operations available to this record.

For details, see the REST API Browser's paymentMethod reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the payment method REST record is **paymentMethod.**

Sublist

The payment method record has a sublist of the **paymentMethodVisual** subrecord, which is not exposed to REST web services.

Code Samples

The code samples in this section show common use cases for payment methods.

Creating a Payment Method

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/paymentMethod
{ "name": "Credit Card PM", "methodtype": {"id": "1"}, /* Id of Payment Card type */ "cardBrands":{ "items":[ {"id": "6"},
 {"id": "3"} ] }, "merchantAccounts":{ "items":[ {"id": "1"} ] }
}
```

Updating a Payment Method

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/paymentMethod/{id} { "name": "Updated Payment Method"
}
```

Phone Call

A phone call record exposes a phone call to REST web services.

This record:

- is not a subrecord
- has one subrecord: EventFile

The REST API Browser includes information about the field names and field types of the phone call record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's phone call reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the phone call REST record is **phonecall**.

Exposed Elements

The following phone call record elements are exposed to REST web services:

- accesslevel
- assigned
- bom (Requires Advanced Bill of Materials feature)
- bomversion (Requires Advanced Bill of Materials feature)
- company
- completed date (a change here can be overwritten by change in status when set to completed in a separate modification, and cannot be set to custom date in the same message that completes the call)
- contact
- create date (Read only)
- customform (Enumerated)
- enddate
- endtime
- externalid
- id
- lastmodifieddate (Cannot be manually changed)
- message
- owner
- phone
- priority
- relateditem
- reminderminutes (Enumerated)
- startdate
- starttime

- status
- supportcase
- timedvent
- title
- transaction
- mfgrouting (Manufacturing Routing feature required)

Code Sample

In the following example, <accountID> represents your Account ID.

HTTP request: `https://<accountID>/services/rest/record/v1/phonecall`

Request Body

```
{ "title": "Call with Tom", "phone": "650-555-1234", "priority": "MEDIUM", "starttime": "16:30", "timedevent": true
}
```

To Obtain Previously-Created Phone Call Record (where ID is 100525)

HTTP Method: GET

URL: `https://<accountID>/services/rest/record/v1/phonecall/100525`

Response Body


```
{ "links": [ { "rel": "self", "http://<accountID>/services/rest/record/v1/phonecall/100525 } ], "accessLevel": false, "assigned":
 { "links": [ { "rel": "self", "href": "http://<accountID>/services/rest/record/v1/customer/-5" } ], "id": "-5", "refName": "A
 Wolfe-admin" }, "createdDate": "2020-11-25T16:54:00Z", "customForm": { "links": [], "id": "-150", "refName": "Standard Phone Call
 Form" }, "endTime": "17:30", "id": "100525", "lastModifiedDate": "2020-11-25T16:54:00Z", "owner": { "links": [ { "rel": "self",
 "href": "http://<acountID>/services/rest/record/v1/customer/-5" } ], "id": "-5", "refName": "-5" }, "phone": "650-555-1234",
 "priority": { "links": [], "id": "MEDIUM", "refName": "Medium" }, "sendEmail": false, "startDate": "2022-11-01", "startTime":
 "16:30", "status": { "links": [], "id": "SCHEDULED", "refName": "Scheduled" }, "timedEvent": true, "timezone": "America/Los_Ange
les", "title": "Call with Tom"
}
```


Price Book

A price book record exposes a SuiteBilling price book to REST web services.

This record:

- is not a subrecord
- has no subrecords

The REST API browser includes information about the field names and field types of the price book record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's price book reference page.

For information about using the REST API browser, see the help topic The REST API Browser.

Record ID

The record ID for a price book REST record is **pricebook**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Code Samples

These samples show common use cases for creating price books with all header fields.

```
{ "subscriptionplan" : {"id" : "{{existing_subscription_plan_id}}"}, "currency" : {"id" : "1"}, "name": "My Price Book {{$timestamp}}", "priceinterval": { "items": [ { "priceplan": {"id": "{{last_priceplan}}"}, "frequency": "MONTHLY", "repeatevery": "1", "startoffsetvalue": 1 } ] }
}
```

```
{ "priceinterval": { "items": [ { "groupid":"438", "linenumber" : "1", "chargetype":"2", "priceplan": {"id": "1017"}, "frequency": "MONTHLY", "repeatevery": "1", "startoffsetvalue": "8", "startoffsetunit": "MONTH", "isrequired": true, "item": { "id": "15" }, "subscriptionplanlinenumber" : "1", "subscriptionplanline":"20" } ] }
}
```

Price Level

A price level record exposes a price level to REST web services.

To access this record in NetSuite, go to Setup > Accounting > Accounting Lists > New > Price Level.

The REST API Browser includes information about the field names and field types of the price level record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's priceLevel reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a price level REST record is **pricelevel**.

Code Samples

The following samples show common use cases for price levels. The example ID is 6.

Creating a Price Level Using a POST Request

```
POST: {{REST_SERVICES}}/record/v1/pricelevel

{
   "name": "Test Price Level",
   "discountPct": -15.0,
   "isInactive": false,
   "isOnline": false,
```

```
    "updateExistingPrices": false
}
```

Retrieving a Price Level Using a GET Request

```
GET: {{REST_SERVICES}}/record/v1/pricelevel/6
```

Updating a Price Level Using a PATCH Request

```
PATCH: {{REST_SERVICES}}/record/v1/pricelevel/6

{
    "discountPct": -20.0,
    "isInactive": false,
    "isOnline": true,
    "name": "Toolbox Price Level",
}
```

Deleting a Price Level Using a DELETE Request

```
DELETE: {{REST_SERVICES}}/record/v1/pricelevel/6
```

Price Plan

A price plan record exposes a SuiteBilling price plan to REST web services.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the price plan record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's price plan reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a price plan REST record is **priceplan**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Code Sample

```
{ "currency": {"id": "1"}, "priceplantype": "2", "pricetiers": { "items": [ { "fromval": 0, "pricingoption": {"id": "-102"}, "val
ue": 50.00 } ] }
```

```
}
```

Pricing Group

A pricing group record exposes a pricing group to REST web services.

To access this record in NetSuite, go to Setup > Accounting > Accounting Lists > New > Pricing Group.

The REST API Browser includes information about the field names and field types of the pricing group record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's pricingGroup reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a pricing group REST record is **pricinggroup**.

Prerequisites

You must enable Multiple Prices before you can use this record through REST web services. To do so, go to Setup > Company > Enable Features. On the **Transactions** subtab under Sales, check the **Multiple Prices** box, and then click **Save**.

Actions

The pricing group record has the following actions through REST web services:

- Create
- Read
- Update
- Delete

Code Samples

The following samples show common use cases for pricing groups. The example ID is 6.

Creating a Pricing Group Using a POST Request

```
POST {{REST_SERVICES}}/services/rest/record/v1/pricinggroup
{ "name": "Test Pricing Group", "externalid": "pricinggroup1", "isInactive": false
}
```

Retrieving a Pricing Group Using a GET Request

```
GET {{REST_SERVICES}}/services/rest/record/v1/pricinggroup/6
```

Updating a Pricing Group Using a PATCH Request

```
PATCH {{REST_SERVICES}}/services/rest/record/v1/pricinggroup/6
{ "name": "Updated Pricing Group", "externalid": "pricinggroup2", "isInactive": true
}
```

Deleting a Pricing Group Using a DELETE Request

```
DELETE {{REST_SERVICES}}/services/rest/record/v1/pricinggroup/6
```

Project Task

A project task record exposes a project task to REST web services. You can use the project task record to keep track of specific activities and milestones associated with a project.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the project task record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's project task reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a project task REST record is **projecttask**.

Prerequisites

You must enable the Project management feature before you can use this record through REST web services.

Limitations

Tax-related fields are not supported in REST.

The following fields are not supported in REST:

- isMilestone
- isOnCriticalPath
- isSummaryTask
- lateEnd
- lateStart
- slackMinutes

- bBudgetShowCalculatedLines
- cBudgetShowCalculatedLines

The following sublists are not supported in REST:

- ProjectTaskAssignee
- ProjectTaskBillingBudget
- ProjectTaskCostBudget

Code Sample

This sample shows how to update the record:

```
POST /services/rest/record/v1/projecttask
{ "project": "328", "title": "startDateTest", "constraintType": "ASAP", "startDate": "2013-11-15"
}
```

Promotion Code

Promotions enable you to track the source of revenue and to offer discounts in the form of coupons. Each promotion has a promotion code that can be applied to transactions and campaigns.

This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the promotion code record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's promotionCode reference page.

For information about promotion, see the help topic Promotion.

For information about working with promotions in the UI, see the help topic The Promotion Record.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the promotion code REST record is **promotioncode**.

Prerequisites

- You must enable the SuitePromotions feature before you can use this record through REST web services.
- If you need to create shipping promotions, you need to enable Charge for Shipping.
- If you need to use the auto-apply configuration, you need to enable the Auto-apply feature.
- Other subtabs or sublists may require additional features to be enabled.

Elements with Different Functionality

The following sublist has different functionality through REST web services:

- campaigns

The following fields related to CSV import files under the Coupon Codes subtab are not accessible through REST web services:

- lastUploadDate
- lastUploadStatus
- lastUploadFile
- couponCodeFile

The following field is not accessible through REST web services:

- useMaterializedSavedSearch

> **Note:** The useMaterializedSavedSearch field can be found under the Saved search preferences subtab and it applies to promotions with items saved search.

Elements Related to Taxation Features

> **Note:** REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The discount item for accounting field on the promotioncode REST record is related to taxation.

Additional Details

Considerations related to features:

- For sales channels, you need both the locations and website features enabled even if you do not have the sites configured.

Considerations related to forms:

- Only forms provided by NetSuite for SuitePromotions are supported.

Considerations related to fields:

- Do not use fields and forms that do not apply to SuitePromotions or are not supported. For example, forms and fields specific to advanced promotions.
- The lastUploadDate, lastUploadStatus, lastUploadFile, and couponCodeFile fields are related to the coupon code file uploads and are not available.
- The campaigns field is not available through REST.
- When you set a dynamic customer group as audience, you must explicitly set a value to the customerGroupPreference field.
- Before updating sublists, refer the documentation for working with sublists. For example, working with items, discountItems, and so on.
- You should use the supported SuitePromotion form IDs for customForm.
- Do not use customForm to try to change the promotion type. Instead, delete the existing one and create a new one.

Note: The promotion type we create is based on the value of customForm field. You can create an Item promotion by setting the customForm field value to -10501. Similarly, we can use -10502 to create an Order promotion, -10503 to create a Shipping promotion, -10504 to create a Fixed price promotion, and -10507 to create a Free gift promotion.

Considerations when testing your requests:

- Open the promotion in the user interface and verify that all the values are the same as if you were to create the promotion manually.
- Verify the promotion and its sublist and tab contents in the user interface when updating it.
- Use orders that meet the conditions of the promotion and verify that the promotion is applied as expected.

Other considerations:

- In case of both single-use and multi-use coupon code types, delete and create a new promotion if you want to change the coupon code type. Updating the coupon code type from single-use to multi-use or from multi-use to single-use is not supported.
- You cannot remove a promotion that is being used in a transaction.

Code Sample

In the following example, we create an order promotion scheduled for year 2024 applying a 10% discount for orders with a minimum amount of 150.00 and 10 automatically generated coupon codes. Also, in the request we use 10 as the discount item ID for accounting.

The following code shows how to create a promotion:

```
POST: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/promotionCode
{ "customForm": { "id": "-10502" }, "name": "My order promotion", "startDate": "2024-01-01", "endDate": "2024-12-31", "combination
Type": "ORDERTYPEEXCLUSIVE", "discount": { "id": "10" }, "whatTheCustomerNeedsToBuy": { "id": "MINIMUMORDERAMOUNTORSPECIFICITEMS"
 }, "minimumOrderAmountCheck": true, "minimumOrderAmount": 150.0, "discountType": { "id": "percent" }, "rate": 10.0, "useType":
 { "id": "SINGLEUSE" }, "codePattern": "ABC-[AN.3]-[N.3]", "numberToGenerate": 10
}
```

In the following example, 206 represents the promotion ID.

The following code shows how to retrieve a promotions record using a GET Request:

```
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/promotioncode/206
```

The following code shows how you can add an optional parameter to the GET request so that the subrecords and sublists are expanded and not returned only as links:

```
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/promotioncode/206?expandSubResources=true
```

The following code shows how to update a promotions record by removing the minimum order amount condition and setting the condition for specific items with IDs 150 and 200 and a quantity of 10:

```
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/promotionCode/206
{ "minimumOrderAmountCheck": false, "minimumOrderAmount": null, "specificItemsCheck": true, "items": { "items": [ {"item":
 {"id": 150}}, {"item": {"id": 200}} ] }, "itemQuantifier": 10
}
```

The following code shows how to delete a promotions record:

```
DELETE: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/promotioncode/206
```

Note: If the promotion is already being used in a transaction, you won't be able to remove the promotions record.

Purchase Contract

This record enables you to take advantage of contracted quantity-based terms and discounts when creating purchase orders.

This record is available only when the Purchase Contracts feature is enabled at Setup > Company > Setup Tasks > Enable Features (Administrator), on the Transactions subtab.

In the UI, you access this record at Vendor Dashboard > Transactions > Enter Purchase Contracts (Administrator).

For help working with this record in the UI, see the help topic Creating Purchase Contracts.

The item element on this record is not accessible through REST web services.

The purchase contract record type includes the following actions from the Actions Framework:

- New
- Make Copy
- Make Copy without Pricing Tiers
- Email

The REST API Browser includes information about the field names and field types of the purchase contract record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's purchaseContract reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a purchase contract REST record is **purchaseContract**.

Prerequisites

This record is available only when the Purchase Contract feature is enabled at Setup > Company > Setup Tasks > Enable Features (Administrator), on the Transactions subtab.

Code Samples

The following examples show how to read, create, update, or delete a purchase contract:

Read:

```
GET {{REST_SERVICES}}/record/v1/purchaseContract/${transaction-id}
```

Create:

```
POST {{REST_SERVICES}}/record/v1/purchaseContract { "entity": { "id": {entity-id} }, "effectivitybasedon": { "id" : "ORDERDATE"
 }, "item": { "items": [ { "item": { "id": "${item-id}" }, "itemPricing": { "baserate": 10.0, } }
}
```

Update:

```
PATCH {{REST_SERVICES}}/record/v1/purchaseContract/${transaction-id}

{
"memo" : "Memo text"
}
```

Delete:

```
DELETE {{REST_SERVICES}}/record/v1/purchaseContract/${transaction-id}
```

Purchase Order

NetSuite exposes the purchase order record to REST web services. This record has the following subrecords:

- billingAddress
- shippingAddress

The following elements are not accessible through REST web services:

- accounting books
- items
- tax detail

If the Allow Expenses on Purchases accounting preference is set, the expenses sublist is also not accessible.

All other elements on this record are accessible through REST web services.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the purchase order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's purchase order reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a purchase order REST record is **purchaseOrder**.

Prerequisites

You must enable the Purchase Orders feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for a purchase order record.

Creating a Purchase Order

```
POST https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/purchaseOrder
Body:
{ "entity": { "id": {{vendorId}} }, "item": { "items": [{ "item": { "id": {{ItemId}} }, "rate": 10 }] }
}
```

Reading a Purchase Order

```
GET https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/purchaseOrder/{{id}}?expandSubResources=true
```

Updating a Purchase Order

```
PATCH https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/purchaseOrder/{{id}}
Body:
{ "memo": "This is a test"
}
```

Deleting a Purchase Order

```
DELETE https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/purchaseOrder/{{id}}
```

Return Authorization

A return authorization record exposes a return authorization to REST web services. This record is not a subrecord. This record has one subrecord: record ID item.

The REST API Browser includes information about the field names and field types of the return authorization record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's returnAuthorization reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a return authorization REST record is **returnAuthorization**.

Exceptions

These fields are not exposed:

- Taxable

- Tax
- Tax %

Actions

The return authorization record has the following actions through REST web services:

- New
- Email
- Show Activity
- Go To Register
- GL Impact

Label Mapping

The REST label for customer is **entity**. The corresponding UI form label is **CUSTOMER**.

Taxation

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The return authorization record includes the following fields related to taxation features:

- Taxable
- Tax
- Tax %
- Customer Tax Reg. Number
- Subsidiary Tax Reg. Number
- Tax Details Override
- Tax Point Date
- Tax Point Date Override
- Tax Registration Override
- Tax

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup >

Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

Code Samples

These samples show common use cases for return authorizations. The example ID is 132.

Creating a Return Authorization Using a POST Request

```
POST https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/returnauthorization
{ "altSalesTotal": 0.0, "canHaveStackable": false, "currency": { "id": 3 }, "deferredRevenue": 0.0, "discountTotal": 1.0, "entity":
 { "id": 164 }, "exchangeRate": 1.0, "excludeCommission": false, "item": { "items": [ { "item": { "id": 5 }, "line": 1, "quanti
ty": 2.0, "rate": 12.78 } ] }, "netAltSalesTotal": 0.0, "salesEffectiveDate": "2023-05-30", "salesTeam": { "items": [ { "contribu
tion": 100.00, "employee": { "id": "40" }, "isPrimary": false, "salesRole": { "id": "-2" } }, { "contribution": 100.00, "employee":
 { "id": "134" }, "isPrimary": true, "salesRole": { "id": "3" } } ] }, "shipIsResidential": false, "shipOverride": false, "subto
tal": 39.95, "toBeEmailed": false, "toBeFaxed": false, "toBePrinted": false, "total": 39.95, "tranDate": "2023-07-13", "tranIsVsoe
Bundle": false, "vsoeAutoCalc": false, "weekendpreference": "ASIS"
}
```

Retrieving a Return Authorization Using a GET Request

```
GET https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/returnauthorization/132
```

Updating a Return Authorization Using a PATCH Request

```
PATCH https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/returnauthorization/132{ "altSalesTotal": 0.0, "can
HaveStackable": false, "currency": { "id": 3 }, "deferredRevenue": 0.0, "discountTotal": 1.0, "entity": { "id": 164 }, "exchang
eRate": 1.0, "excludeCommission": false, "item": { "items": [ { "item": { "id": 5 }, "line": 1, "quantity": 2.0, "rate": 12.78
 } ] }, "netAltSalesTotal": 0.0, "salesEffectiveDate": "2023-05-30", "salesTeam": { "items": [ { "contribution": 100.00, "em
ployee": { "id": "40" }, "isPrimary": false, "salesRole": { "id": "-2" } }, { "contribution": 100.00, "employee": { "id": "134"
 }, "isPrimary": true, "salesRole": { "id": "3" } } ] }, "shipIsResidential": false, "shipOverride": false, "subtotal": 39.95, "to
BeEmailed": false, "toBeFaxed": false, "toBePrinted": false, "total": 39.95, "tranDate": "2023-07-13", "tranIsVsoeBundle":
 false, "vsoeAutoCalc": false, "weekendpreference": "ASIS"
}
```

Deleting a Return Authorization Using a DELETE Request

```
DELETE https://0000071.suitetalk.api.snap.netsuite.com/services/rest/record/v1/returnauthorization/132
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Requisition

Requisition records are used to initiate the purchase process for goods and services needed within your company. For help working with this record in the UI, see the help topic Entering a Requisition.

NetSuite exposes the purchase requisition record to REST web services. This record is not a subrecord and does not contain a subrecord.

The following elements are not accessible through REST web services:

- Item
- Expense sublist
- Accounting book sublist

The REST API Browser includes information about the field names and field types of the requisition record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's requisition reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a purchase requisition REST record is **purchaserequisition**.

Prerequisites

To use the purchase requisition REST record, you must have the Requisition feature enabled.

Code Samples

The following code sample shows how to create a purchase requisition:

```
POST {{REST_SERVICES}}/record/v1/purchaseRequisition Body:
{ "entity": { "id": {{entityId}} }, "item": { "items": [{ "item": { "id": {{item}} }, "rate": 10 }] }
}
```

Revenue Recognition Schedule

A revenue recognition schedule indicates the posting periods in which revenue should be recognized, and the amount to be recognized in each period, for an item sale. A revenue recognition schedule is generated for any sales transaction item that has an associated revenue recognition template. The point at which a revenue recognition schedule is generated for an item sale depends upon the type of sales transaction and enabled features and preferences set in your account. The schedule could be generated

when a transaction is first saved, when it is approved, or when it is billed. Revenue recognition schedules provide a basis for the generation of journal entries that record the impact of item sales. This record is available when the Revenue Recognition feature is enabled.

For help working with this record in the UI, see the help topic Working with Revenue Recognition Schedules.

Revenue recognition schedules are system-generated.

The REST API Browser includes information about the field names and field types of the revenue recognition schedule record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's revRecSchedule reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a revenue recognition schedule REST record is **RevRecSchedule**.

Prerequisites

You must enable the following features or preferences to access this record:

- Revenue Recognition
- Set Up Auto-Generated Numbers
- Allow Users to Modify Revenue Recognition
- Sales Order Revenue Forecasting

For more information about using record, see the help topic Working with Revenue Recognition Schedules.

Limitations

All fields of the revenue recognition schedule record are read-only except for the name and recurrence fields. To modify these two fields, you must enable the Allow Users to Modify Revenue Recognition Schedule preference.

Additional Information

If you set **isAmortization** to true, the schedule changes to amortization. If you set **isSchedule** to true, the schedule changes to revenue recognition. The **isrecognized** field appears when the **amortizationtype** is Variable. **forecastTemplate** appears when Sales Order Revenue Forecasting is enabled.

Code Samples

The following example shows how to read a revenue recognition schedule record:

Read:

```
GET {{REST_SERVICES}}/record/v1/RevRecSchedule/805?expandSubResources=true
```

Revenue Recognition Template

A revenue recognition template indicates how revenue from associated items should be posted. For each template: you can select from a choice of standard terms or define your own custom terms, set the time period over which recognition occurs, define an offset to delay the start of recognition, and set up an initial amount to be recognized. This record is available when the Revenue Recognition feature is enabled.

For help working with this record in the UI, see the help topic Creating Revenue Recognition Templates.

The REST API Browser includes information about the field names and field types of the revenue recognition template record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's revRecTemplate reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a revenue recognition template REST record is **revrectemplate**.

Prerequisites

You must enable the following features or preferences to access this record:

- Revenue Recognition

For more information about using record, see the help topic Creating Revenue Recognition Templates.

Limitations

All fields of the revenue recognition template record are read-only except for the name and recurrence fields. To modify these two fields, you must enable the Allow Users to Modify Revenue Recognition Schedule preference.

The **isAmortization**, **isPublic**, and **isSchedule** fields on this record are not available in REST.

Code Samples

The following example shows how to read a revenue recognition template record:

Read:

```
GET {{REST_SERVICES}}/record/v1/revrectemplate/104?expandSubResources=true
```

Sales Order

A sales order record exposes a sales order to REST web services. This record has one subrecord: salesOrderItem.

To access this record in NetSuite, go to Transactions > Sales > Enter Sales Order.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the sales order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's sales order reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a sales order REST record is **salesorder**.

Usage Notes

To use the **Choose Team** and **Update Customer** fields in the Sales Team subtab, the Team Selling feature must be enabled. To enable Team Selling, go to Setup > Company > Enable Features > CRM. Check the Team Selling box, and then click Save.

The Multi-Partner Management Feature must also be enabled to use the **Update Customer** field in the Relationships subtab. An administrator can enable the Multi-Partner Management feature at Setup > Company > Setup Tasks > Enable Features. Click the CRM subtab, and then under Partners, check the Multi-Partner Management box. Click Save.

When you transform a sales order into an item fulfillment, you must include the `inventoryLocation` field in the request body. For more information, see the help topic Transforming Sales Orders with the Intercompany Cross-Subsidiary Feature

```
POST /record/v1/salesOrder/<source Sales Order id>/!transform/itemFulfillment
{
        "inventoryLocation": {
                "id" : <location id>
        }
...
}
```

You can create a sales order with an ItemGroup item but it is not possible to change or update the quantity, rate, and amount value of the member items using the create request. You can only update the quantity, rate, and amount values using the PATCH request.

Code Samples

These samples show common use cases for sales orders. The example ID is 1504.

A Basic GET Request of a Sales Order

```
GET: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/salesorder/1504
```

Updating a Sales Order

```
PATCH: https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/salesorder/1504
```

```
{"istaxable":false}
```

Updating the Amount of a Line Item Within a Sales Order

```
{
"item": {
"items": [{
"line": 1,
"item": {
"id": 328
},
"amount": 54.0
}]
}
}
```

Setting the Sales Order's Sales Team Members to Match Sales Group 164

```
"salesGroup": {
    "id": 164 }
```

Update the Customer's Partner Team to Match the Transaction Partners

```
"syncPartnerTeams": true
```

Update the Customer's Sales Team to Match the Transaction Sales Team

```
"syncSalesTeams": true
```

Sales Role

A sales role record exposes a sales role to REST web services.

For more information about sales roles and how they function in the user interface, see the help topic NetSuite Roles Overview.

To access this record in NetSuite, go to Setup > Users/Roles > Manage Roles.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the sales role record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's sales role reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the sales role REST record is **salesrole**.

Prerequisites

You must enable the Team Selling feature before you can use this record through REST web services. For more information, see the help topic Team Selling.

Code Sample

The following samples show common use cases for a sales role record. The example ID is 6.

Retrieving Sales Role Metadata

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/metadata-catalog/salesrole
```

Deleting a Sales Role Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/salesrole/6
```

Sales Tax Item

For help working with this record in the UI, see the help topic Sales Taxes.

This record is not a subrecord.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the sales tax item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's salesTaxItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the sales tax item REST record is **salesTaxItem**.

Prerequisites

The SuiteTax feature must be enabled.

Actions

The sales tax item record has the following actions through REST web services:

- Submit new
- Submit

Code Samples

This code sample shows how to post the sales tax item record:

```
POST /services/rest/record/v1/salesTaxItem { "name": "Test Tax Code", "description": "Test Tax Code", "taxType": "13"
}
```

This code sample shows how to get the sales tax item record:

```
GET /services/rest/record/v1/salesTaxItem/91 { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-deuf1-001.en
g.netsuite.com/services/rest/record/v1/salesTaxItem/91" } ], "description": "Test Tax Code", "id": "91", "isInactive": false, "item
Type": "TaxItem", "name": "Test Tax Code", "taxType": { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-
deuf1-001.eng.netsuite.com/services/rest/record/v1/taxtype/13" } ], "id": "13", "refName": "US Tax Type" }
}
```

Service Purchase Item

A service purchase item record exposes a service purchase item report to REST web services. Service purchase item records are used to track charges for a service your business pays for that is not billed to a customer.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the service purchase item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's service purchase item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a service purchase item REST record is **servicepurchaseitem**.

Limitations

Tax schedule-related fields are not supported in REST.

Code Sample

This sample shows how to create the record:

```
POST /services/rest/record/v1/servicepurchaseitem
{
   "itemId": "YourItemName"
}
```

Service Resale Item

A service resale item record exposes a service resale item report to REST web services. Service resale item records are used to track charges for a service your business pays for initially and then bills to a customer.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the service resale item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's service resale item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a service resale item REST record is **serviceresaleitem**.

Limitations

Tax schedule-related fields, billing rate-related and item task template-related sublists are not supported in REST.

Code Sample

This sample shows how to create the record:

```
POST /services/rest/record/v1/serviceresaleitem
{
   "itemId": "YourItemName"
}
```

Service Sale Item

A service sale item record exposes a service sale item report to REST web services. Service sale item records are used to track charges for a service your business performs that is billed to a customer.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the service sale item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's service sale item reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a service sale item REST record is **servicesaleitem**.

Limitations

Tax schedule-related fields, billing rate-related and item task template-related sublists are not supported in REST.

Code Sample

This sample shows how to create the record:

```
POST /services/rest/record/v1/servicesaleitem
{
   "itemId": "YourItemName"
}
```

Ship Item

A ship item record exposes a ship item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the ship item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's shipItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a ship item REST record is **shipitem**.

Taxation

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The ship item record includes the following fields related to taxation features:

- Charge Tax on This Shipping Portion of Item
- Charge Tax on This Handling Portion of Item

Code Samples

The following samples show common use cases for ship items. The example ID is 4.

Creating a Ship Item Using a POST Request

This sample shows how to create a $10.00 flat-rate ship item.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/shipitem
{ "itemid": "Flat Rate Priority Shipping", "displayname": "Flat Rate Priority Shipping", "description": "Only to be used for priority orders - flat rate 10.00", "subsidiary": "1", "isonline": false, "account": "54", "costbasis": "fr", "shippingflatrateamount": 10.00, "accounthandling": "238", "taxschedule": "1", "taxschedulehandling": "1"
}
```

Retrieving a Ship Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/shipitem/4
```

Updating a Ship Item Using a PATCH Request

This sample shows how to update fields on a ship item.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/shipitem/4
{ "itemid": "Flat Rate Priority Shipping - $15", "displayname": "Flat Rate Priority Shipping - $15", "description": "Only to be used for priority orders - flat rate 15.00", "shippingflatrateamount": 15.00
}
```

Deleting a Ship Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/shipitem/4
```

Subscription

A subscription record exposes a SuiteBilling subscription to REST web services. This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the subscription record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subscription reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a subscription REST record is **subscription**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Limitations

A subscription REST record does not show change orders, related records, or system information.

Code Samples

These samples show common use cases for creating subscriptions.

Create with all Header Fields

```
{ "name": "REST-Subscription", "customer": 4, "billingAccount": 1, "subscriptionPlan": 8, "priceBook": 1, "initialTerm": {"id" : -102}, "startDate": "2019-2-20", "billingSchedule": 1, "billingFrequency": "MONTHLY", "defaultRenewalTerm": {"id" : 2}, "autoRenewal": true, "advanceRenewalPeriodNumber": 5, "advanceRenewalPeriodUnit": "DAYS", "defaultRenewalMethod": "CREATE_NEW_SUBSCRIPTION", "defaultRenewalPriceBook": {"id" : 103}, "defaultRenewalPlan": {"id" : 8}, "defaultRenewalTranType": "SalesOrd", "endDate": "2022-3-5"
}
```

Create with Different Pricing Values

```
{ "name": "REST-Subscription_with_pricing", "customer": 4, "billingAccount": 1, "subscriptionPlan": 8, "priceBook": 1, "initialTerm": {"id" : 1}, "startDate": "2019-2-20", "billingFrequency": "MONTHLY", "priceinterval": { "items": [{ "subscriptionplanlinenumber": 2, "quantity": 3, "repeatEvery": 2, "pricePlan": {"id": 820}, "frequency": "ANNUALLY" }] }
}
```

Create a New Price Interval

```
{ "name": "REST-Subscription_with_pricing", "customer": 4, "billingAccount": 1, "subscriptionPlan": 8, "priceBook": 1, "initialTerm": {"id" : 1}, "startDate": "2019-2-20", "billingFrequency": "MONTHLY", "priceinterval": { "items": [{ "subscriptionplanlinenumber": 2, "linenumber": 2, "quantity": 2, "startDate": "2019-4-20", "pricePlan": {"id": 1020}, "frequency": "MONTHLY", "repeatEvery": 1 }] }
}
```

Create with Different Subline Values

```
{ "name": "REST-Subscription_with_pricing", "customer": 4, "billingAccount": 1, "subscriptionPlan": 8, "priceBook": 1, "initialTerm": {"id" : 1}, "startDate": "2019-2-20", "billingFrequency": "MONTHLY", "subscriptionline": { "items": [{ "linenumber": 1, "includeinrenewal": true, "revrecoption": "OVER_TERM" }, { "linenumber": 2, "proratestartdate": false, "prorateenddate": false, "includeinrenewal": false }] }
}
```

Create with an Add-On Item

```
{ "name": "REST-Subscription_with_pricing", "customer": 4, "billingAccount": 1, "subscriptionPlan": 8, "priceBook": 1, "initialTerm": {"id" : 1}, "startDate": "2019-2-20", "billingFrequency": "MONTHLY", "subscriptionline": { "items": [{ "linenumber": 4, "in
```

```
cludeinrenewal": true, "item": 5, "subscriptionlinetype": 2, "billingMode": "IN_ADVANCE", "proratestartdate": false, "prorateend
date": true, "isincluded": false }] }, "priceinterval": { "items": [{ "subscriptionplanlinenumber": 4, "linenumber": 4, "quanti
ty": 3, "repeatEvery": 2, "pricePlan": {"id": 1217}, "frequency": "ANNUALLY" }] }
}
```

Subscription Line

A subscription line record exposes a SuiteBilling subscription line to REST web services.

This record is not a subrecord, but is always tied to a subscription.

The REST API Browser includes information about the field names and field types of the subscription line record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subscription line reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a subscription line REST record is **subscriptionline**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Limitations

A subscription line REST record does not show system notes.

You cannot directly create or delete the exposed subscription line record, but you can update some of the fields.

Code Sample

```
Sample Update:
{ "subscriptionlinestatus": "PENDING_ACTIVATION", "proratestartdate": false, "prorateenddate": false, "includeinrenewal": false
}
```

Subscription Plan

A subscription plan is a stand-alone record that exposes a SuiteBilling subscription plan to REST web services.

A subscription requires a subscription plan, but the subscription plan is not always tied to a subscription.

The REST API Browser includes information about the field names and field types of the subscription plan record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subscription plan reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a subscription plan REST record is **subscriptionplan**.

The record ID for a subscription plan REST subrecord is **subscriptionplanmember**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Limitations

A subscription plan REST record does not show price books or related records.

When you access this record through REST web services, there is no link available for creating price books.

Code Sample

This sample shows a common use case for creating a subscription plan.

Create with all Header Fields

```
{ "itemid": "NotRested-2", "initialterm" : {"id" : 1}, "member": { "items": [{ "item": {"id": 5 }, "isrequired": true, "subscrip
tionlinetype": "2", "renewaloption": "DIFFERENT_PLAN" }] }
}
```

Subscription Term

A subscription term is a stand-alone record that exposes a SuiteBilling subscription term to REST web services.

The REST API Browser includes information about the field names and field types of the subscription term record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subscription term reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a subscription term REST record is **subscriptionterm**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

Code Sample

This sample shows a common use case for creating a subscription plan.

Create a Six-Month Subscription Term

```
POST services/rest/record/v1/subscriptionterm
{ "name": "Six Months", "subscriptionTermDuration": 6, "subscriptionTermUnit": { "id": "MONTHS" }
}
```

Statistical Journal Entry

The statistical journal entry record lets you increase or reduce the balance of statistical accounts.

To use this record, the Statistical Accounts feature must be enabled at Setup > Company > Setup Tasks > Enable Features (Administrator), on the Accounting subtab. Also, you must have already created at least one statistical account. In the UI, you access this record at Transactions > Financial > Make Journal Entries > List (Administrator).

For help working with this record in the UI, see the help topic Working with Statistical Journal Entries.

The REST API Browser includes information about the field names and field types of the statistical journal entry record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's statisticalJournalEntry reference page.

For information about using the REST API Browser, see the help topic The REST API Browser

Record ID

The record ID for the statistical journal entry REST record is **statisticaljournalentry**.

The statistical journal entry record does not include any subrecords.

Limitations

Read-only sublists, including system notes, are not supported in REST.

Code Sample

```
{ "approved": true, "createdDate": "2023-08-03T06:27:00Z", "customForm": { "id": "30" }, "lastModified
Date": "2023-08-03T06:27:00Z", "line": { "items": [ { "account": { "id": "122" }, "debit": 10 } ] }, "memo": "memo test", "posting
Period": { "id": "267" }, "reversalDefer": false, "subsidiary": { "id": "4" }, "tranDate": "2023-08-02", "unitsType": { "id": 1
 }, "tranId": "1", "unit": 1
}
```

Subsidiary

NetSuite exposes the subsidiary record to REST web services. The subsidiary record enables you to manage data for a hierarchical structure of separate legal entities. See the help topic Subsidiaries in OneWorld.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the subsidiary record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subsidiary reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the subsidiary REST record is **subsidiary**.

The record IDs for the subsidiary REST subrecords are:

- **mainaddress**
- **shippingaddress**
- **returnaddress**

Prerequisites

You must either use NetSuite OneWorld or have the Subsidiaries hidden feature enabled before you can use this record through REST web services.

Exposed Elements

The following subsidiary record elements are exposed to REST web services:

- (internalid) - internal id
- (externalid) - external id
- (isinactive) - subsidiary is inactive
- (legalname) - legal name
- (name) - name
- (fullname) - (search-only field)
- (parent) - subsubsidiary of
- (currency) - currency
- (iselimination) - elimination
- (country) - country
- (state) - state/province
- (url) - web site
- (email) - return email address
- (fax) - fax

The following subsidiary subrecord elements are exposed to REST web services:

- (mainaddress) - main address

- (shippingaddress) - shipping address
- (returnaddress) - return address

Limitations

Only GET (Read, Search) operation is supported for REST web services. POST (create), PATCH (update), and DELETE are not supported. Tax-related fields are not supported in REST.

Code Samples

The following code samples show a common use case for getting supported elements from the subsidiary REST record.

```
{GET /services/rest/record/v1/subsidiary/<id>}
```

```
{GET /services/rest/record/v1/subsidiary?q=fullName CONTAIN "parent"}
```

Subtotal Item

A subtotal item record exposes a subtotal item to REST web services. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the subtotal item record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's subtotalItem reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a subtotal item REST record is **subtotalItem**.

Code Samples

The following samples show common use cases for subtotal items. The example ID is 4.

Creating a Subtotal Item Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/subtotalItem
{ "itemId": " Subtotal Item A", "displayName": "Subtotal Item A DisplayName"
}
```

Retrieving a Subtotal Item Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/subtotalItem/4
```

Updating a Subtotal Item Using a PATCH Request

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/subtotalItem/4
{ "itemId": "Payment Item B", "displayName": "Payment Item B DisplayName", "description": "description"
}
```

Deleting a Subtotal Item Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/subtotalItem/4
```

Support Case

Support cases are support issues logged for specific companies and used for case management.

Use the support case record to create new support cases and expose them to REST web services.

To learn more, see the help topic Support Management.

This record:

- is not a subrecord
- has one subrecord: supportcaseescalatehist

The REST API Browser includes information about the field names and field types of the support case record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's supportCase reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a support case REST record is **supportcase**.

Prerequisites

You must enable the Lists -> Case permission before you can use this record through REST web services.

Elements Not Accessible Through REST Web Services

The following fields are present in the UI but not accessible using REST web services:

- initialresponsetime
- timeelapsed
- timeopen
- timetoassign

The following fields are not accessible through REST web services:

- autoname
- customform

- emailform
- emailemployees
- externalid
- escalationmessage
- firstissueremoved
- helpdesk
- htmlmessage
- incomingmessage
- internalonly
- mediaitem
- messagenew
- messagesave
- origbinactive
- outgoingmessage
- transactionid
- timeitem
- timeonhold
- timetoclose
- solutions
- serialnumber
- useemployeetemplate

Additional Details

The support case record is complex because it is integrated with many features.

Some elements of the support case form behave differently when accessed through REST web services.

Code Sample

The following samples show common use cases for support cases.

Creating a Support Case

```
POST https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/supportCase
With body: { "title": "some subject", "company": { "id": 931 }
}
```

Note: To create a support case, you need to have the permission of lists > customers with at least a value of view.

Reading a Support Case

```
GET https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/supportCase/{{id}}
```

Updating a Support Case

```
PATCH https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/supportCase/{{id}}
Body:
{ "title": "different subject"
}
```

Deleting a Support Case

```
DELETE https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/supportCase/{{id}}
```

Task

A task record exposes a task to REST web services.

This record:

- is not a subrecord
- has the following subrecords
  - EventFile
  - TaskCompanyMap
  - TaskContactMap

The REST API Browser includes information about the field names and field types of the task record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's task reference page.

For more information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the task REST record is **task**.

Exposed Elements

The following task record elements are exposed to REST web services:

- accesslevel
- assigned
- bom (Requires Advanced Bill of Materials feature)
- bomrevision (Requires Advanced Bill of Materials feature)
- company
- contact
- createdate (Read only)
- customform
- duedate
- endtime

- externalid
- id
- nessage
- order
- owner
- percentcomplete
- priority
- relateditem
- reminderminutes
- sendemail
- startdate
- starttime
- status
- supportcase
- timedevent
- title
- transaction
- usernotes

Code Sample

In the following example, <accountID> represents your account ID.

HTTP request: https://<accountID>/services/rest/record/v1/task

Request Body

```
{ "title": "Prepare sales slides", "message": "Charts are required", "priority": "HIGH", "duedate": "2022-11-03", "timedevent": true
}
```

To Obtain Previously-Created Task Record (where ID is 100526)

HTTP Method: GET

URL: https://<accountID>/services/rest/record/v1/task

Response Body

```
{ "links": [ { "rel": "self", "href": "http://<accountID>/services/rest/record/v1/task/100526 } ], "accessLevel": false, "assigned": { "links": [ { "rel": "self", "href": "http://<account>/service/rest/record/v1/customer/-5" } ], "id": "-5", "refName": "A Wolfe-admin" }, "createdDate": "2020-11-25T17:13:00Z", "links": [], "id": "-120", "refName": "Standard Task Form" }, "dueDate": "2022-11-03", "endTime": "10:00", "id": "100526", "lastModifiedDate": "2020-11-25T17:13:00Z", "message": "Charts are required", "owner": { "links": [ { "rel": "self", "href": "http://<accountID>/services/rest/record/v1/customer/-5" } ], "id": "-5", "refName": "-5" }, "percentTimeComplete": 0.0, "priority": { "links": [], "id": "HIGH", "refName": "High" }, "sendEmail": false, "startDate": "2020-11-25", "startTime": "09:00", "status": { "links": [], "id": "NOTSTART", "refName": "Not Started" }, "timedEvent": true, "timeRemaining": "0:00", "timeRemaining": "0:00", "timezone": "America/Los_Angeles",
"title": "Prepare sales slides"
```

```
}
```

Tax Type

A tax type is associated with a nexus, and tells the system where to post the tax amounts on the balance sheet. For each nexus, you must assign at least one account in your general ledger for posting tax.

For more information about tax types in SuiteTax, see the help topic Understanding Tax Types and Tax Codes in SuiteTax.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the tax type record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's taxType reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a taxtype REST record is **taxType**.

Subrecord

This record has the **taxTypeNexusAccounts** subrecord.

Prerequisites

The SuiteTax features must be enabled.

Actions

The tax type record has the following actions through REST web services:

- Submit new
- Submit
- Reset
- Delete

Code Samples

This code sample shows how to post the tax type record:

```
POST /services/rest/record/v1/taxType { "country": "US", "name": "US Tax Type", "description": "US Tax Type", "doesNotAddToTotal":
 true, "nexusAccounts": { "items": [ { "nexus": { "id": "1" }, "payablesAccount": { "id": "24" }, "receivablesAccount": { "id": "6"
 } } ] }
}
```

This code sample shows how to get the tax type records:

```
GET /services/rest/record/v1/taxType/13
{ "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-deuf1-001.eng.netsuite.com/services/rest/record/v1/taxType/13" } ], "country": { "id": "US", "refName": "United States" }, "description": "US Tax Type", "doesNotAddToTotal": true, "id": "13", "isInactive": false, "name": "US Tax Type", "nexusAccounts": { "links": [ { "rel": "self", "href": "https://barochpatrik-runbox-parity-deuf1-001.eng.netsuite.com/services/rest/record/v1/taxtype/13/nexusAccounts" } ] }, "postToItemCost": false, "reverseCharge": false, "taxInNetAmount": false
}
```

Term

NetSuite exposes the term record to REST web services. Terms are used to specify when payment is due on your customers' invoices. Define the specific requirements of a term of payment by creating a term record. You can create different payment terms for different customers. In the UI, this is a user defined list at Setup > Accounting > Setup Tasks > Accounting Lists > New. For more information about payment terms, see the help topic Creating Terms of Payment.

Record ID

The record ID for the term REST record is **term**.

Subrecord

The term record has the CustomInstallmentPercent subrecord.

Note that certain fields become available only after you enable the Installments feature.

There are two types of terms: standard and date driven. The type is determined by the dateDriven field. Certain fields are valid only for one of the term types.

The REST API Browser includes information about the field names and field types of the estimate record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's term reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Code Samples

The code samples in this section show common use cases for standard and date driven terms.

Creating a Standard Term

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/term
{ "dateDriven": false, "name": "St Term name", "daysuntilnetdue": 15, "discountpercent": 10, "daysuntilexpiry": 5
}
```

Creating a Date Driven Term

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/term
```

```
{ "dateDriven": true, "name": "DD Term name", "dayofmonthnetdue": 15, "duenextmonthifwithindays": 10, "discountpercentdatedriv
en": 5, "daydiscountexpires": 5
}
```

Updating a Standard Term

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/term/{id}
{ "name": "Updated St Term", "daysuntilnetdue": 17, "discountpercent": 17, "daysuntilexpiry": 7 }
```

Updating a Date Driven Term

```
PATCH https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/term/{id}
{ "name": "Updated DD Term", "dayofmonthnetdue": 17, "duenextmonthifwithindays": 7, "discountpercentdatedriven": 7, "daydiscountex
pires": 7
}
```

Time Bill (Track Time)

A time transaction, also known as TimeBill, records the hours worked by an employee. You can use this transaction to record billable hours and invoice customers. This transaction is available when the Time Tracking feature is enabled.

Important: The TimeBill record is labelled as the Track Time record in the UI.

The REST API Browser includes information about the field names and field types of the time bill record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's time bill reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a time bill REST record is **timebill**.

Prerequisites

You must enable the Time Tracking feature before you can use this record through REST web services.

Code Sample

The following sample shows how to update a timebill record with id=28, where you want to change the number of hours to 4:00 and set a memo explaining what occurred.

```
PATCH https://3553905.suitetalk.api.netsuite.com/services/rest/record/v1/timebill/28Request headers
Content-Type: application/json
Authorization: OAuth realm="3553905",oauth_consumer_key="secret",oauth_token="secret",oauth_signature_method="H
MAC-SHA256",oauth_timestamp="1643201657",oauth_nonce="iuZQtDG47ET",oauth_version="1.0",oauth_signature="uW43ATVUbQM
CFD6TE%2FloGHUhqkQQiZ59x00YnUOtJAo%3D"
User-Agent: PostmanRuntime/7.29.0
```

```
Accept: */*
Postman-Token: c559cac0-1f7a-4275-9037-4c97ff85cb57
Host: 3553905.suitetalk.api.netsuite.com
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Content-Length: 71
Cookie: NS_ROUTING_VERSION=LAGGING
Request Body
{ "memo": "I updated the hours to reflect reality", "hours": "4:00" }
```

Topic

Knowledge base solutions are organized by a hierarchy of topics and subtopics. When you publish a topic, you also publish its subtopics and the solutions attached to these subtopics.

Use the topic record to create new topics and expose them to REST web services.

To learn more, see Support Case.

This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the topic record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's topic reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a topic REST record is **topic**.

Prerequisites

You must enable Enable Features > CRM > Support > Knowledge Base feature and Permissions > Lists > Knowledge Base before you can use this record through REST web services.

Elements Not Accessible Through REST Web Services

The following elements are not accessible through REST web services:

- subtopics
- solutions
- notes
- publish

Usage Notes

Review the following notes on working with the topic REST record:

- `title` is required, all other fields are optional

Code Sample

The following samples show common use cases for topic.

Creating a Topic

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/topic
Body: { "title": "REST title", "description": "<p>A Description</p>", "isInactive": false, "longDescription": "<p>Long Description</p>", "parentTopic": 2
}
```

Retrieving a Topic

```
GET
https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/topic/4
```

Updating a Topic

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/topic/4
Body:{ "title": "Updated REST title", "description": "<p>Updated Description</p>", "isInactive": true, "longDescription": "<p>Updated Description</p>", "parentTopic": 3
}
```

Deleting a Topic

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/paymentItem/4
```

Transfer Order

A transfer order transaction records the change in location of inventory items within the same subsidiary. It also records the quantity and shipping details.

This transaction is available when the Locations feature and the Multi-Location Inventory (MLI) feature are enabled. A transfer order can initialize item fulfillment and item receipt transactions. It has one subrecord: inventory detail. See the help topics Item Fulfillment and Item Receipt. For more information about transfer orders, see the help topic Inventory Transfer Orders.

The REST API Browser includes information about the field names and field types of the transfer order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's transfer order reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a transfer order REST record is **transferorder**.

The record ID for an inventory detail REST subrecord is **inventorydetail**.

Prerequisites

There are no prerequisites for using the record in REST.

Code Samples

The following sample shows you how to update a transfer order:

```
PATCH {{REST_SERVICES}}/record/v1/transferorder/118 { "memo": "test", "currency": { "id": "1", "refName": "British pound"
 }, "incoTerm": { "id": "1", "refName": "DAP" }, "item": { "items": [ { "amount": 0.0, "inventoryDetail": { "inventoryAssignment":
 { "items": [], "totalResults": 0 } }, "isClosed": true, "item": { "id": "20" }, "itemType": { "id": "InvtPart" }, "line": 1, "quan
tity": 10.0, "rate": 0.0 }, { "amount": 0.0, "inventoryDetail": { "inventoryAssignment": { "items": [], "totalResults": 0
 } }, "isClosed": true, "item": { "id": "21" }, "itemType": { "id": "InvtPart" }, "line": 4, "quantity": 5.0, "rate": 0.0
 } ] }, "location": { "id": "1", "refName": "WMS Warehouse 1" }, "orderStatus": { "id": "B", "refName": "B" }, "subsidiary":
 { "id": "1", "refName": "WMS" }, "transferlocation": { "id": "2", "refName": "WMS Warehouse 2" }, "status": { "id": "Closed", "ref
Name": "Closed" } }
```

Unit of Measure

A unit of measure record exposes a unit of measure to REST web services. The unit of measure is also known as a unit type. This record is not a subrecord.

The REST API Browser includes information about the field names and field types of the unit type record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's unitsType reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a unit of measure REST record is **unitsType**.

Code Samples

The following samples show common use cases for unit of measure. The example ID is 4.

Creating a Unit of Measure Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType
{ "name": "unit A", "uom": { "items": [ { "abbreviation": "UA", "baseUnit": true, "conversionRate": 1.0, "inUse": false, "pluralAb
breviation": "UsA", "pluralName": "units A", "unitName": "unit A" } ] }
}
```

Retrieving a Unit of Measure Using a GET Request

This sample shows how to retrieve an existing units type.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType/4
```

This sample shows how to retrieve an existing units type unit of measure with the internal id of 567.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType/4/uom/567
```

Updating a Unit of Measure Using a PATCH Request

This sample shows how to update an existing units type.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType/4
{
"name": "unit B",
" description": "Unit description",
}
```

This sample shows how to update an existing unit of measure.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType/4
{ "abbreviation": "UBx", "baseUnit": true, "conversionRate": 1.0, "pluralAbbreviation": "UsBx", "pluralName": "units Bx", "unit
Name": "unit Bx"
}
```

Deleting a Unit of Measure Using a DELETE Request

```
DELETE https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/unitsType/4
```

Usage

A usage record exposes SuiteBilling usage to REST web services. This record:

- is not a subrecord
- has no subrecords

The REST API Browser includes information about the field names and field types of the usage record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's usage reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a usage REST record is **usage**.

Prerequisites

You must Enabling SuiteBilling Features before you can use this record through REST web services.

This record also requires, at a minimum, a customer, subscription, subscription plan, and subscription line.

Code Sample

This sample shows a common usage case.

```
POST https://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/usage
{ "memo": "Record of usage", "usageQuantity": 33, "usageDate": "2020-07-14", "customer": 5, "subscriptionPlan": 8, "usageSubscrip
tion": 208, "usageSubscriptionLine": 235
}
```

Vendor

A vendor is a company or person you purchase goods and services from. Vendor records track information about your vendors and enable you to view past transactions and communications with them. For information about working with vendor records in the UI, see the help topic Vendor Records Overview.

NetSuite exposes the vendor record to REST web services. This record enables you to manage a specific vendor record.

All elements on this record are accessible through REST web services.

The REST API Browser includes information about the field names and field types of the vendor record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendor reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a vendor REST record is **vendor**.

Code Samples

The following code sample shows how to retrieve information about a specific vendor record with the id 94:

```
GET https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/vendor/94
```

The following code sample shows how to create a new vendor record:

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/vendor
{ "id": "95", "legalName": "AU Vendor"
}
```

Vendor Bill

Vendor bills are records which you can use to track payables as they arrive from vendors.

For information about working with vendor bills in the UI, see the help topic Vendor Bills.

NetSuite exposes the vendor bill record to REST web services. This record enables you to manage a specific vendor bill record.

The REST API Browser includes information about the field names and field types of the vendor bill record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorBill reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

The vendor bill record contains the following subrecord:

- billingAddress

The following elements are not accessible through REST web services:

- expensePlanMessage
- accountingBookDetail
- taxDetails

Record ID

The record ID for a vendor bill REST record is **vendorBill**.

Code Samples

The following samples show some common actions for a vendor bill.

Creating a Vendor Bill

```
POST {{COMPANY_URL}}/services/rest/record/v1/vendorBill
{ "entity":{"id": {{vendorId}} }, "item": { "items": [{ "item": {"id": {{ItemId}} }, "rate": 10 }] }
}
```

Reading a Vendor Bill

```
GET {{COMPANY_URL}}/services/rest/record/v1/vendorBill/{{id}}?expandSubResources=true
```

Updating a Vendor Bill

```
PATCH {{COMPANY_URL}}/services/rest/record/v1/vendorBill/{{id}}
{ "memo": "This is a test"
}
```

Deleting a Vendor Bill

```
DELETE {{COMPANY_URL}}/services/rest/record/v1/vendorBill/{{id}}
```

Vendor Category

A vendor category is a record that is used to group vendors. For information about creating vendor categories in the UI, see the help topic Setting Up Accounting Lists.

NetSuite exposes the vendor category record to REST web services. This record is not a subrecord and does not contain subrecords.

The REST API Browser includes information about the field names and field types of the vendor category record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorCategory reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a vendor category REST record is **vendorcategory**.

Code Sample

The following code sample shows how to create a vendor category:

```
POST {{REST_SERVICES}}/record/v1/vendorCategory Body:
{ "name":"name"
}
```

Vendor Credit

A vendor credit transaction creates a credit from a vendor that can be applied to a payables account. For example, a vendor credit transaction may occur when items are returned to a vendor or when a discount is negotiated with a vendor.

For information about working with vendor credits in the UI, see the help topic Vendor Credits.

NetSuite exposes the vendor credit record to REST web services. This record contains the subrecord **billingAddress**.

The REST API Browser includes information about the field names and field types of the vendor credit record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendor credit reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a vendor credit REST record is **vendorcredit**.

Code Sample

The following code sample shows how to create a vendor credit record:

```
POST {{REST_SERVICES}}/record/v1/vendorCredit Body:
{ "entity": { "id": {{vendorId}} }, "item": { "items": [{ "item": { "id": {{itemId}} }, "rate": 10, "location":{ "id":
 {{locationId}} } }] }
}
```

Vendor Payment

A vendor payment transaction posts to the general ledger as an expense and the amount of the payment is deducted from your accounts payable total. A vendor payment can be applied to one or more vendor bills. Vendor payments can help to track expenditures and total payables due.

For help working with this record in the UI, see the help topic Vendor Payments Overview.

The vendor payment record type includes the following actions from the Actions Framework:

- New
- Fax
- Show activity
- Go to register
- GL Impact

The REST API Browser includes information about the field names and field types of the vendor payment record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorPayment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a vendor payment REST record is **vendorPayment**. Vendor payment has one subrecord, with the record ID **payeeAddress**.

Prerequisites

There are no prerequisites to using this record through REST web services.

Code Samples

The following examples show how to read, create, update, or delete a vendor payment:

Read:

```
GET {{REST_SERVICES}}/record/v1/vendorPayment/${transaction-id}
```

Create:

```
POST {{REST_SERVICES}}/record/v1/vendorPayment { "entity": { "id": "${entity-id}" }, "apply": { "items": [ { "doc":
 { "id": "${bill-id}" }, "apply": true, "amount": ${amount} } ] }
}
```

Update:

```
PATCH {{REST_SERVICES}}/record/v1/vendorPayment/${transaction-id}

{
"externalId" : "AAA"
}
```

Delete:

```
DELETE {{REST_SERVICES}}/record/v1/vendorPayment/${transaction-id}
```

Vendor Prepayment

A vendor prepayment is a posting transaction that impacts the general ledger without offsetting the accounts payables account. When the vendor prepayment is applied, the accounts payables account is offset. Vendor prepayments record and track payments to vendors before they accept a purchase order for a good or service. You can apply these prepayment amounts against open bills for the vendor.

For information about working with vendor prepayments in the UI, see the help topic Entering Vendor Prepayments.

NetSuite exposes the vendor prepayment record to REST web services. This record is not a subrecord and does not contain subrecords.

The REST API Browser includes information about the field names and field types of the vendor prepayment record. It also includes information about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendor prepayment reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record ID

The record ID for a vendor prepayment REST record is **vendorprepayment**.

Prerequisites

To use the vendor prepayment REST record, you must have the Vendor Prepayments feature enabled, and a default vendor prepayment account set. The account can be a new or existing account that is an Other Current Asset type. This is the account that NetSuite selects by default when entering a new vendor prepayment transaction.

Code Sample

The following code sample shows how to create a vendor prepayment record:

```
POST {{REST_SERVICES}}/record/v1/vendorPrepayment Body:
{ "entity": { "id": {{entityId}} }, "account": { "id": {{accountId}} }, "payment": amount
}
```

Vendor Prepayment Application

After you make a prepayment to a vendor, you must apply the prepayment against the vendor bills. A vendor prepayment application is a posting transaction that impacts the general ledger. The Prepayment

account is credited, offsetting the Accounts Payable account. The application decreases the total amount you must pay to your vendor.

For information about applying vendor prepayments in the UI, see the help topic Vendor Prepayment Application.

NetSuite exposes the vendor prepayment application record to REST web services. This record is not a subrecord and does not contain subrecords.

The REST API Browser includes information about the field names and field types of the vendor prepayment application record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorPrepaymentApplication reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for a vendor prepayment application REST record is **vendorprepaymentapplication**.

Prerequisite

To use the vendor prepayment application REST record, you must have the Vendor Prepayments feature enabled, and a default vendor prepayment account set. This account is what NetSuite selects by default when entering a new vendor prepayment transaction.

Exception

On the vendor prepayment application record, the Bills subtab is not accessible through REST web services.

Code Sample

The following example shows how to create a vendor prepayment application by transforming a vendor prepayment into a vendor prepayment application. This transformation process applies the vendor prepayment.

```
POST {{REST_SERVICES}}/record/v1/vendorPrepayment/{{vendorPrepaymentId}}/!transform/vendorPrepaymentApplication Body:
{ "bill": { "bills": [{ "line": {{lineId}}, "apply": true }] }
}
```

Vendor Return Authorization

When you track vendor returns using authorizations, you know the status of each return and you can track accurate information about each return you process. The vendor return workflow includes obtaining Return Materials Authorization (RMA) from the vendor, then processing the return in NetSuite to track information about each return.

For information about working with vendor return authorizations in the UI, see the help topic Vendor Return Authorization Overview.

NetSuite exposes the vendor return authorization record to REST web services. This record enables you to manage a specific vendor return authorization record.

The REST API Browser includes information about the field names and field types of the vendor return authorization record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorReturnAuthorization reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

The vendor return authorization record contains the following subrecords:

- itemSublist
- expenseSublist
- billingAddress

The vendor return authorization record contains fields that are related to taxation features.

REST web services do not support legacy tax features. To work with taxation through REST web services you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

Record ID

The record ID for a vendor return authorization REST record is **vendorReturnAuthorization**. To use the vendor return authorization REST record, you must have the Vendor Return Authorization feature enabled.

Code Samples

The following samples show some common actions for a vendor return authorization.

Creating a Vendor Return Authorization

```
{ "entity": { "id": "36", "refName": "EU Vendor" }, "item": { "items": [{ "item": { "id": "89" }, "quantity": 5 }] }
}
```

Updating a Vendor Return Authorization

```
{ "entity": { "id": "38" }, "currency": {"id": 1}
}
```

Vendor Subsidiary Relationship

NetSuite exposes the vendor subsidiary relationship record to REST web services. This record enables you to manage a specific vendorSubsidiaryRelationship record. You can also get a list of vendor subsidiary relationship records. This record is not accessible through the user interface.

The REST API Browser includes information about the field names and field types of the vendor subsidiary relationship record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's vendorSubsidiaryRelationship reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record ID

The record ID for the vendor subsidiary relationship REST record is **vendorSubsidiaryRelationship**.

Prerequisites

You must either use NetSuite OneWorld or have the Subsidiaries hidden feature enabled before you can use this record through REST web services.

Updatable Fields on the vendorSubsidiaryRelationship Record

You can update the following fields on the vendorSubsidiaryRelationship record:

- credit limit (available on the vendor record, Subsidiary subtab)
- tax item (available on the vendor record, Subsidiary subtab)
- custom fields (available on the vendor record, Subsidiary subtab)
- (externalid) - external id

Supported Operations

The following operations are supported for REST web services:

- GET (Read, Search for a list of vendor subsidiary relationship records)
- POST (insert a record)
- DELETE (a specific record)
- GET (Read, Search a specific record)
- PATCH (update a specific record)
- PUT (insert or update a specific record)

Tax-related fields are not supported in REST.

Code Samples

The following code sample shows how to get a specific vendorSubsidiaryRelationship record:

```
GET https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/vendorsubsidiaryrelationship?q=entity
EQUAL 13 AND subsidiary EQUAL 5
```

The following code sample shows how to get the ID of a specific vendorSubsidiaryRelationship record:

```
POST https://123456.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5
```

```
Header:
'prefer: transient'
'Content-Type: application/json'
Body:
'{ "q": "SELECT id FROM vendorSubsidiaryRelationship WHERE entity = 13 AND subsidiary = 5"
```

```
}'
```

The following code sample shows how to set the externalId and creditLimit for a specific vendorSubsidiaryRelationship record:

```
PATCH 'https://123456.suitetalk.api.netsuite.com/services/rest/record/v1/vendorsubsidiaryrelationship/1'
```

```
Header:
'Content-Type: application/json'
Body:
'{ "externalId": "VSR1", "creditLimit": 150
}'
```

Website

A website record exposes a website to REST web services.

This record has the following subrecords:

- imagesize
- siteentrypoint
- websitefieldset

Note: REST web services do not support legacy tax features. To work with taxation through REST web services, you must have the SuiteTax feature enabled. For more information about using SuiteTax, see the help topic SuiteTax.

The REST API Browser includes information about the field names and field types of the website record. It also includes the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's website reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for the website REST record is **website**.

Prerequisites

You must enable the Website feature to be able to use the website record.

Limitations

The following sublists are not accessible through REST web services:

- Storetab
- Sitecurrency
- Sitelanguage
- Cartcolumn
- Imagisize

Additionally, a number of fields are also not accessible. For the list of accessible fields, see the REST API browser.

The website record does not work with external ID.

Code Samples

The following code sample shows how to load website data from a website with the ID 1.

```
GET {{COMPANY_URL}}/services/rest/record/v1/website/1
```

The following code sample shows how to find a website by name.

```
GET {{COMPANY_URL}}/services/rest/record/v1/website?q=displayName START_WITH "My website"
```

Weekly Timesheet

With weekly time tracking, you can track the hours you and other employees work for a week at a time.

Weekly timesheets work in conjunction with the existing Time Tracking feature to help you customize how you capture time entries in a weekly format.

For help working with this record in the UI, see the help topics Managing Time Tracking and Weekly Time Tracking.

The timesheet record type includes the following actions from the Actions Framework:

- Approve
- Reject
- Submit
- Retract

The REST API Browser includes information about the field names and field types of the timesheet record and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's time sheet reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a timesheet REST record is **timesheet**.

Prerequisites

This record is available only when the Time Tracking and Weekly Timesheets features are enabled. For more information about how to enable these features, see the help topics Managing Time Tracking and Weekly Time Tracking.

Limitations

The following fields on this record are read-only:

- total hours
- hours total
- approval status
- submitted hours
- rejected hours
- allocated hours
- work calendar hours
- lock status

This record does not support the Update operation through REST.

Code Samples

The following examples show how to create a new timesheet and approve an existing timesheet.

Create a new timesheet:

```
POST /services/rest/record/v1/timesheet HTTP/1.1
Host: yourServer
Content-Type: application/json
Authorization: yourAuthorizationToken
Content-Length: 46
{
"employee": "Your_ID", "tranDate" : "YYYY-MM-DD"
}
```

Approve an existing timesheet

```
POST /services/rest/record/v1/timesheet/346/@approve HTTP/1.1
Host: burespetr-runbox-ml-deuf1-001.eng.netsuite.com
Content-Type: application/json
Authorization: yourAuthorizationToken
```

Work Order

A work order record exposes a work order to REST web services. This record is not a subrecord.

This record:

- is not a subrecord
- has one subrecord: item

The REST API Browser includes information about the field names and field types of the work order record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's workOrder reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a work order REST record is **workOrder**.

Prerequisites

You must enable the Work Orders feature.

Code Samples

The following samples show common use cases for work orders. The example ID is 4.

Creating a Work Order Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrder
{ "assemblyItem": {"id": "66"}, "subsidiary": {"id": "5"}, "location": {"id": "5"}, "quantity": 5, "isWip": true
}
```

Converting Work Orders Using PATCH Requests

The following code sample shows how to convert a work order to a work order issue.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/!transform/workOrderIssue
```

The following code sample shows how to convert a work order to a work order completion.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/!transform/workOrderCompletion
{ "startOperation": {"refName": "10"}, "endOperation": {"refName": "20"}
}
```

The following code sample shows how to convert a work order to a work order completion with backflush.

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/!transform/workOrderCompletion
{ "isBackflush": true, "startOperation": {"refName": "10"}, "endOperation": {"refName": "40"}, "component" : { "items":
 [{ "item": "59", "quantity": 1 }] }
}
```

The following code sample shows how to convert a work order to a work order close.

```
https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/!transform/workOrderClose
```

The following code sample shows how to create an assembly build from a non-WIP work order.

```
https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/!transform/assemblyBuild
{ "location": {"id": 5}
}
```

Creating a Work Order With Items Using a POST Request

```
POST https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrder
{ "subsidiary": { "id": "5" }, "assemblyItem": { "id": "67" }, "quantity": 1.0, "item": { "items": [{ "item": { "id": 63 } },
 { "item": { "id": 65 } }] }
}
```

Retrieving a Work Order Using a GET Request

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrder/4
```

Updating Work Orders Using a PATCH Request

The following code sample shows how to add an item to the item members of a work order.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrder/4
{ "item": { "items": [ { "item": { "id": 63 }, "quantity": 6 }] }
}
```

The following code sample shows how to update a member item of a work order. The item ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/record/v1/workOrder/4/item/7
{ "quantity": 55
}
```

Work Order Close

A work order close record exposes a work order close to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - component
  - routingItem

The REST API Browser includes information about the field names and field types of the work order close record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's workOrderClose reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a work order close REST record is **workOrderClose**.

Prerequisites

You must enable the Manufacturing Work In Process feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for work order closes. The example ID is 4.

Retrieving a Work Order Close Using a GET Request

A work order close is created by doing a transformation on a work order.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderClose/4
```

Work Order Completion

A work order completion record exposes a work order completion to REST web services.

This record:

- is not a subrecord
- has the following subrecords:
  - component
  - operation

The REST API Browser includes information about the field names and field types of the work order completion record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's workOrderCompletion reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a work order completion REST record is **workOrderCompletion**.

Prerequisites

You must enable the Manufacturing Work In Process feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for work order completion. The example ID is 4.

Retrieving a Work Order Completion Using a GET Request

A work order completion is created by doing a transformation on a work order.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderCompletion/4
```

Updating a Work Order Completion Using a PATCH Request

The following code sample shows how to edit the setup time in the operation sequence. The ID for the operation is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderCompletion/4/operation/7
{ "laborSetupTime": 20.0
}
```

The following code sample shows how to edit the component with backflush. The component ID is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderCompletion/4/component/lineNumber=7
{ "quantity": 3
}
```

Work Order Issue

A work order issue record exposes a work order issue to REST web services.

This record:

- is not a subrecord
- has one subrecord: component

The REST API Browser includes information about the field names and field types of the work order issue record, and about the HTTP methods, request parameters, and operations available to this record. For details, see the REST API Browser's workOrderIssue reference page.

For information about using the REST API Browser, see the help topic The REST API Browser.

Record IDs

The record ID for a work order issue REST record is **workOrderIssue**.

Prerequisites

You must enable the Manufacturing Work In Process feature before you can use this record through REST web services.

Code Samples

The following samples show common use cases for work order issues. The example ID is 4.

Retrieving a Work Order Issue Using a GET Request

A work order issue is created by doing a transformation on a work order.

```
GET https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderIssue/4
```

Updating a Work Order Issue Using a PATCH Request

The following code sample shows how to edit the quantity of the component member item. The ID for the item is 7.

```
PATCH https://demo123.suitetalk.api.snap.netsuite.com/services/rest/record/v1/workOrderIssue/4/component/item=/7
{ "quantity": 5
}
```

REST Web Services Tutorials

The following sections contain end-to-end guidelines for specific business scenarios.

Sales Order Use Cases

This section includes the following sample use cases for managing sales order transactions:

A sales order is a transaction that records a commitment to sell items or services to a customer. Sales orders have no accounting impact until items ship or services are completed. For more information on NetSuite sales orders, see the help topic Creating Sales Orders.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).

Variable	Definition
{{*_ID}}	These variables refer to the internal ID of the record. For example, ESTIMATE_ID or SALES_ORDER_ID.

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case For Creating Your Sales Order

There are two ways you can create a sales order:

- Create a Sales Order from an Estimate
- Create a New Stand-Alone Sales Order

Create a Sales Order from an Estimate

To create a sales order from an existing estimate, use the transform function in the REST API:

```
POST {{REST_SERVICES}}/record/v1/estimate/{{ESTIMATE_ID}}/!transform/salesOrder
```

This function takes an estimate with an ID of {{ESTIMATE_ID}} and transforms it into a sales order. You can also specify in the body of the request which fields you want to change during the transformation.

```
POST {{REST_SERVICES}}/record/v1/estimate/{{ESTIMATE_ID}}/!transform/salesOrder

{
    "memo": "Transformed Estimate to Sales Order!"
}
```

Create a New Stand-Alone Sales Order

In the Basic Sales Order example, you can create a sales order without filling in the details of the other fields.

The system automatically calculates many fields in the sales order based on the value of the entity field.

The transaction Date (or tranDate in REST) is the creation date of the sales order. If you leave the field blank, it defaults to the current date.

For a complete list of fields you can populate on a sales order, see the Sales Order Records Browser.

Use the Custom Form field (customform in REST) to determine what type of record the system creates after you bill a sales order. For more information about the custom form field, see the help topic Creating Sales Orders.

Custom Form Types include:

- Standard Sales Order
- Standard Sales Order - Cash Sale

- Standard Sales Order - Invoice
- Standard Sales Order - Progress Billing

```
POST {{REST_SERVICES}}/record/v1/salesOrder
```

Basic Sales Order

```
{
    "entity": {
        "id": "110"
    },
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1
            }
        ]
    }
}
```

More Complicated Sales Order

```
{
    "entity": {
        "id": "110"
    },
    "customForm": "68",
    "tranDate": "2020-6-15",
    "otherrefnum": "11037",
    "memo": "This is a test memo!",
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1,
                "Description": "Test Item"
            }
        ]
    }
}
```

Use Case For Applying a Promotion to Your Sales Order

To apply a promotion to a sales order, provide the ID value to the `promocode` field.

You can do this when you create a new sales order or when you make an update to an existing sales order.

Applying a Promotion to a Sales Order

```
POST {{REST_SERVICES}}/record/v1/salesOrder

{
    "entity": {
        "id": "110"
```

```
    },
    "promocode": { "id": 1 },
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1
            }
        ]
    }
}
```

Apply a Discount Item to a Sales Order

If you have a discount item you want to apply to a sales order, you can either add in the discount item when you create a new sales order or when you make an update to an existing sales order. For the `discountitem id`, you specify the internal ID of the discount item.

Applying a Discount Item When Creating a New Sales Order

```
POST {{REST_SERVICES}}/record/v1/salesOrder

{
    "entity": {
        "id": "110"
    },
    "discountitem": { "id": 99 };
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1
            }
        ]
    }
}
```

Applying a Discount Item When Updating an Existing Sales Order

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}

{
    "discountitem": { "id": 99 }
}
```

Use Case For Retrieving Your Sales Order

Use a GET request with the Sales Order's ID number to retrieve the sales order data.

```
GET {{REST_SERVICES}}/record/v1/salesOrder/{{ID}}
```

In the response body, you can see all the data pertaining to the sales order. The data includes the specific field values that you set, as well as default values for the fields that you did not set.

Each of the sublists has its own endpoint for you to gather field data for each line.

GET Request Results


```
{
    "links": [
        {
            "rel": "self",
            "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesOrder/18"
        }
    ],
    "altSalesTotal": 0.0,
    "balance": 0.0,
    "balreadyrefunded": "F",
    "billingaddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/billingaddress"
            }
        ]
    },
    "canBeUnapproved": true,
    "carrier": "nonups",
    "ccApproved": false,
    "ccProcessAsPurchaseCard": false,
    "checkCommitted": false,
    "createdDate": "2020-07-29T08:34:00Z",
    "credholdentity": 110,
    "credholdoverride": "F",
    "currency": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/currency/1"
            }
        ],
        "id": "1",
        "refName": "USA"
    },
    "currencyName": "USA",
    "currencyPrecision": 2,
    "currencysymbol": "USD",
    "customForm": "68",
    "defaultCatchUp": 92,
    "deferredrevenue": 0.0,
    "discountTotal": 0.0,
    "edition": "US",
    "entity": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/customer/110"
            }
        ],
        "id": "110",
        "refName": "Anonymous Customer"
    },
    "entity_nexus_country": "US",
    "entityfieldname": "entity",
    "entityNexus": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/nexus/1"
            }
        ],
        "id": "1",
        "refName": "US-CA"
    },
    "exchangeRate": 1.0,
    "excludecommission": false,
    "getAuth": false,
    "giftCertApplied": 0.0,
    "giftCertRedemption": {
```

```
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/giftCertRedemption"
            }
        ]
    },
    "hasanydelayedrevrec": false,
    "hasFedexFreightService": false,
    "hasprecalcs": false,
    "id": "18",
    "ignoreAvs": false,
    "ignoreAvsVis": false,
    "ignoreCsc": false,
    "ignoreCscVis": false,
    "installmentcount": 0,
    "invertrevrecschedule": false,
    "iseitf81on": false,
    "isOnlineTransaction": "F",
    "isRecurringPayment": false,
    "item": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/item"
            }
        ]
    },
    "itemShippingCostFxRate": 1,
    "lastModifiedDate": "2020-07-29T08:34:00Z",
    "linkedTrackingNumberList": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/linkedTrackingNumberList"
            }
        ]
    },
    "manualCreditHold": "F",
    "needsbill": "F",
    "needsPick": true,
    "nextbill": "2020-07-29",
    "nexus": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/nexus/1"
            }
        ],
        "id": "1",
        "refName": "US-CA"
    },
    "nexus_country": "US",
    "oldrevenuecommitment": false,
    "ordbilled": "F",
    "orderStatus": "A",
    "ordpicked": false,
    "ordrecvd": false,
    "originalNexus": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/nexus/1"
            }
        ],
        "id": "1",
        "refName": "US-CA"
    },
    "originalNexusCountry": "US",
    "overallbalance": 0.0,
    "overallunbilledorders": 39.95,
    "paymentSessionAmount": 39.95,
    "payPalOverride": false,
```

```
    "payPalProcess": false,
    "prevDate": "2020-07-29",
    "primarycurrency": 1.0,
    "primarycurrencyfxrate": 1.0,
    "recognizedrevenue": 0.0,
    "revenuestatus": "A",
    "saleseffectivedate": "2020-07-29",
    "salesTeam": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/salesTeam"
            }
        ]
    },
    "shipComplete": false,
    "shipDate": "2020-07-29",
    "shipIsResidential": false,
    "shipItemHasFreeShippingItems": false,
    "shipOverride": false,
    "shippingAddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/salesorder/18/shippingAddress"
            }
        ]
    },
    "shippingCostOverridden": false,
    "status": "A",
    "storeorder": "F",
    "subsidiary": {
        "links": [
            {
                "rel": "self",
                "href": "https://runbox3.corp.netsuite.com/services/rest/record/v1/subsidiary/1"
            }
        ],
        "id": "1",
        "refName": "Parent Company"
    },
    "subtotal": 39.95,
    "taxTotal": 0.0,
    "threedStatusCodeVis": false,
    "toBeEmailed": false,
    "toBeFaxed": false,
    "toBePrinted": false,
    "total": 39.95,
    "tranDate": "2020-07-29",
    "tranId": "1",
    "tranisvsoebundle": false,
    "unbilledOrders": 39.95,
    "usingVSOEFldSet": false,
    "vsoeautocalc": false,
    "warnNexusChange": false,
    "webstore": "F",
    "weekendpreference": "ASIS"
}
```


Use Case For Updating Your Sales Order

Some common updates to an existing sales order include the following options:

- Update Field Values
- Update a Field for a Specific Line
- Add a New Transaction Line Without Updating a Field Value
- Remove an Item

Update Field Values

To update a specific field on the sales order, send a PATCH request with the changed field in the body.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}

{     "orderStatus": "B" }
```

Update a Field for a Specific Line

Updating a transaction line item is similar to updating a field by sending a PATCH request with the changed value in the body.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}
```

To update a field for a specific line, specify the line number in the body of your request.

Updating an Existing Item

```
{
    "item": {
        "items": [
            {
                "line": 1,
                "isclosed": true
            }
        ]
    }
}
```

Add a New Transaction Line Without Updating a Field Value

To add a new transaction line without updating a field value, leave out the line number. The request creates a new transaction line for the sales order.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}
```

Creating a New Line Item

```
{
    "item": {
        "items": [
            {
            "item": { "id": 98 },
            "quantity": 2
            }
        ]
    }
}
```

Remove an Item

After saving a sales order, you can use the sales order's ID to remove an item by using the following PATCH call. Using an existing sales order, update `{{salesOrderId}}` with the ID of the sales order you wish to modify and specify the new items. When specifying the new items in the sales order, make sure the item you wish to remove is no longer listed.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{salesOrderId}}?replace=item
```

Note: Currently, there is no option to individually delete a specific line through REST, so you can make the lines optional and let the sales person enforce this rule.

Removing a Line Item (Future)

```
{
   "item": {
      "items": [{
         "item": { "id": {{itemId}} },
         "rate": 40
      }, {
         "item": { "id": {{itemId}} },
         "rate": 50
      }]
   }
}
```

Use Case For Approving Your Sales Order

After you create a sales order, the `orderStatus` field of the sales order is initially set to value "A", and later changes to value "B". These letters denote the state of the sales order in the approval workflow as follows:

orderStatus	Approval Status
A	Pending Approval
B	Pending Fulfillment

To create a preapproved sales order, set the `orderStatus` in the POST request to "B".

Setting Sales Order Approval Status

```
POST {{REST_SERVICES}}/record/v1/salesOrder

{
    "entity": {
        "id": "110"
    },
   "orderStatus": "B",
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1
            }
        ]
    }
}
```

If you want to create a sales order that requires approval after you create it, send in a PATCH request with `orderStatus "B"` in the body.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}

{
   "orderStatus": "B"
}
```

Reset Approval Status

If you already approved the sales order but need to reset the approval status, send a PATCH request with the updated `orderStatus` in the body.

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}

{
   "orderStatus": "A"
}
```

Close a Sales Order

To close the sales order, close individual line items on the sales order using the following PATCH call and body.

Patch Call for Closing a Sales Order

```
PATCH {{REST_SERVICES}}/record/v1/salesOrder/{{SALES_ORDER_ID}}

{
   "item": {
        "items": [
            {
                "line": 1,
            "isclosed": true
            },
         {
                "line": 2,
            "isclosed": true
            },
         {
                "line": 3,
            "isclosed": true
            }
        ]
    }
}
```

Use Case For Fulfilling Your Sales Order

To start the sales order fulfillment process, create a fulfillment from your sales order. You can create a fulfillment by sending a POST request with the ID of the sales order you want to fulfill.

```
POST {{REST_SERVICE}}/record/v1/salesorder/{{SALES_ORDER_ID}}/!transform/itemfulfillment
```

The following code sample contains two orderLine groupings:

- In the first grouping, you partially fulfill an order by specifying a `quantity` of less than the full order (**"quantity": 1**).
- In the second grouping, you leave the order unfulfilled by setting the `itemreceive` parameter to false (**"itemreceive": false**).

```
{
    "item": {
        "items": [
            {
                "orderLine": 1,
                "location": 6,
                "quantity": 1
```

```
            },
            {
                "orderLine": 4,

                "location": 6,
                "itemreceive": false
            }
        ]
    }
}
```

Note: Like the provided code sample, your sales order may not have sequential order lines. For example, `"orderLine": 4` follows `"orderLine": 1` in the code sample.

To check the correct `orderLine` number for an item, use a GET call to retrieve the sales order item information through REST as show here:

```
GET {{REST_SERVICES}}/record/v1/salesOrder/{{ID}}/item
```

Use Case For Creating Invoices or Cash Sales from Your Sales Order

This use case shows you how to create an invoice or cash sale for all fulfilled lines on the sales order.

Create an Invoice

To create an invoice from a sales order, you need an approved sales order with the Custom Form field set to Standard sales order - Invoice and some fulfilled items.

```
POST {{REST_SERVICES}}/record/v1/salesorder/{{SALES_ORDER_ID}}/!transform/invoice
```

If you leave the request body empty, NetSuite creates an Invoice for all fulfilled items and quantities. However, if you plan to fulfill only part of the order, you need to be able to create a matching partial invoice. As shown in the example, you can create a partial invoice by specifying the quantity for each item line.

Invoice From a Sales Order

```
{
    "item": {
        "items": [
            {
                "orderLine": 1,
                "quantity": 3
            },
            {
                "orderLine": 2,
                "quantity": 0
            }
        ]
    }
}
```

Create a Cash Sale

To create a cash sale from a sales order, you need an approved sales order with the Custom Form field set to Standard Sales Order - Cash Sale and some fulfilled items.

```
POST {{REST_SERVICES}}/record/v1/salesorder/{{SALES_ORDER_ID}}/!transform/cashsale
```

If you leave the request body empty, NetSuite creates a cash sale for all fulfilled items and quantities. You can create a partial cash sale by specifying the quantity for each item line as shown in the following example.

Cash Sale From a Sales Order

```
{
    "item": {
        "items": [
            {
                "orderLine": 1,
                "quantity": 0
            },
            {
                "orderLine": 2,
                "quantity": 4
            }
        ]
    }
}
```

Use Case For Creating a Progress Sales Order

Creating a progress sales order is similar to creating a new sales order. For a progress sales order, change the custom form to Standard Sales Order - Progress Billing. You do this through REST by setting the custom form value when sending your REST request.

Custom Form Name	Custom Form ID
Standard Sales Order (default)	68
Standard Sales Order - Progress Billing	86

To complete the sales order, you need to add:

- a value for the entity field that corresponds to the customer on the sales order
- at least one item

You can also modify the following optional fields:

Field on the Sales Order	Corresponding Field on the Custom Form	Valid Values
date	trandate	YYYY-MM-DD
status	orderstatus	A - Pending Approval B - Pending Fulfillment
billing terms	terms	see the terms table in the database

For more information, see the help topic Creating Progress Sales Orders.

Progress Sales Order

```
POST {{REST_SERVICES}}/record/v1/salesorder

{
    "customform": 86,  //For custom form "Standard Sales Order - Progress Billing"
```

```
    "entity": {
        "id": "110"
    },
    "trandate": "2020-07-30",  //optional field
    "orderstatus": "B",  //optional field
    "terms": 3,  //optional field
    "item": {
        "items": [
            {
                "item": {
                    "id": 98
                },
                "quantity": 1
            },
            {
                "item": {
                    "id": 98
                },
                "quantity": 2
            },
            {
                "item": {
                    "id": 98
                },
                "quantity": 3
            }
        ]
    }
}
```

Use Case For Deleting a Sales Order

To delete a sales order, send the DELETE call to delete the specific record with the specified ID.

```
DELETE {{REST_SERVICES}}/record/v1/salesOrder/{{ID}}
```

SuiteBilling Use Cases

This section includes the following sample use cases for managing subscriptions using REST web services:

- Use Case For Managing Your Subscription Catalog
- Use Case For Managing Your Subscription Sales

Use Case For Managing Your Subscription Catalog

REST services provide a convenient channel to manage your subscription catalog.

In this use case, you create the necessary records using REST for your fictitious company, CED Cyber Solutions.

This use case demonstrates how to:

1. Create Service Items
2. Create a Subscription Plan
3. Create a Price Plan
4. Create a Price Book
5. Retrieve Records

Create Service Items

Before you begin, you need to create service items for your subscription plan. The security subscription plan that you want to sell includes the following items:

- Setup
- License
- Seats
- Bronze Support
- Silver Support
- Gold Support

You need to create a subscription term because you want your subscription plan to have an initial term of three years.

Although the subscription plan depends on these records, this use case does not cover how to create them through REST. For now, you can create them through the user interface.

After you create each record, make note of its internal ID so that you can reference it. You can find the internal ID of each record by navigating to the record's list page.

The following table lists prerequisite records along with the record type and example internal ID for each:

Record Type	Record Name	Example Internal ID
Service Items	License	1
Service Items	Setup	2
Service Items	Seats	3
Service Items	Bronze Support	4
Service Items	Silver Support	5
Service Items	Gold Support	6
Subscription Term	Three Years	1

Create a Subscription Plan

To create records, you send POST requests to the endpoint for the record type. The request body should contain information for all required fields and any optional fields.

Your requirements are as follows:

- The plan name is "DEFEND Package"
- Setup is a one-time line
- Licensing and the number of users (seats) are recurring lines
- Customers must choose one of the three support tiers

  Note: At present time, there is no way to enforce this behavior through NetSuite, so you can make the lines optional and let the sales person enforce this rule.
- All recurring lines should be prorated except for seats
- All lines are billed in advance

To set the initialTerm to the subscription term that you have created, you need to reference an existing record. Because you created a custom subscription term, wrap the ID of that term in an object with the ID property. Wrapping the ID in an object is commonly how records are referenced in REST.

Lines on subscription plans are specified in the member sublist. The member sublist is an object with an items property. The items property is an array of objects that each represent a line. The lines have their own fields that must be defined. Although the lineNumber can be inferred in some cases, it is best to always specify it. Lines are 1-indexed. This is the common structure for sublists in requests.

Some fields on the lines in the member sublist require enumeration values. Usually, the possible values are clearly defined in the Metadata Catalog. However, sometimes they can be unclear. For example, subscriptionLineType has an enumeration value of integer strings. Those strings map to subscription line types:

- **1** – One-Time
- **2** – Recurring
- **3** – Usage

A successful request returns a response with a HTTP status of **204 No Content**. In the response headers, there is a Location key with an associated value representing the endpoint at which you can access your newly created record. Make note of the ID so that you can reference your subscription plan later. The ID for your example subscription plan is 1.

Example Subscription Plan

```
{
    "itemId": "DEFEND Package",
    "initialTerm": { "id": "1" },
    "member": {
        "items": [
            {
                "lineNumber": 1,
                "item": { "id": "1" },
                "isRequired": true,
                "subscriptionLineType": "1",
                "billingMode": "IN_ADVANCE",
                "renewalOption": "DIFFERENT_PLAN"
            },
            {
                "lineNumber": 2,
                "item": { "id": "2" },
                "isRequired": true,
                "subscriptionLineType": "2",
                "prorateStartDate": true,
                "prorateEndDate": true,
                "billingMode": "IN_ADVANCE",
                "renewalOption": "ALWAYS"
            },
            {
                "lineNumber": 3,
                "item": { "id": "3" },
                "isRequired": true,
                "subscriptionLineType": "2",
                "prorateStartDate": false,
                "prorateEndDate": false,
                "billingMode": "IN_ADVANCE",
                "renewalOption": "ALWAYS"
            },
            {
                "lineNumber": 4,
                "item": { "id": "4" },
                "isRequired": false,
                "subscriptionLineType": "2",
                "prorateStartDate": true,
                "prorateEndDate": true,
                "billingMode": "IN_ADVANCE",
```

```
                "renewalOption": "ALWAYS"
            },
            {
                "lineNumber": 5,
                "item": { "id": "5" },
                "isRequired": false,
                "subscriptionLineType": "2",
                "prorateStartDate": true,
                "prorateEndDate": true,
                "billingMode": "IN_ADVANCE",
                "renewalOption": "ALWAYS"
            },
            {
                "lineNumber": 6,
                "item": { "id": "6" },
                "isRequired": false,
                "subscriptionLineType": "2",
                "prorateStartDate": true,
                "prorateEndDate": true,
                "billingMode": "IN_ADVANCE",
                "renewalOption": "ALWAYS"
            }
        ]
    }
}
```

Create a Price Plan

Your pricing requirements are as follows:

- Setup costs $100.00
- A license costs $1000.00
- Seats have a volume-based cost that decreases as your customer adds more users
  - For 0 - 20 seats, each seat costs $7.00
  - For 21- 50 seats, each seat costs $6.00
  - For 51+ seats, each seat costs $5.00
- Support prices change over time
  - Bronze starts at $30.00 and drops to $20.00 after the first year
  - Silver starts at $60.00 and drops to $40.00 after the first year
  - Gold starts at $90.00 and drops to $60.00 after the first year

There are six lines on your subscription plan, which means that you need at least six price plans. You need three more price plans to account for the price drop after the first year on your support lines. To satisfy these requirements, you need nine price plans.

Each price plan requires its own request.

Price Plan 1

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 100.00
            }
        ]
    }
```

```
}
```

Price Plan 2

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 250.00
            }
        ]
    }
}
```

Price Plan 3

```
{
    "currency": { "id": "1" },
    "pricePlanType": "4",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-101" },
                "value": 7.00
            },
            {
                "fromVal": 20,
                "pricingOption": { "id": "-101" },
                "value": 6.00
            },
            {
                "fromVal": 50,
                "pricingOption": { "id": "-101" },
                "value": 5.00
            }
        ]
    }
}
```

Price Plan 4

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 30.00
            }
        ]
    }
}
```

Price Plan 5

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
```

```
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 20.00
            }
        ]
    }
}
```

Price Plan 6

```
{
    "currency": { "id": 1 },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 60.00
            }
        ]
    }
}
```

Price Plan 7

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 40.00
            }
        ]
    }
```

Price Plan 8

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 90.00
            }
        ]
    }
}
```

Price Plan 9

```
{
    "currency": { "id": "1" },
    "pricePlanType": "2",
    "priceTiers": {
        "items": [
            {
                "fromVal": 0,
                "pricingOption": { "id": "-102" },
                "value": 60.00
```

```
            }
        ]
    }
}
```

Create a Price Book

Your requirements are as follows:

- Setup is a one-time cost
- License cost is billed annually and prorated by month
- Seats are billed monthly and prorated by day
- Support is billed monthly, prorated by day, and prices decrease after the first year

The subscriptionPlanLineNumber has an integer value and is the number of the line to which the price interval should apply. The sublist line number cannot be inferred by the system because the relationship between subscription plan lines and price intervals is one-to-many.

The repeatEvery is connected to the frequency and determines how often the line should be billed. It has an enum value instead of an integer. The values map directly to their integer values.

For one-time lines, you must specify that repeatEvery is 0 and prorateBy is an empty string.

The startOffsetUnit and startOffsetValue are related to time-based pricing. Combined, the two fields determine when the pricing of a certain line should follow the tiers in the referenced price plan. For example, the support lines (4, 5, and 6) have two price intervals each. The first interval defines the pricing for the first year as startOffsetUnit = YEAR and startOffsetValue = 1. The second interval defines the pricing for the second year as startOffsetUnit = YEAR and startOffsetValue = 2. This results in a reduced recurring price for support in every year after the first.

On a successful response, you have completed the process for creating a subscription plan and price plans linked by a single price book.

Price Book

```
{
    "subscriptionPlan": { "id": "1" },
    "currency": { "id": "1" },
    "name": "TEST Pricing",
    "priceInterval": {
        "items": [
            {
                "subscriptionPlanLineNumber": 1,
                "pricePlan": { "id": "10" },
                "frequency": "ONETIME",
                "repeatEvery": "0",
                "startOffsetUnit": "MONTH",
                "startOffsetValue": 1,
                "prorateBy": ""
            },
            {
                "subscriptionPlanLineNumber": 2,
                "pricePlan": { "id": "11" },
                "frequency": "ANNUALLY",
                "repeatEvery": "1",
                "startOffsetUnit": "MONTH",
                "startOffsetValue": 1,
                "prorateBy": "MONTH"
            },
            {
                "subscriptionPlanLineNumber": 3,
                "pricePlan": { "id": "12" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
```

```
                "startOffsetUnit": "MONTH",
                "startOffsetValue": 1,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 4,
                "pricePlan": { "id": "13" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 1,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 4,
                "pricePlan": { "id": "14" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 2,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 5,
                "pricePlan": { "id": "15" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 1,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 5,
                "pricePlan": { "id": "16" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 2,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 6,
                "pricePlan": { "id": "17" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 1,
                "prorateBy": "DAY"
            },
            {
                "subscriptionPlanLineNumber": 6,
                "pricePlan": { "id": "18" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "YEAR",
                "startOffsetValue": 2,
                "prorateBy": "DAY"
            }
        ]
    }
}
```

Retrieve Records

You can retrieve records using a GET request. You can also use a GET request for validating records after creation. The endpoint is similar to the POST requests. The only difference is that the ID of the record is appended to the end.

Example: In this example, the GET request returns the response body that follows.

Note: The response body is slightly different from the request body because there are other optional fields. These fields were set to their default values internally. The member sublist has its own endpoint that you can access to gather field data for each line.

This GET request:

```
GET http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionplan/1
```

Returns this response body:


```
{
    "links": [
        {
            "rel": "self",
            "href": "http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionplan/1"
        }
    ],
    "autoRenewal": false,
    "createdDate": "2020-1-1T00:00:00Z",
    "customForm": "-950",
    "defaultRenewalTerm": {
        "links": [],
        "id": "1",
        "refName": "Three Years"
    },
    "id": "1",
    "includeChildren": false,
    "incomeAccount": {
        "links": [
            {
                "rel": "self",
                "href": "http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/account/1"
            }
        ],
        "id": "1",
        "refName": "Sales"
    },
    "initialTerm": {
        "links": [],
        "id": "1",
        "refName": "Three Years"
    },
    "isInactive": false,
    "itemId": "DEFEND Package",
    "lastModifiedDate": "2020-1-1T00:00:00Z",
    "member": {
        "links": [
            {
                "rel": "self",
                "href": "http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionplan/1/member"
            }
        ]
    },
    "subsidiary": {
        "links": [
            {
                "rel": "self",
                "href": "http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionplan/1/subsidiary"
            }
        ]
    },
    "userNotes": {
        "links": [
            {
                "rel": "self",
                "href": "http://demo123.suitetalk.api.netsuite.com/services/rest/record/v1/subscriptionplan/1/userNotes"
            }
        ]
    }
```

```
60 }
```

Use Case For Managing Your Subscription Sales

You can use REST services to manage your subscription sales.

In this use case, you work through the process of using REST to sell a subscription plan for a fictitious company, CED Cyber Solutions.

This use case demonstrates how to:

1. Create Prerequisite Records
2. Create a Billing Account
3. Create a Draft Subscription
4. Edit Your Draft Subscription
5. Add an Add-On Item

Create Prerequisite Records

Before you begin, you need to create the following records:

- Subscription Plan – If you have not created a subscription plan (or its associated pricing records) yet and would like to do so using REST, see Use Case For Managing Your Subscription Catalog.
- Price Plan – See the help topic Creating Price Plans
- Price Book - See the help topic Creating Price Books
- Customer – You need a customer to sell your subscription plan to. In this example, your customer is called Clean Water Co.
- Billing Schedule – You need to create a charge-based billing schedule to determine how frequently we bill the customer. Your billing schedule is monthly, on the first of the month.

After creating these records, make note of their internal IDs. Throughout this example, use the IDs listed in the following table:

Record Type	Name	ID
Subscription Plan	DEFEND Package	1
Price Book	Standard Pricing	1
Customer	Clean Water Co.	1
Subscription	Three Years	1
Billing Schedule	Monthly, 1st of the month	1
Service Items	Managed Detection and Response	7

Create a Billing Account

Clean Water Co. wants to be billed on the first of the month. They use the same currency as your primary currency. Since they are starting their subscription on January 1, 2020, that is when billing starts.

The **startDate** must always be in the format YYYY-MM-DD. Use this format regardless of the date format preferences set.

A successful request returns a response with HTTP status **204 No Content**. The returned response headers contain the endpoint for the new record. Make note of the Billing Account ID. The example Billing Account internal ID is 1.

Example Billing Account

```
{
    "billingSchedule": { "id": "1" },
    "currency": { "id": "1" },
    "customer": { "id": "16" },
    "name": "REST Billing Account",
    "startDate": "2020-01-01"
}
```

Create a Draft Subscription

Your subscription references all records that you have created.

On a successful response, make note of the subscription ID. Use the ID to make changes to your draft subscription. The ID for your example subscription is 1.

Example Subscription

```
{
    "customer": { "id": "1" },
    "billingAccount": { "id": "1" },
    "subscriptionPlan": { "id": "1" },
    "priceBook": { "id": "1" },
    "initialTerm": { "id": "1" }
}
```

Edit Your Draft Subscription

When you created your subscription, you chose not to supply information about the line items included or the pricing of those line items. This means that the fields use the default values from the subscription plan.

If desired, you can change the defaults. To change the defaults, you send a PATCH request with the ID of the target record appended to the end of the patch request. The contents of the request body only need to include your changes. Make sure that all changes to dependent fields are included.

Clean Water Co. wants the silver support tier and has 25 users. Because they are one of your first customers, they receive a 50% discount on their setup costs.

Note: Leaving a field undefined leaves it unchanged. If you want to set the value of a field to an empty value, you must explicitly set the field to null in the request body.

A successful request returns a response with HTTP status **204 No Content**. The endpoint for the modified record is returned in the response headers.

Example Edited Subscription

```
{
```

```
    "subscriptionLine": {
        "items": [
            {
                "lineNumber": 4,
                "isIncluded": true
            }
        ]
    },
    "priceInterval": {
        "items": [
            {
                "subscriptionPlanLineNumber": 1,
                "discount": 50.0
            },
            {
                "subscriptionPlanLineNumber": 3,
                "quantity": 25
            }
        ]
    }
}
```

Add an Add-On Item

Before adding add-on items to any subscription plan, you must Enabling SuiteBilling Features.

Clean Water Co. would like to add Managed Detection and Response to their subscription. The requirements for the line are as follows:

- Bill in advance
- Always add to renewal subscriptions
- Have a prorated start and end date
- Charge $50.00 per month, prorated by day

Adding an add-on item to a draft subscription is an edit. You complete this edit by sending a PATCH request to the subscription endpoint with the ID appended to the end.

Because the add-on item is not tied to any subscription plan or price book, you must supply all of the subscription line and price book line information.

Because the add-on line requires new pricing information, you must create another price plan. If you are unfamiliar with this process, see Use Case For Managing Your Subscription Catalog. You assign an ID of 19 to the new price plan.

If the customer decides that they no longer want the add-on item, you can PATCH the subscription to mark the line as not included.

Example Add-On Items

```
{
    "subscriptionLine": {
        "items": [
            {
                "lineNumber": 7,
                "item": { "id": "7" },
                "subscriptionLineType": "2",
                "billingMode": "IN_ADVANCE",
                "renewalOption": "ALWAYS",
                "prorateStartDate": true,
                "prorateEndDate": true
            }
        ]
```

```
    },
    "priceInterval": {
        "items": [
            {
                "subscriptionPlanLineNumber": 7,
                "pricePlan": { "id": "19" },
                "frequency": "MONTHLY",
                "repeatEvery": "1",
                "startOffsetUnit": "MONTH",
                "startOffsetValue": 1,
                "prorateBy": "DAY"
            }
        ]
    }
}
```

Inventory Item Use Cases

This section includes the following sample use cases for inventory items:

- Use Case for Changing the Location of an Inventory Item
- Use Case for Adding a Vendor to an Inventory Item
- Use Case for Finding Items that are Assigned a Pricing Group
- Use Case for Deleting an Inventory Item

Inventory item records are used to track information about items for which you maintain a stock. For more information, read the help topic Inventory Items.

About This Tutorial

The following {{ VARIABLE }} expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
{{REST_SERVICES}}	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
{{*_ID}}	These variables refer to the internal ID of the record (for example, INVENTORY_ITEM_ID or SALES_ORDER_ID).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Changing the Location of an Inventory Item

To change the location of the inventory item, enter the internal ID of the new location in the body.

```
PATCH {{REST_SERVICES}}/record/v1/inventoryItem/{{INVENTORY_ITEM_ID}}

{
    "location" : 3
```

```
}
```

Use Case for Adding a Vendor to an Inventory Item

To add vendor information to the inventory item, send a PATCH request with the vendor ID and subsidiary ID in the body.

```
PATCH {{REST_SERVICES}}/record/v1/inventoryItem/{{INVENTORY_ITEM_ID}}
{
    "itemvendor" : {
        "items" : [{
            "vendor" : {"id" : "1316"},
            "subsidiary" : {"id" : "1"}
            }
        ]
    }
}
```

Use Case for Finding Items that are Assigned a Pricing Group

Use the GET command and query with internal ID of the pricing group to retrieve the list of items that are assigned a pricing group.

```
GET {{REST_SERVICES}}/record/v1/inventoryitem/?q=pricinggroup EQUAL 1
```

In the response body, you will see all the inventory item records that have the pricing group selected.

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryitem/?q=pricinggroup+EQUAL+1"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryitem/86"
                }
            ],
            "id": "86"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/inventoryitem/84"
                }
            ],
            "id": "84"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```

Use Case for Deleting an Inventory Item

To delete an inventory item, send the DELETE command to delete the record with the specified ID.

```
DELETE {{REST_SERVICES}}/record/v1/inventoryitem/{{INVENTORY_ITEM_ID}}
```

Customer Use Cases

This section includes the following sample use cases for customer record:

- Use Case for Finding Customers with Shipping Carrier Set to UPS
- Use Case for Updating Contact Details of a Customer
- Use Case for Creating a Customer from an External ID
- Use Case for Fetching a List of All Customers with Pagination
- Use Case for Creating an Invoice from a Customer

Customer records track information about your customers and enable you to view past transactions and communications with them. For more information, read the help topic Customers.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `CUSTOMER_ID` or `SALES_ORDER_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Finding Customers with Shipping Carrier Set to UPS

Use the GET command to retrieve the list of customers that have the shipping carrier set to UPS.

```
GET {{REST_SERVICES}}/record/v1/customer/?q=shippingCarrier IS "ups"
```

In the response body, you will see all the customer records that have the shipping carrier set to UPS.

```
{
    "links": [
```

```
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/?q=shippingCarrier+IS+%22ups%22"
        }
    ],
    "count": 3,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/315"
                }
            ],
            "id": "315"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/614"
                }
            ],
            "id": "614"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/1317"
                }
            ],
            "id": "1317"
        }
    ],
    "offset": 0,
    "totalResults": 3
}
```


Use Case for Updating Contact Details of a Customer

To update the contact details, enter the updated email and phone number in the body.

```
PATCH {{REST_SERVICES}}/record/v1/customer/{{CUSTOMER_ID}}/
{
    "email" : "john.doe@abc.com",
    "phone": "1234567890"
}
```

Use Case for Creating a Customer from an External ID

Use the PUT command to create a customer using an external ID.

```
PUT {{REST_SERVICES}}/record/v1/customer/eid:{{customerExternalId}}
{
   "companyName": "New Company External Id {{$timestamp}}",
   "email": "customer@example.com",
   "location" : "1"
}
```

For more information about using external IDs Using External IDs.

Use Case for Fetching a List of All Customers with Pagination

Add the limit and offset criteria to your request to get the results in multiple pages. For more information about pagination, read the help topic Collection Paging.

```
GET {{REST_SERVICES}}/record/v1/customer?q=dateCreated ON_OR_AFTER "{{thisYearStart}}" AND dateCreated BEFORE "{{thisYearEnd}}"&lim
it=3&offset=6
```

The following is an example of a response:

```
{
    "links": [
        {
            "rel": "previous",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer?limit=3&offset=3"
        },
        {
            "rel": "first",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer?limit=3&offset=0"
        },
        {
            "rel": "next",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer?limit=3&offset=9"
        },
        {
            "rel": "last",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer?limit=3&offset=12"
        },
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer?limit=3&offset=6&q=dateCre
ated+ON_OR_AFTER+%221%2F1%2F2022%22+AND+dateCreated+BEFORE+%221%2F1%2F2023%22"
        }
    ],
    "count": 3,
    "hasMore": true,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/514"
                }
            ],
            "id": "514"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/614"
                }
            ],
            "id": "614"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/716"
                }
            ],
            "id": "716"
        }
    ],
    "offset": 6,
    "totalResults": 14
```

```
}
```

Use Case for Creating an Invoice from a Customer

To create an invoice from a customer, include the values of required fields such as posting period, location, and item details in the request body.

```
POST {{REST_SERVICES}}/record/v1/customer/{{CUSTOMER_ID}}/!transform/invoice
{
    "postingperiod" : "15",
    "location" : "1",
    "item": {
    "items": [
         {
            "item": { "id": "86" },
            "amount": 10
         }
      ]
   }
}
```

Employee Use Cases

This section includes the following sample use cases for employee record:

- Use Case for Adding a Supervisor to an Employee
- Use Case for Adding Employee Address
- Use Case for Adding Subscriptions to an Employee
- Use Case for Creating an Expense Report from an Employee Record
- Use Case for Finding All Employee Records Under a Supervisor
- Use Case for Retrieving Permissions Assigned to a Role
- Use Case for Finding IDs of Roles Assigned to an Employee
- Use Case for Finding the Name of a Role

Employee records represent your employees. Use the employee record to store information for contact, login, payroll, and human resources purposes. For more information, see the help topic Employee Management.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `CUSTOMER_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Adding a Supervisor to an Employee

To add or update the supervisor of the employee, enter the internal ID of the supervisor in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/employee/<id>/
{
    "supervisor": "13"
}
```

Use Case for Adding Employee Address

To add employee address, send a PATCH request with the address details in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/employee/<id>/
{
    "addressbook": {
        "items": [
            {
                "label": "New York HQ",
                "addressbookaddress": {
                    "country": "US",
                    "state": "NY",
                    "zip": "10001",
                    "addressee": "New company"
                }
            }
        ]
    }
}
```

Use Case for Adding Subscriptions to an Employee

To add subscriptions to the employee, use the PATCH command and add the subscription details in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/employee/1720/
{
    "subscriptions": {
        "items": [
            {
                "subscription": {
                    "id": "1"
                },
                "subscribed": true
            }
        ]
    }
}
```

Use Case for Creating an Expense Report from an Employee Record

To create an expense report from an employee record, include the values of required fields such as expense account, curency, and amount in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/employee/<id>/!transform/ExpenseReport
{
    "expense": {
        "items": [
            {
                "expenseaccount": "58",
                "currency": "1",
                "amount": 100
            }
        ]
    }
}
```

Use Case for Finding All Employee Records Under a Supervisor

Use the GET command to fetch all employee records under a supervisor.

```
GET {{REST_SERVICES}}/record/v1/employee/?q=supervisor EQUAL 13
```

In the response body, you will see all the employee records under the supervisor.

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/employee/?q=supervisor+EQUAL+13"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/employee/1720"
                }
            ],
            "id": "1720"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/employee/1519"
                }
            ],
            "id": "1519"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```

Use Case for Retrieving Permissions Assigned to a Role

To get permissions assigned to a role, send a SuiteQL query with the role id. Add the limit and offset criteria to your request to get the results in multiple pages. For more information about pagination, read the help topic Collection Paging.

Note that `Prefer: transient` is a required header parameter.

```
POST {{REST_SERVICES}}/query/v1/suiteql?limit=5&offset=10
{
    "q": "SELECT * FROM rolepermissions WHERE role = 1028"
}
```

The following is an example of a response:

```
{
    "links": [
        {
            "rel": "previous",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5&offset=5"
        },
        {
            "rel": "first",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5&offset=0"
        },
        {
            "rel": "next",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5&offset=15"
        },
        {
            "rel": "last",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5&offset=60"
        },
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql?limit=5&offset=10"
        }
    ],
    "count": 5,
    "hasMore": true,
    "items": [
        {
            "links": [],
            "name": "Custom Fields",
            "permkey": "ADMI_CUSTFIELD",
            "permlevel": "4",
            "restriction": "-1",
            "role": "1028"
        },
        {
            "links": [],
            "name": "Custom Item Fields",
            "permkey": "ADMI_CUSTITEMFIELD",
            "permlevel": "4",
            "restriction": "-1",
            "role": "1028"
        },
        {
            "links": [],
            "name": "Custom Lists",
            "permkey": "ADMI_CUSTLIST",
            "permlevel": "4",
            "restriction": "-1",
            "role": "1028"
        },
        {
            "links": [],
            "name": "Custom PDF Layouts",
            "permkey": "ADMI_CUSTLAYOUT",
            "permlevel": "4",
            "restriction": "-1",
            "role": "1028"
        },
        {
            "links": [],
            "name": "Custom Record Types",
            "permkey": "ADMI_CUSTRECORD",
```

```
            "permlevel": "4",
            "restriction": "-1",
            "role": "1028"
        }
    ],
    "offset": 10,
    "totalResults": 61
}
```

Use Case for Finding IDs of Roles Assigned to an Employee

To find IDs of roles assigned to an employee, send a SuiteQL query with the employee ID. Note that `Prefer: transient` is a required header parameter.

```
POST {{REST_SERVICES}}/query/v1/suiteql?
{
    "q": "SELECT rolesforsearch FROM employee WHERE id = 13"
}
```

The following is an example of the response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql"
        }
    ],
    "count": 1,
    "hasMore": false,
    "items": [
        {
            "links": [],
            "rolesforsearch": "1, 3"
        }
    ],
    "offset": 0,
    "totalResults": 1
}
```

Use Case for Finding the Name of a Role

To find the name of a role, send a SuiteQL query with the role ID. Note that `Prefer: transient` is a required header parameter.

```
POST {{REST_SERVICES}}/query/v1/suiteql?
{
    "q": "SELECT name FROM role where id = 7"
}
```

The following is an example of the response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql"
        }
    ],
    "count": 1,
```

```
    "hasMore": false,
    "items": [
        {
            "links": [],
            "name": "A/P Clerk"
        }
    ],
    "offset": 0,
    "totalResults": 1
}
```

Invoice Use Cases

This section includes the following sample use cases for invoice record:

- Use Case for Creating an Invoice
- Use Cases for Updating an Invoice
- Use Case or Retrieving an Invoice
- Use Case for Fetching Invoices from a Lead Source
- Use Case for Deleting an Invoice

An invoice is a record of a sale to a customer. Invoicing is the process of creating bills for goods and services that customers receive. For more information, see the help topic Invoices.

About this Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `INVOICE_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating an Invoice

To create an invoice with an item, include the values of required fields such as posting period, location, item ID, and amount in the request body.

```
POST {{REST_SERVICES}}/record/v1/invoice/
{
    "entity": {
        "id": "614"
    },
    "postingperiod": "21",
    "location": 1,
    "item": {
```

```
        "items": [
            {
                "amount": 1000.0,
                "item": {
                    "id": "86"
                }
            }
        ]
    },
    "subsidiary": {
        "id": "1"
    },
    "terms": {
        "id": "1"
    }
}
```

For more options to create an invoice using REST API, see Invoice.

Use Cases for Updating an Invoice

Use the PATCH command to add a line item to an invoice:

```
PATCH {{REST_SERVICES}}/record/v1/invoice/<id>/
{
    "item": {
        "items": [
            {
                "item": {
                    "id": "13"
                },
                "quantity": 50,
                "amount": 100
            }
        ]
    }
}
```

To update a line item in an invoice, add the line item ID in the request.

```
PATCH {{REST_SERVICES}}/record/v1/invoice/<id>/
{
    "item": {
        "items": [
            {
                "line": 5,
                "item": {
                    "id": "10"
                },
                "quantity": 75,
                "amount": 90
            }
        ]
    }
}
```

Use Case or Retrieving an Invoice

Use the GET command to retrieve an invoice.

```
GET {{REST_SERVICES}}/record/v1/invoice/1215/
```

In the response body, you will get details of the requested invoice.


```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1215/"
        }
    ],
    "amountPaid": 0.0,
    "amountRemaining": 1000.0,
    "amountRemainingTotalBox": 1000.0,
    "billAddress": "Dwight Schrute\nNew York NY 10001\nUnited States",
    "billAddressList": {
        "links": [],
        "id": "401",
        "refName": "New York HQ"
    },
    "billingAddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1215/billingAddress"
            }
        ]
    },
    "billingAddress_text": "Dwight Schrute\nNew York NY 10001\nUnited States",
    "createdDate": "2023-02-06T09:48:00Z",
    "currency": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/currency/1"
            }
        ],
        "id": "1",
        "refName": "USA"
    },
    "custbody_atlas_exist_cust_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_cust_type/2"
            }
        ],
        "id": "2",
        "refName": "Existing Customer"
    },
    "custbody_atlas_new_cust_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_cust_type/1"
            }
        ],
        "id": "1",
        "refName": "New Customer"
    },
    "custbody_atlas_no_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_appr_by_creator/2"
            }
        ],
        "id": "2",
        "refName": "No"
    },
    "custbody_atlas_yes_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_appr_by_creator/1"
```

```
            }
        ],
        "id": "1",
        "refName": "Yes"
    },
    "custbody_esc_created_date": "2023-02-06",
    "custbody_esc_last_modified_date": "2023-02-06",
    "customForm": {
        "id": "91",
        "refName": "Standard Service Invoice"
    },
    "dueDate": "2023-02-21",
    "email": "another.customer@example.com",
    "entity": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/614"
            }
        ],
        "id": "614",
        "refName": "8 Another Company 1661496973"
    },
    "estGrossProfit": 1000.0,
    "estGrossProfitPercent": 100.0,
    "exchangeRate": 1.0,
    "id": "1215",
    "item": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1215/item"
            }
        ]
    },
    "lastModifiedDate": "2023-02-06T09:48:00Z",
    "location": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/location/1"
            }
        ],
        "id": "1",
        "refName": "California"
    },
    "originator": "restWebServices",
    "postingPeriod": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/accountingperiod/21"
            }
        ],
        "id": "21",
        "refName": "Feb 2023"
    },
    "prevDate": "2023-02-06",
    "salesEffectiveDate": "2023-02-06",
    "shipAddress": "Dwight Schrute\nNew York NY 10001\nUnited States",
    "shipAddressList": {
        "links": [],
        "id": "401",
        "refName": "New York HQ"
    },
    "shipDate": "2023-02-06",
    "shipIsResidential": false,
    "shipOverride": false,
    "shippingAddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1215/shippingAddress"
```

```
            }
        ]
    },
    "shippingAddress_text": "Dwight Schrute\nNew York NY 10001\nUnited States",
    "source": {
        "id": "REST Web Services",
        "refName": "REST Web Services"
    },
    "status": {
        "id": "Open",
        "refName": "Open"
    },
    "subsidiary": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/subsidiary/1"
            }
        ],
        "id": "1",
        "refName": "Parent Company"
    },
    "subtotal": 1000.0,
    "terms": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/term/1"
            }
        ],
        "id": "1",
        "refName": "Net 15"
    },
    "toBeEmailed": false,
    "toBeFaxed": false,
    "toBePrinted": false,
    "total": 1000.0,
    "totalCostEstimate": 0.0,
    "tranDate": "2023-02-06",
    "tranId": "INV07"
}
```


Use Case for Fetching Invoices from a Lead Source

To fetch invoices from a lead source, query the lead source's ID in the request.

```
GET {{REST_SERVICES}}/record/v1/invoice/?q=leadsource EQUAL "-4"
```

In the response body, you will get all the invoices from the requested lead source.


```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/?q=leadsource+EQUAL+%22-4%22"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1109"
                }
            ],
            "id": "1109"
        },
```

```
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/invoice/1115"
                }
            ],
            "id": "1115"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```

Use Case for Deleting an Invoice

To delete an invoice, send the DELETE command with the invoice ID.

```
DELETE {{REST_SERVICES}}/record/v1/invoice/<id>/
```

Journal Entry Use Cases

This section includes the following sample use cases for journal entry record:

- Use Case for Creating a Journal Entry
- Use Cases for Approving a Journal Entry
- Use Case for Creating a Reverse Journal Entry
- Use Case for Updating Line Items in a Journal Entry
- Use Case for Retrieving Journal Entries Created After a Date

You use the journal entry record to adjust balances in accounts. Journal entries let you change the value of any set of accounts without having to enter a posting transaction. For more information, read the help topic Journal Entries.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `JOURNAL_ENTRY_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating a Journal Entry

To create a journal entry, include the values of required fields such as subsidiary, account, and debit and credit amounts in the request body. Make sure the debits and credits in the journal entry balance.

```
POST {{REST_SERVICES}}/record/v1/journalentry/
{
    "subsidiary": {
        "id": "1"
    },
    "line": {
        "items": [
            {
                "account": {
                    "id": "10"
                },
                "debit": 300
            },
            {
                "account": {
                    "id": "14"
                },
                "credit": 300
            }
        ]
    }
}
```

Use Cases for Approving a Journal Entry

To approve a journal entry, send a PATCH command in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/journalentry/<id>/
{
    "approved": true
}
```

Resetting Approval Status

If you already approved a journal entry but need to reset the approval status, send a PATCH command with the updated status in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/journalentry/<id>/
{
    "approved": false
}
```

Use Case for Creating a Reverse Journal Entry

To create a reverse journal entry, send a PATCH command with the reversal date in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/journalentry/<id>/
{
    "reversaldate": "2023-2-22"
}
```

Use Case for Updating Line Items in a Journal Entry

To add line items to a journal entry, include the values of required fields such as account, debit and credit amounts in the request body. Make sure the debits and credits in the journal entry balance.

```
PATCH {{REST_SERVICES}}/record/v1/journalentry/<id>/
```

```
{
    "line": {
        "items": [
            {
                "account": {
                    "id": "15"
                },
                "debit": 250
            },
            {
                "account": {
                    "id": "17"
                },
                "credit": 250
            }
        ]
    }
}
```

To update an existing line item, mention the line number in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/journalentry/<id>/
{
    "line": {
        "items": [
            {
                "line": 2,
                "account": {
                    "id": "15"
                },
                "debit": 1150
            },
            {
                "line": 3,
                "account": {
                    "id": "17"
                },
                "credit": 1150
            }
        ]
    }
}
```

Use Case for Retrieving Journal Entries Created After a Date

To retrieve journal entries created after a date, enter the created date in the request body.

```
{{REST_SERVICES}}/record/v1/journalentry/?q=createddate AFTER "2/14/23"
```

Following is a sample response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/?q=createddate+AFTER+
%222%2F14%2F23%22"
        }
    ],
    "count": 7,
    "hasMore": false,
    "items": [
        {
```

```
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1415"
                }
            ],
            "id": "1415"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1416"
                }
            ],
            "id": "1416"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1516"
                }
            ],
            "id": "1516"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1417"
                }
            ],
            "id": "1417"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1517"
                }
            ],
            "id": "1517"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1518"
                }
            ],
            "id": "1518"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/journalentry/1618"
                }
            ],
            "id": "1618"
        }
    ],
    "offset": 0,
    "totalResults": 7
}
```


Event Use Cases

This section includes the following sample use cases for event record:

You use event records to reserve time on your calendar for appointments and meetings. For more information, read the help topic Working with Events.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `CALENDAREVENT_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating an Event

To create an event, include values for fields such as title, status, and start date in the request body.

```
POST {{REST_SERVICES}}/record/v1/calendarevent/
{
    "title": "Meeting with Tom",
    "status": "CONFIRMED",
    "message": "Let's discuss work",
    "location": "Caffe West",
    "startdate": "2023-11-04"
}
```

Use Case for Adding an Attendee to an Event

Use the PATCH command to add an attendee to an event.

```
PATCH {{REST_SERVICES}}/record/v1/calendarevent/<id>/
{
    "attendee": {
        "items": [
            {
                "attendee": {
                    "id": "5"
                },
                "response": "ACCEPTED"
            }
```

```
        ]
    }
}
```

Use Case for Getting a List of Events with Confirmed Status and that are Scheduled After a Date

Use the GET command and query with status and start date to retrieve the list of events.

```
GET {{REST_SERVICES}}/record/v1/calendarevent/?q=status IS "CONFIRMED"&q=startdate AFTER "3/16/23"
```

The following is an example of the response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/?q=startdate+AFTER+%223%2F16%2F23%22"
        }
    ],
    "count": 5,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/103"
                }
            ],
            "id": "103"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/2"
                }
            ],
            "id": "2"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/203"
                }
            ],
            "id": "203"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/4"
                }
            ],
            "id": "4"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/calendarevent/3"
```

```
                }
            ],
            "id": "3"
        }
    ],
    "offset": 0,
    "totalResults": 5
}
```

Use Case for Fetching Details of Events at a Location

To get details of events at a location, send a SuiteQL query with the location. Note that `Prefer: transient` is a required header parameter.

```
{{REST_SERVICES}}/query/v1/suiteql?
{
    "q": "SELECT title, startdate,  FROM calendarevent WHERE location = 'Caffe West'"
}
```

The following is an example of the response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql"
        }
    ],
    "count": 4,
    "hasMore": false,
    "items": [
        {
            "links": [],
            "startdate": "11/4/2023",
            "title": "All hands"
        },
        {
            "links": [],
            "startdate": "11/3/2023",
            "title": "Customer Engagement"
        },
        {
            "links": [],
            "startdate": "11/2/2023",
            "title": "Meeting with Tom"
        },
        {
            "links": [],
            "startdate": "11/6/2023",
            "title": "Private Appointment"
        }
    ],
    "offset": 0,
    "totalResults": 4
}
```

Use Case for Deleting an Event

To delete an event, send the DELETE command with the event ID.

```
DELETE {{REST_SERVICES}}/record/v1/calendarevent/<id>/
```

Credit Memo Use Cases

This section includes the following sample use cases for credit memo record:

- Use Case for Creating a Credit Memo
- Use Case for Adding Line Items to a Credit Memo
- Use Case for Fetching List of Credit Memos Created After a Date
- Use Case for Fetching Credit Memos from a Lead Source
- Use Case for Getting Details of a Credit Memo

A credit memo is a transaction that decreases the amount a customer owes you. You can use a credit memo to reverse a charge you billed to a customer. For more information, read the help topic Customer Credit Memos.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `CREDITMEMO_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating a Credit Memo

To create a credit memo, include values for fields such as customer, location, and item details in the request body.

```
POST {{REST_SERVICES}}/record/v1/creditmemo/

{
    "entity": {
        "id": "14"
    },
    "item": {
        "items": [
            {
                "amount": 10.0,
                "item": {
                    "id": "13"
                }
            }
        ]
    },
    "location": "1"
}
```

Use Case for Adding Line Items to a Credit Memo

Use the PATCH command to add line items to a credit memo.

```
PATCH {{REST_SERVICES}}/record/v1/creditmemo/<id>/

{
    "item": {
        "items": [
            {
                "item": {
                    "id": 10
                },
                "amount": 100,
                "quantity": 25
            }
        ]
    }
}
```

Use Case for Fetching List of Credit Memos Created After a Date

Use the GET command and query with the createddate field to get a list of credit memos after the specified date.

```
GET {{REST_SERVICES}}/record/v1/creditmemo/?q=createddate AFTER "03/14/2023"
```

In response you will get the list of credit memos, something like this:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/?q=createddate+AFTER+
%2203%2F14%2F2023%22"
        }
    ],
    "count": 3,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1818"
                }
            ],
            "id": "1818"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1819"
                }
            ],
            "id": "1819"
        },
        {
            "links": [
                {
```

```
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1919"
                }
            ],
            "id": "1919"
        }
    ],
    "offset": 0,
    "totalResults": 3
}
```

Use Case for Fetching Credit Memos from a Lead Source

To fetch credit memos from a lead source, query the lead source's ID in the request.

```
GET {{REST_SERVICES}}/record/v1/creditmemo/?q=leadsource EQUAL "-2"
```

In the response body, you will get all the credit memos from the requested lead source, something like this:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/?q=leadsource+EQUAL+%22-2%22"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1818"
                }
            ],
            "id": "1818"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1919"
                }
            ],
            "id": "1919"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```

Use Case for Getting Details of a Credit Memo

Use the GET command and query with internal ID of the credit memo to retrieve the details. In the following example, the internal ID of the credit memo is 1819.

```
GET {{REST_SERVICES}}/record/v1/creditmemo/1819/
```

The response will look something like this:


```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1819/"
        }
    ],
    "amountPaid": 0.0,
    "amountRemaining": 10.0,
    "applied": 0.0,
    "billingAddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1819/billingAddress"
            }
        ]
    },
    "createdDate": "2023-03-30T10:14:00Z",
    "currency": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/currency/1"
            }
        ],
        "id": "1",
        "refName": "USA"
    },
    "custbody_atlas_exist_cust_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_cust_type/2"
            }
        ],
        "id": "2",
        "refName": "Existing Customer"
    },
    "custbody_atlas_new_cust_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_cust_type/1"
            }
        ],
        "id": "1",
        "refName": "New Customer"
    },
    "custbody_atlas_no_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_appr_by_creator/2"
            }
        ],
        "id": "2",
        "refName": "No"
    },
    "custbody_atlas_yes_hdn": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customlist_atlas_appr_by_creator/1"
            }
        ],
        "id": "1",
        "refName": "Yes"
    },
    "custbody_esc_created_date": "2023-03-30",
    "custbody_esc_last_modified_date": "2023-03-31",
```

```
    "customForm": {
        "id": "94",
        "refName": "Standard Credit Memo"
    },
    "email": "customer@example.com",
    "entity": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/customer/14"
            }
        ],
        "id": "14",
        "refName": "1 Company 1660029525"
    },
    "estGrossProfit": 0.0,
    "estGrossProfitPercent": 0.0,
    "exchangeRate": 1.0,
    "id": "1819",
    "item": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1819/item"
            }
        ]
    },
    "lastModifiedDate": "2023-03-31T08:57:00Z",
    "location": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/location/1"
            }
        ],
        "id": "1",
        "refName": "California"
    },
    "originator": "restWebServices",
    "postingPeriod": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/accountingperiod/22"
            }
        ],
        "id": "22",
        "refName": "Mar 2023"
    },
    "prevDate": "2023-03-30",
    "salesEffectiveDate": "2023-03-30",
    "shipIsResidential": false,
    "shipOverride": false,
    "shippingAddress": {
        "links": [
            {
                "rel": "self",
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/creditmemo/1819/shippingAddress"
            }
        ]
    },
    "source": {
        "id": "REST Web Services",
        "refName": "REST Web Services"
    },
    "status": {
        "id": "Open",
        "refName": "Open"
    },
    "subsidiary": {
        "links": [
            {
                "rel": "self",
```

```
                "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/subsidiary/1"
            }
        ],
        "id": "1",
        "refName": "Parent Company"
    },
    "subtotal": 10.0,
    "toBeEmailed": false,
    "toBeFaxed": false,
    "toBePrinted": false,
    "total": 10.0,
    "totalCostEstimate": 10.0,
    "tranDate": "2023-03-30",
    "tranId": "CM02",
    "unapplied": 10.0
}
```

Vendor Use Cases

This section includes the following sample use cases for vendor record:

- Use Case for Creating a Vendor
- Use Case for Creating a Tax Agency Vendor
- Use Case for Providing Role Access to a Vendor
- Use Case for Inactivating a Vendor
- Use Case for Setting Credit Limit for a Vendor
- Use Case for Defining the Preferred Transaction Delivery Methods for a Vendor
- Use Case for Finding Vendors that are Assigned an Expense Account
- Use Case for Getting Details of Vendors that are Assigned a Category
- Use Case for Getting List of Vendors that have Credit Limits Above a Value

A vendor is a company or person you purchase goods and services from. Vendor records track information about your vendors and enable you to view past transactions and communications with them. For information about working with vendor records in the UI, see the help topic Vendor Records Overview.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `VENDOR_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating a Vendor

To create a vendor, include values for fields such as company name in the request body.

```
POST {{REST_SERVICES}}/record/v1/vendor/
{
    "companyName": "NZ Vendor"
}
```

Use Case for Creating a Tax Agency Vendor

To create a tax agency vendor, include the id of tax agency in the category field.

```
POST {{REST_SERVICES}}/record/v1/vendor/
{
    "companyName": "BZ Vendor",
    "category": "3"
}
```

Use Case for Providing Role Access to a Vendor

To give a role access to a vendor, enable the **Give Access** box and add the role details in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/vendor/<id>/
{
    "giveaccess": true,
    "roles": {
        "items": [
            {
                "selectedRole": {
                    "id": "6"
                }
            }
        ]
    }
}
```

Use Case for Inactivating a Vendor

Use the PATCH command and enable the **Inactive** box in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/vendor/<id>/
{
    "isinactive": true
}
```

Use Case for Setting Credit Limit for a Vendor

Use the PATCH command and set the credit limit in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/vendor/<id>/
{
    "creditlimit": 10000
}
```

Use Case for Defining the Preferred Transaction Delivery Methods for a Vendor

Use the PATCH command and enable the desired preferred transaction delivery methods in the request body. In the following example, email and print transactions are enabled.

```
PATCH {{REST_SERVICES}}/record/v1/vendor/<id>/
{
    "emailtransactions": true,
    "printtransactions": true
}
```

Use Case for Finding Vendors that are Assigned an Expense Account

Use the GET command and query with the id of the expense account to get the list of vendors that are assigned an expense account.

```
GET {{REST_SERVICES}}/record/v1/vendor/?q=expenseaccount EQUAL "10"
```

Following is a sample response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendor/?q=expenseaccount+EQUAL+%2210%22"
        }
    ],
    "count": 3,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendor/1723"
                }
            ],
            "id": "1723"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendor/1722"
                }
            ],
            "id": "1722"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendor/1823"
                }
            ],
            "id": "1823"
        }
    ],
    "offset": 0,
```

```
    "totalResults": 3
}
```

Use Case for Getting Details of Vendors that are Assigned a Category

To get details of vendors that are assigned a category, send a SuiteQL query with the category. Note that `Prefer: transient` is a required header parameter.

```
POST {{REST_SERVICES}}/query/v1/suiteql?
{
    "q": "SELECT id, legalname, expenseaccount FROM vendor WHERE category = 3"
}
```

Following is a sample response:

```
{
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [],
            "expenseaccount": "12",
            "id": "1725",
            "legalname": "BZ Vendor"
        },
        {
            "links": [],
            "expenseaccount": "10",
            "id": "1823",
            "legalname": "Test State Tax Agency"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```

Use Case for Getting List of Vendors that have Credit Limits Above a Value

To get vendors that have a credit limit above a certain value, send a SuiteQL query with the credit limit. Note that `Prefer: transient` is a required header parameter.

```
POST {{REST_SERVICES}}/query/v1/suiteql?
{
    "q": "SELECT id, entityid AS 'vendor name', creditlimit FROM vendor WHERE creditlimit > 5000"
}
```

Following is a sample response:

```
{
    "links": [
```

```
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/query/v1/suiteql"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [],
            "creditlimit": "8000",
            "id": "1725",
            "vendor name": "BZ Vendor"
        },
        {
            "links": [],
            "creditlimit": "10000",
            "id": "1722",
            "vendor name": "Test"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```


Vendor Bill Use Cases

This section includes the following sample use cases for vendor bill record:

- Use Case for Creating a Vendor Bill
- Use Case for Approving a Vendor Bill
- Use Case for Adding Line Items to a Vendor Bill
- Use Case for Getting a List of Bills from a Vendor

Vendor bills are records which you can use to track payables as they arrive from vendors. For information about working with vendor bills in the UI, see the help topic Vendor Bills.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
{{REST_SERVICES}}	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
{{*_ID}}	These variables refer to the internal ID of the record (for example, `VENDORBILL_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating a Vendor Bill

To create a vendor bill, include values for fields such as vendor and item details in the request body.

```
POST {{REST_SERVICES}}/record/v1/vendorbill
{
    "entity": {
        "id": "1722"
    },
    "item": {
        "items": [
            {
                "item": {
                    "id": 84
                },
                "quantity": 25,
                "rate": 100
            }
        ]
    }
}
```

Use Case for Approving a Vendor Bill

To approve a vendor bill, use the PATCH command and include the approval status in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/vendorbill/<id>
{
    "approvalstatus": 2
}
```

Use Case for Adding Line Items to a Vendor Bill

To add line items to a vendor bill, use the PATCH command and include the item details in the request body.

```
PATCH {{REST_SERVICES}}/record/v1/vendorbill/<id>
{
    "item": {
        "items": [
            {
                "item": {
                    "id": 84
                },
                "quantity": 30,
                "rate": 50
            }
        ]
    }
}
```

Use Case for Getting a List of Bills from a Vendor

Use the GET command and query with the vendor id to get the list of bills from a vendor.

```
GET {{REST_SERVICES}}/record/v1/vendorbill/?q=entity EQUAL 1722
```

Following is an example of the response:

```
{
```

```
    "links": [
        {
            "rel": "self",
            "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendorbill/?q=entity+EQUAL+%221722%22"
        }
    ],
    "count": 2,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendorbill/1920"
                }
            ],
            "id": "1920"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://<accountID>.suitetalk.api.netsuite.com/services/rest/record/v1/vendorbill/2020"
                }
            ],
            "id": "2020"
        }
    ],
    "offset": 0,
    "totalResults": 2
}
```


Cash Sale Use Cases

This section includes the following sample use cases for cash sale record:

- Use Case for Creating a Cash Sale
- Use Case for Updating a Cash Sale
- Use Case for Issuing a Return Authorization from a Cash Sale
- Use Case for Fetching List of Cash Sales for a Customer

A cash sale is a transaction that records the sale of goods or services for which you receive immediate payment. For more information, read the help topic Cash Sales.

About This Tutorial

The following `{{ VARIABLE }}` expressions are variables that you need to customize when reusing the code samples.

Variable	Definition
`{{REST_SERVICES}}`	Indicates your access to the NetSuite REST Service (for example, `https://testaccount.corp.netsuite.com/services/rest`).
`{{*_ID}}`	These variables refer to the internal ID of the record (for example, `CASHSALE_ID` or `EMPLOYEE_ID`).

The IDs and the variables used in the following scenarios are examples only, and are included to show how to write REST commands.

Use Case for Creating a Cash Sale

To create a cash sale, include values for fields such as customer ID, location, and item details in the request body.

```
POST {{REST_SERVICES}}/record/v1/cashsale
{
    "entity": {
        "id": "14"
    },
    "location": "1",
    "item": {
        "items": [
            {
                "item": {
                    "id": "13"
                },
                "quantity": 25,
                "amount": 275
            }
        ]
    }
}
```

Use Case for Updating a Cash Sale

Use the PATCH command to update a cash sale.

```
PATCH {{REST_SERVICES}}/record/v1/cashsale/<id>
{
    "item": {
        "items": [
            {
                "item": {
                    "id": "14"
                },
                "quantity": 50,
                "amount": 300
            }
        ]
    }
}
```

Use Case for Issuing a Return Authorization from a Cash Sale

To create a return authorization from a cash sale, use the transform function in the REST API:

```
POST {{REST_SERVICES}}/record/v1/cashsale/<id>/!transform/returnauthorization
```

Use Case for Fetching List of Cash Sales for a Customer

Use the GET command and query with the customer id to get the list of cash sales from a customer.

```
GET {{REST_SERVICES}}/record/v1/cashsale/?q=entity EQUAL "14"
```

Following is an example of the response:


```
{
    "links": [
        {
            "rel": "self",
            "href": "https://5629979.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale/?q=entity+EQUAL+%2214%22"
        }
    ],
    "count": 3,
    "hasMore": false,
    "items": [
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://5629979.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale/2121"
                }
            ],
            "id": "2121"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://5629979.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale/2122"
                }
            ],
            "id": "2122"
        },
        {
            "links": [
                {
                    "rel": "self",
                    "href": "https://5629979.suitetalk.api.netsuite.com/services/rest/record/v1/cashsale/2123"
                }
            ],
            "id": "2123"
        }
    ],
    "offset": 0,
    "totalResults": 3
}
```