Oracle Fusion Cloud Student Financial Aid

Student Financial Aid Questions and Answers

Oracle Fusion Cloud Student Financial Aid
Student Financial Aid Questions and Answers

G18290-06

Copyright © 2025, Oracle and/or its affiliates.

Author: Higher Education Information Development Team

Contents

Get Help

There are a number of ways to learn more about your product and interact with Oracle and other users.

Get Help in the Applications

Some application pages have help icons to give you access to contextual help. If you don't see any help icons on your page, click your user image or name in the global header and select Show Help Icons. If the page has contextual help, help icons will appear.

Get Support

You can get support at *My Oracle Support*. For accessible support, visit *Oracle Accessibility Learning and Support*.

Get Training

Increase your knowledge of Oracle Cloud by taking courses at *Oracle University*.

Join Our Community

Use *Cloud Customer Connect* to get information from industry experts at Oracle and in the partner community. You can join forums to connect with other customers, post questions, suggest *ideas* for product enhancements, and watch events.

Learn About Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program*. Videos included in this guide are provided as a media alternative for text-based topics also available in this guide.

Share Your Feedback

We welcome your feedback about Oracle Applications user assistance. If you need clarification, find an error, or just want to tell us what you found helpful, we'd like to hear from you.

You can email your feedback to *oracle_fusion_applications_help_ww_grp@oracle.com*.

Thanks for helping us improve our user assistance!

1 Student Financial Aid Questions and Answers

Can I request an NSLDS file from the US Department of Education (ED) from a Test SFP environment?

(SFP only) No, the test environment doesn't communicate with ED via the SAIG mailbox.

How can I manage groups in Student Self-Service?

Administrators with Manage Self Service Permissions access can manage groups in the Roles Management section of the Settings interface.

You can control what type of access a group of users have, and to which specific permissions and roles they have access to. This lets your users to have one access point to all production and test environments for the Financial Aid Administration user interface and Student/Parent Self-Service portal. This centralized approach provides enhanced security and simplifies the enforcement of credentials for your end users.

Related Topics

- Manage Users and Groups in OCI IAM

Add a Group

1. Navigate to **Settings > Role Permissions**.
2. Click **Add Group** under the Details tab.
3. Provide a name for the group.
4. Provide a brief description of the group of its purpose.
5. Click **Save**.

The Groups section displays all groups defined across all your Student Financial Aid (SFA) environments (Test vs Production) and SFA applications (Administration user interface and Self-Service portal).

After a group is created, you can’t delete it from an SFA application. You will need to do this from the Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) integration only.

Assign Groups to Roles

You can assign roles to multiple groups. Also, you can assign groups to multiple roles. Users in a group with multiple roles get all the permissions from each role combined.

If you are trying to log in, but you are part of a different group or environment than expected, you will see an error message. To avoid this, it's a good practice to name your groups clearly, including information about the environment and application they are related to.

How can I manage users in Student Self-Service?

From Manage Users, administrators can search for, view, edit, and remove users.

Search for a User

1. Click the drop-down arrow in the tool bar.
2. Click **Manage Users**.
3. Enter external student ID, user name, first name, last name, email, or phone number to search for a user.
4. Click **Search**.
5. Select a user from the results to view.

View a User

1. Click the drop-down arrow in the tool bar.
2. Click **Manage Users**.
3. View the user type that denotes whether the user is a student, parent or an administrator.

Edit a User

1. Click the drop-down list in the tool bar.
2. Click **Manage Users**.
3. Select one of the users and click the **Edit** Icon that lets you modify user accounts created in the Admin UI before IAM integration.

   For all other user accounts, you can view when the user last logged in and how they logged in, you can hover over the link icon within the Edit column.
   a. External system linked user (OCI SAML): User logged in using OCI Identity and Access Management Integration
   b. External system linked user (SAML): User logged in using your External IDP not via IAM Integration
4. Edit the user information such as External Student ID, Username, Name, Email, Phone Number, or Password.
5. Click **Save**.

Remove a User

1. Click the drop-down arrow in the tool bar.

2. Click **Manage Users**.
3. Select one of the users and click the **Remove** icon.

   For users logged in via SSO (External system linked user (OCI SAML or External system linked user (SAML), this will only be a soft delete. This will not delete the user account from OCI IAM, but the user account will not be displayed in the Manage User interface within Self-Service (Portal) unless they log in again.

Create, Edit, and Disable SSO Users

New user accounts and updates to existing user accounts are managed via the OCI Cloud Console. See *Manage Users and Groups in OCI IAM* for more information.

When you delete a user account in the OCI (Oracle Cloud Infrastructure) Cloud Console, the user account isn’t permanently erased from the application. Instead, the user account record is retained in the User Management interface for audit and compliance purposes.

Automatic User Log Off

The standard configuration for user sessions in the Student Self-Service application includes a security feature that automatically logs users out after 20 minutes of inactivity, ensuring data privacy and protection. If your users require a different inactivity timeout period, you can configure this setting by submitting a service request to the support team.

This modification enhances user experience, allowing for more flexibility and convenience while maintaining the security of student sensitive information.

How do I assign global or local logout permissions?

The global sign out feature improves security by terminating all your active sessions and reduces the risk of sharing sensitive data unknowingly.

By default, local sign out permission is enabled for the students, guest (parents), and administrator role categories. You need to assign the global sign out permission to the role categories.

On the Administration UI, the permission is labeled **Enable Global Logout**, and on the Student Self-service portal its labeled **Logout Globally**.

Navigation:

- Student Self-service portal: **Settings** > **Role Permissions** > **Permissions**.
- Administration UI: **Administration** > **Role Management** > **General Permissions**.

How do I authenticate guests in Student Self-Service Portal?

To manually authenticate guests, go to **Settings** > **Guest Authentication Mode**.

These are the possible authentication methods for accessing the Self-Service portal:

- OCI Authentication. This is enabled by default unless you selected **Enable External IDP Authentication**. If you do, make sure you save your changes.
- External Identity Provider (IDP) Authentication

For more information about authentication and access, see *Manage Authentication for Administrators and Non-Administrators*.

How do I configure attribute names to access the Administration UI?

You can use a workbook to define user-friendly names for data items used in the Security Assertion Markup Language (SAML) single sign-on feature for the Administration UI.

The configuration workbooks are delivered in the form of Excel spreadsheets. The file name for this workbook is **SAML_ATTRIBUTE.csv**.

Every configurable field of the Administration UI is represented as a spreadsheet column. Here are the column descriptions:

- Column: Spreadsheet column associated with the field.
- Required?: Denote if you're required to populate the field or not.
- Field Type and Accepted Values:
  - String: These are usually alphanumeric.
  - Integer: Numeric, a whole number.
  - Double: Numeric with places to the right of the decimal.
  - Enumeration: List of values.
  - Boolean: True or False, Yes or No.
  - Groovy script: A logical query using groovy script.
  - Date: yyyy-mm-dd format.

Field Name	Column	Required	Field Type and Accepted Values	Description

Data_Item	A	Y	String Alphanumeric ASCII text, up to 255 characters. Example Values: • First Name • Last Name	Fixed list of SAML Assertion attributes.
SAML_ Atribute_ Name	B	Y	String Alphanumeric ASCII text, up to 255 characters. Example Values: • FirstName • LastName	Customer defined SAML attribute name

You don't have to complete any setup in the UI to enable this configuration. You must have the administrator role permission to enable and view the configuration on the Administration UI. See *Set General Permissions Matrix* for more information.

How do I configure the data items names on the student portal?

You can configure user-friendly names for data items used in Security Assertion Markup Language (SAML) single sign-on when accessing the Student Self-Service portal.

To assign names to the SAML attribute data items:

1. Sign in as an administrator to the Student Self-Service portal.
2. Navigate to **Settings** > **SAML Attribute Name Mapping**.
3. Select the data item to change.
4. Enter an attribute name or friendly name.
5. Click **Save**.

Here are the delivered data items with the corresponding default attribute names:

Data Item	Attribute Name or Friendly Name
First Name	FirstName
Last Name	LastName
Email	EmailAddress
Roles	Roles

Data Item	Attribute Name or Friendly Name
Phone	PhoneNumber
External ID	StudentID

How do I exchange information with Student Financial Aid?

Institutions exchange information with Student Financial Aid (SFA) by sending or retrieving messages and files through the Message Processing Gateway (MPG) application.

Outbound event messages are pulled from MPG in first in first out (FIFO) sequence. Once a message is pulled from the MPG, it disappears from the outbound queue.

Messages and files from the institution or student must be saved with a `.xml` extension because they contain XML payloads. To send and retrieve messages or files, you need a software client to send HTTP requests, like *cURL* or *Postman*. You can use any client you're comfortable with as long as you provide the correct user name, password, file, and web service endpoint.

Related Topics

- Set up Access to REST API

Post a File to MPG

Here's an example of issuing a POST command using cURL:

```
curl -kv -X POST -H "Authorization: Bearer <token value>" --key fasapi https://sfp.ocs.oraclecloud.com/
customer/mpg/v2/channels/in.ev/publish -H 'Vocado-Message-Class: FasStudentInitiationEvent' -H
 "Content-Type: application/vocado-message+xml" --data-binary "@/Users/Username/Desktop/student/
FasStudentInitiationEvent.xml" -H'Vocado-Message-Id: 11111111-1111-1111-1111-111111111111
```

Pull a File from MPG

These examples use cURL to show how you can retrieve messages or files based on various situations.

Here's how you pull messages using its message ID:

```
curl -k -X POST -H "Authorization: Bearer <token value>" https://sfp.ocs.oraclecloud.com/mpg/v2/info/store/
messages?messageId=11111111-1111-1111-1111-111111111111
```

You can use the *Online GUID Generator* to create unique message IDs.

Here's how you pull messages using the FIFO sequence:

```
curl -X POST -H "Authorization: Bearer <token value>" -H 'Accept: application/vocado-
message+xml' https://sfp.ocs.oraclecloud.com/mpg/v2/channels/out.ev/consume -H 'Vocado-Pull-
Id:11111111-1111-1111-1111-111111111111'
```

How do I exchange information with the US Department of Education?

Institutions, through Student Financial Aid, exchange information like files and messages with the US Department of Education (ED) through the Vocado US Department of Education Gateway (VUG) application.

SFA is updated whenever ED makes changes to any record file layouts.

Before you start

Make sure you save financial aid history files like Institutional Student Information Reports (ISIRs) and National Student Loan Data System (NSLDS) with a `.dat` extension. Common Origination and Disbursements (COD) are XML payloads and must be saved with a `.xml` extension.

To send and retrieve messages or files, you need a software client to send HTTP requests, like *cURL* or *Postman*. You can use any client you're comfortable with as long as you provide the correct user name, password, file, and web service endpoint.

Naming Conventions for SAIG File Types

During the testing and demo phases of implementation, you have to make sure that mock SAIG files adhere to the SAIG Message Class naming conventions defined by the U.S. Department of Education at *https://fsapartners.ed.gov*.

Note: If an NSLDS file load is successful using Postman, but the NSLDS data doesn't appear in **Administration** > **FAS Management**, review the content of the file and check that it meets all specifications outlined in the NSLDS record layouts from ED.

Related Topics

- SAIG Administration
- Set up Access to REST API

Post a SAIG File to VUG

Here's an example of issuing a POST command using cURL:

```
curl -ki -X POST -H "Authorization: Bearer <token value>" --key fasapi -F "file=@/Users/Username/Desktop/
student/idap20op-a.dat" https://sfp.ocs.oraclecloud.com/customer/vug/v2/in/publish
```

Pull a SAIG File from VUG

Here's how you pull information (file) from VUG:

```
curl -X POST -H "Authorization: Bearer <token value>" https://sfp.ocs.oraclecloud.com/vug/v2/out/consume?
limit=1
```

How do I manage roles and permissions in Student Self-Service?

You can create and maintain permissions for the role categories to control access to the Student Self-service portal.

To add permissions to the **Student**, **Guest**, and **Admin** role categories:

1. Sign in as an administrator to the Student Self-service portal.
2. Navigate to **Settings** > **Role Permissions**.
3. Click **Add Role**, and on the **Details** tab, select a role category.
4. Enter a **Description**. This value populates the **Code** field by default.
5. Click the pencil icon to edit the **Code** field. This field must be unique. Don't enter a before used Role Code.
6. Select the desired **Permissions** for this **Role** by selecting the associated check boxes.
7. Click **Save**.

To view users, and assign or remove a role to the users:

1. Select a **Role** and click the **User Assignment** tab.
2. All users are displayed. Users who are assigned that role have an active check box.
3. To search for a user, enter any part of the user's First Name, Last Name, or External Student ID and click **Search**. Searches are in the context of the selected **Role**.
4. Select or clear the check box beside the name to assign or remove the assigned role.
5. Click **Save**.

How do I prepare for a successful implementation of Student Financial Aid?

Before you begin with your implementation, make sure you have these items ready and that you've completed the referenced tasks to successfully implement Student Financial Aid (SFA).

1. Activate your order. See *Oracle Student Financial Aid: How to autoprovision a new SFP instance*.
2. Confirm you received the Welcome e-mail from Oracle.
3. Verify access to your instance of Student Financial Aid (SFA).
4. Verify access to your instance of Student Self Service (if applicable).
5. Verify access to your instance of Oracle Analytics Server (OAS).
6. (Optional) Submit a Service Request in My Oracle Support (MOS) if you want to request your environments only be accessible from specific IP addresses and IP address ranges through the use of IP Allow lists. When you submit the Service Request, provide the CIDR blocks that should be allowed access.
7. (Optional) If you want to set up firewall rules within your network to allow inbound or outbound access for Oracle Services Network (OSN), refer to the *OSN Public IP Ranges* documentation. Use that documentation as the source of

truth for SFA public IP addresses. You can poll the published file to check for new IP address ranges as frequently as every 24 hours. We recommend that you poll the published file at least weekly.

For SFA, we recommend that you use the CIDR blocks for the **us-ashburn-1** and **us-phoenix-1** regions.

> **Note:** Egress traffic sourcing from the SFA Oracle Analytics Server (OAS) implementation uses a static address. For more information, see *Oracle Student Financial Planning Cloud Service - Oracle Analytics Server Static Address*.

8. Access the latest Release Notes available on *Cloud Customer Connect*. You can search for a specific release using the "Release Notes" tag.
9. Download the latest API assets and the Baseline Configuration Release Extract from *Cloud Customer Connect*. You can search for a specific release using the "Release Notes" tag.
10. See *Oracle Fusion Cloud Student Financial Planning: Integration Management* for supplemental information:

How do I switch between multiple student accounts?

The Student Self-service landing page displays a welcome greeting with the user's name and student's name and student's current enrollment and program type. As a parent, if you've been granted parent access to more than one student, you can click the student's name in the upper right area to toggle between students without signing out.

The Student Self-service portal landing page defaults to the last student account viewed and signed out from. Pay attention and switch to the correct student account after signing back in.

When you add a parent to your student account, you’ll receive an email alert. You don’t need to create another account for the parent to sign in.

> **Note:** Contact the Student Self-service portal administrator to:
>
> - Assign multiple students to one parent account.
> - Update email and receive notifications for a second parent.

How do I set up an external document management system?

To set up an external document management system (DMS) instead of using the delivered DMS in Student Self Service, log a service request providing the base URL of your DMS.

For example, if you access your DMS using a URL like `schooldms.com/docid123` where docid123 is the document's unique ID, then your DMSs base URL would be `schooldms.com`. Usually if you're using an external DMS, you're also using an external Student Self Service solution.

How many consumers can I connect to Message Processing Gateway?

You can connect multiple consumers.

Remember that messages are pulled from Message Processing Gateway (MPG) in first in first out sequence, so once a message is pulled from MPG it disappears from the outbound queue. For consistency, we recommend connecting only one consumer for event-based outbound messages.

What are the commands to post SAIG files and files from the institution or student?

Here's where you can find example cURL commands to post SAIG files like ISIR, NSLDS (SFP only), COD (SFP only) as well as files from the institution or student.

To post or load files to the SAIG file repository, see *How do I exchange information with the US Department of Education?*.

To post or load files to the MPG application, see *How do I exchange information with Student Financial Aid?*.

If you need the credentials, contact your service administrator.

What are examples of Groovy scripts in Student Financial Aid?

Customers can configure Groovy scripts to make automation decisions based on their desired business practices and outcomes.

The full set of updated configuration workbooks, including the Groovy scripts, is delivered with every product release as a part of Baseline Configuration. The latest baseline configuration can be found in the release notes posted on *Cloud Customer Connect*. You can search for a specific release using the "Release Notes" tag.

Here's a list of Groovy script examples for how various areas of the application can be configured.

- *awardOverridingCriteria*
- *awardYearSelectionCriteria*
- *coaCostValue*
- *configurableDisbursementCriteria*
- *directLoanCrossOverLpAwySelection*
- *disbursementDates*

- *documentMetadataAdditionalResolutionAction*
- *enrollmentStatusOverride*
- *globalEnrollmentStatusCriteria*
- *gradeLevelProgression*
- *isirAssumptionAutoCodeClearingLogic / isirCCodeAutoCodeClearingLogic / isirHighlightAutoCodeClearingLogic / isirRejectCodeAutoCodeClearingLogic / isirVerificationClearanceRules*
- *isirAssumptionDocumentsRequired / isirCCodeDocumentsRequired / isirHighlightsDocumentsRequired / isirRejectCodeDocumentsRequired / isirVerificationDocumentsRequired*
- *isirDiscrepancyEvaluation*
- *isirUseSubsequentIsir*
- *isirUsedInPackaging*
- *letterCodeCriteria*
- *nfrDisbursementDates*
- *nfrEligibility*
- *nfrMaximumProjectedAwardAmount*
- *nfrPeriods*
- *nonTermAcyMonths*
- *pellEnrollmentIntensity*
- *plusCreditDecisionMatchingCriteria*
- *summerTermType*
- *termPaymentPeriodMonths*

awardOverridingCriteria

Workbook: FAS_FUND_CONFIG.csv

Column: Award_Overriding_Criteria

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.AwardOverridingCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*
import org.joda.time.LocalDate

/* Baseline - Award Overriding Criteria for DSUB DUNSUB - combined Term and Non-Term */
static <T> Closure<T> lazy(Closure<T> init) {
 T value
 boolean isInitDue = true
 return {
 if (isInitDue) {
 isInitDue = false
 value = init()
 }
 value
 }
}

def disbursementsInThisPp = lazy {
 disbursements.getInScope(scope)
}
```

```
def nonCancelledDisbursementsInThisPp = lazy {
 disbursementsInThisPp()
 .getCancelled(false)
 .filterStatuses { status -> status != "DISBURSEMENT_CANCELED" }
}

def disbursedDisbursementsInThisPp = lazy {
 nonCancelledDisbursementsInThisPp().getWithStatus("DISBURSED")
}

def thisPpIsDisbursed = lazy {
 log.debug(!disbursedDisbursementsInThisPp().empty, "DISBURSED PP")
}

def thisPpFundOutput = lazy {
 partialFundOutput.getInLoanPeriod(lpNo).getInPaymentPeriod(ppNo)
}

def thisPpIsProrated = lazy {
 log.debug(thisPpFundOutput().eligible && thisPpFundOutput().prorated, "PRORATED PP")
}

def isInNASuffix = term && term.standard && acy.terms.findAll { t -> t.startDate >= term.startDate &&
 t.standard }
 .every { t -> t.notAttending }

def disbursedAmount = thisPpIsDisbursed()
 ? disbursedDisbursementsInThisPp().totalDisbursementAmount
 : (BigDecimal) null
def acceptedAmount = fundAcceptance.getInScope(scope)
 .filterStatuses {it != "PENDING_ACCEPTANCE" }
 .let {it.empty ? null : it.validatedAcceptedAmount }
def effectiveAcceptedAmount = thisPpIsDisbursed() ? disbursedAmount : acceptedAmount

def now = LocalDate.now()
def activeSummerTerms = acy.terms.filter {
 it.summer && program.courses.getAssociatedTo(it)
 .any {
 it.schedulingStatus != "PROJECTED"
 }
}
def activeSummerTrailers = activeSummerTerms.filter {
 it.summerTrailer
}
def summerTrailerHasCourses = !activeSummerTrailers.empty
def summerTrailerStart = summerTrailerHasCourses ? activeSummerTrailers.startDate : null
def isActiveSummerTerm = activeSummerTerms.contains(term)
def isBeyondSummerTrailerThreshold = summerTrailerStart && now >= summerTrailerStart.minusDays(60)
def summerDocExists = lazy {
 def summerDocsInAwy = receivedDocuments.forAwardYear(awy)
 .getDocumentsForDocCode("SummerApplication", "Student")
 log.debug(summerDocsInAwy != null && !summerDocsInAwy.empty, "SUMMER DOC EXISTS")
}

def isSummerTrailerState = log.debug(summerTrailerHasCourses && isBeyondSummerTrailerThreshold &&
 summerDocExists(), "SUMMER TRAILER STATE")
def isActiveSummerHeader = log.debug(term?.summerHeader && isActiveSummerTerm, "ACTIVE SUMMER HEADER")
def isLessThanHalfTime = enrollmentStatus == "LESS_THAN_HALF_TIME"
def standardOrSummerHeader = (term?.isStandard() || isActiveSummerHeader)

optimizer
 .setCumulativeGradeLevelLimitApplicable(
 (!term || standardOrSummerHeader)
 && (!isLastAcy || program.undergraduate)
 && (!isLastAcy || !isInNASuffix)
```

```
  && !thisPpIsProrated())
  .setFundingForPeriodFilter { fundCode, fundType -> fundCode != "DISCOUNT" }
  .setCompensationValue(0)
  .addPhase(thisPpIsDisbursed(), 1) {
  it.withGiven(disbursedAmount)
  }

 if (!isLessThanHalfTime) {
  optimizer
  .addPhase(!term, 2)
  .addPhase(term && standardOrSummerHeader && (!isSummerTrailerState || effectiveAcceptedAmount), 2) {
  it.withPpMaximum(isSummerTrailerState ? effectiveAcceptedAmount : null)
  }
  .addPhase(isSummerTrailerState && (isActiveSummerTerm || standardOrSummerHeader && effectiveAcceptedAmount
  == null), 3)
 }

 if (program.enrollmentStatus == "X" || pp.status == "CANCELED")
 {
  return awardInfo.withMaxAmount(0.0).withRetainedAmount(0.0)
 }
 if (program.term)
 {
  def overlappingTerms = primaryProgram.terms.getOverlappingWith(term)
  /* Methods returning collection, as well as filters, will be plural; otherwise singular */
  if (overlappingTerms.studentsTermStatuses.contains("WITHDRAWN"))
  {
  log.debug("STUDENT TERM IS WITHDRAWN")

  def r2t4OverlappingTerms = r2t4.getOverlappingWith(term)
  .getWithProcessStatuses(["NOT_REQUIRED", "COMPLETED"])
  .getWithOldas(overlappingTerms.oldas)

  if (!r2t4OverlappingTerms.empty)
  {
  log.debug("R2T4 Overlapping Terms is not empty, FREEZING!")

  def sum = nonCancelledDisbursementsInThisPp().ppMaxDisbursementAmount

  return awardInfo.withMaxAmount(sum).withRetainedAmount(sum)
  }
  else
  {
  log.debug("R2T4 PROCESS NOT YET IN A FREEZING STATE")
  }
  }

  if (enrollmentStatus == "NOT_ATTENDING")
  {
  log.debug("ZEROING - enrollmentStatus={} ", enrollmentStatus)
  return awardInfo.withMaxAmount(0.0).withRetainedAmount(0.0)
  }

 }

 if (log.debug(lp.endDate <= now, "LOAN PERIOD IN THE PAST")
  || (thisPpIsProrated() && thisPpIsDisbursed()))
 {
  def frozenAmount = disbursedAmount == null ? 0.0 : disbursedAmount
  log.debug("FREEZING TO {}", frozenAmount)
  return awardInfo.withMaxAmount(frozenAmount).withRetainedAmount(frozenAmount)
 }

 return awardInfo
```

awardYearSelectionCriteria

Workbook: FAS_FUND_CONFIG.csv

Column: Award_Year_Selection_Criteria

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.AwardYearSelectionCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Pell Award Year Selection Criteria - combined Term and Non-Term*/

//variables for earlier and later year ISIRs
def earlierYearISIRExists = isirRecords.getCurrentIsirForAwardYear(crossoverHelper.getCurrentAwardYear()) !=
 null
def laterYearISIRExists = isirRecords.getCurrentIsirForAwardYear(crossoverHelper.getUpcomingAwardYear()) !=
 null

//variables for earlier and later year Pell disbursements that are in "Disbursed" status
def earlierYearPellDisbursementExists = crossoverHelper.hasDisbursedDisbursementsInCurrentAwardYear()
def laterYearPellDisbursementExists = crossoverHelper.hasDisbursedDisbursementsInUpcomingAwardYear()

//if an ISIR and disbursements exist in the earlier year, then select the earlier year
if (earlierYearISIRExists && earlierYearPellDisbursementExists) {
 return crossoverHelper.useCurrentAlternative()
}

//if an ISIR and disbursements exist the later year, then select the later year
if (laterYearISIRExists && laterYearPellDisbursementExists) {
 return crossoverHelper.useUpcomingAlternative()
}

//if an ISIR exists in the earlier year and remaining eligibility > $0 for pell in the earlier year, then
 select the earlier year
if (earlierYearISIRExists && currentFunds.getFund("PELL").getOverlappingWith(pp).maxAmount > 0) {
 return crossoverHelper.useCurrentAlternative()
}

//if an ISIR exists in the later year and no ISIRs exist in the earlier year, then select the later year
if (laterYearISIRExists && !earlierYearISIRExists) {
 return crossoverHelper.useUpcomingAlternative()
}

//if none of the above criteria are met, then use default pell crossover logic
return crossoverHelper.useDefaultAlternative()
```

coaCostValue

Workbook: COA.csv

Column: Cost_Value

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.CoaCostValueScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

if (receivedDocuments.forAwardYear(awy).hasDoc("Housing")) {
 def housingDoc = receivedDocuments.forAwardYear(awy).get("Housing")
 def housingStatus = housingDoc.getDocumentField("AC1116")
 if (housingStatus == "On Campus" || housingStatus == null || housingStatus.isAllWhitespace()) {
 return 3700
 }
 if (housingStatus == "Off Campus" || housingStatus == "With Parent") {
 return 0
 }
}
else if (isirRecords.getCurrentIsirForAwardYear(2024) != null) {
 def housingCode =
 isirRecords.getCurrentIsirForAwardYear(2024).getIsirFieldValue("FEDERALSCHOOLCODE1HOUSINGPLANS")
 if (housingCode == "1") {
 return 3600
 }
}

return 3600
```

configurableDisbursementCriteria

Workbook: DISBURSEMENT.csv

Column: Disbursement_Criteria

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.ConfigurableDisbursementCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

if (program.isTerm()) {
 def termEnrollmentStatus = term.getStatus();

 if (["HALF_TIME", "THREE_QUARTER_TIME", "FULL_TIME"].contains(termEnrollmentStatus)) {
 return true;
 }
 return false;
}
return null;
```

directLoanCrossOverLpAwySelection

Workbook: PKGSCHEDATTRIB.csv

Column: Direct_Loan_Crossover_LP_AwY_Selection

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.DirectLoanCrossOverLpAwySelectionScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*
import org.joda.time.LocalDate

//If the Loan Period start date is prior to to 07/01/24, return null
if (lp.startDate < new LocalDate(2024, 07, 01)) {
 return null
}

//If there are disbursed TIV DL disbursements for the Loan Period return the earliest award year found for
 those disbursements
def tivDisbursementsInLP = disbursements
 .getTitleIv()
 .getLoans()
 .getInLoanPeriod(lp)
def firstDisbursedAwardYear = candidateAwys.find{tivDisbursementsInLP
 .getInAwardYear(it)
 .getStatuses()
 .contains("DISBURSED")}
if (firstDisbursedAwardYear != null) {
 return firstDisbursedAwardYear
}

//If not a term program, return the current AwY
if (!program.isTerm()) {
 return candidateAwys[0]
}

//If a summer trailer term is the loan period, return the upcoming AwY
def LPIsSummerTrailer = acys
 .terms
 .getOverlappingWith(lp)
 .every{it.summerTrailer}
if (LPIsSummerTrailer) {
 return candidateAwys[1]
}

//If the academic year contains a summer header, return the upcoming AwY
def LPContainsSummerHeader = acys
 .terms
 .getOverlappingWith(lp)
 .any{it.summerHeader}
if (LPContainsSummerHeader) {
 return candidateAwys[1]
}

//If first term is a cross-over, return the upcoming AwY
def firstTermIsCrossover = helper.awardYear.isCrossover(acys.terms[0])
if (firstTermIsCrossover) {
 return candidateAwys[1]
}

//No other conditions met, return the current AwY
return candidateAwys[0]
```

disbursementDates

Workbook: FAS_FUND_CONFIG.csv

Column: Disbursement_Dates

```
return [period.startDate.plusDays(30)];
```

documentMetadataAdditionalResolutionAction

Workbook: DOCMETADATA.csv

Column: Additional_Resolution_Action

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.DocumentMetadataAdditionalResolutionActio
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*
import org.joda.time.LocalDate
```

```
//if no ISIR, then exit the script
if (isirRecord == null) {
 return
}

//Student ISIR variables
def isirSTUDENT_MARITALSTATUS = isirRecord.getField(IsirField.STUDENT_MARITALSTATUS).getOrNull()
def isirSTUDENT_DATEOFBIRTH = isirRecord.getField(IsirField.STUDENT_DATEOFBIRTH).getOrNull()
def isirSTUDENTFTIM_FILINGSTATUSCODE_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_FILINGSTATUSCODE_FTI).getOrNull()
//if FTI value is blank, then get non-FTI value instead
if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == null) {
 isirSTUDENTFTIM_FILINGSTATUSCODE_FTI =
 isirRecord.getField(IsirField.STUDENT_TAXRETURNFILINGSTATUS).getOrNull()
}

//Parent ISIR variables
def isirPARENT_DATEOFBIRTH = isirRecord.getField(IsirField.PARENT_DATEOFBIRTH).getOrNull()
def isirPARENTSPOUSE_DATEOFBIRTH = isirRecord.getField(IsirField.PARENTSPOUSE_DATEOFBIRTH).getOrNull()
def isirPARENTFTIM_FILINGSTATUSCODE_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_FILINGSTATUSCODE_FTI).getOrNull()
//if FTI value is blank, then get non-FTI value instead
if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == null) {
 isirPARENTFTIM_FILINGSTATUSCODE_FTI =
 isirRecord.getField(IsirField.PARENT_TAXRETURNFILINGSTATUS).getOrNull()
}

//Document owner variable
def docOwner = currentDocument.getDocumentOwnerTypes().get(0)

if (isirRecord == null)
{
 return
}

if (docOwner == "Student")
{
 boolean hasVerificationWorksheetIndStudent = false
 if (receivedDocuments.hasDoc("VW-Ind", "Student") && receivedDocuments.get("VW-Ind",
 "Student").isAcceptable())
 {
 hasVerificationWorksheetIndStudent = true
 }

 boolean hasSelfEmploymentStatementStudent = false
 if (receivedDocuments.hasDoc("SelfEmploymentStatement", "Student") &&
 receivedDocuments.get("SelfEmploymentStatement", "Student").isAcceptable())
 {
 hasSelfEmploymentStatementStudent = true
 }

 int selfEmploymentStudent = 0
 if (hasSelfEmploymentStatementStudent)
 {
 if (receivedDocuments.get("SelfEmploymentStatement", "Student").getDocumentField("AC1015") != null && !
receivedDocuments.get("SelfEmploymentStatement", "Student").getDocumentField("AC1015").isAllWhitespace())
 {
 selfEmploymentStudent = receivedDocuments.get("SelfEmploymentStatement",
 "Student").getDocumentField("AC1015").toFloat().round()
 }
 }

 boolean hasSelfEmploymentStatementStudentSpouse = false
 if (receivedDocuments.hasDoc("SelfEmploymentStatement", "Student Spouse") &&
 receivedDocuments.get("SelfEmploymentStatement", "Student Spouse").isAcceptable())
 {
```

```
hasSelfEmploymentStatementStudentSpouse = true
}

int selfEmploymentStudentSpouse = 0
if (hasSelfEmploymentStatementStudentSpouse)
{
if (receivedDocuments.get("SelfEmploymentStatement", "Student Spouse").getDocumentField("AC1015") !
= null && !receivedDocuments.get("SelfEmploymentStatement", "Student
Spouse").getDocumentField("AC1015").isAllWhitespace())
{
selfEmploymentStudentSpouse = receivedDocuments.get("SelfEmploymentStatement", "Student
Spouse").getDocumentField("AC1015").toFloat().round()
}
}

float w2SumStudentFloat = 0.0
def w2DocsStudent = receivedDocuments.getDocumentsForDocCode("W2", "Student")
for (doc in w2DocsStudent)
{
String ac1010 = doc.getDocumentField("AC1010")
float ac1010Value = 0.0
if (ac1010 != null && !ac1010.isAllWhitespace())
{
ac1010Value = ac1010.toFloat()
}
w2SumStudentFloat += ac1010Value
}
if (docOwner == "Student")
{
String currentAC1010 = currentDocument.getDocumentFieldValue("AC1010")
if (currentAC1010 != null && !currentAC1010.isAllWhitespace())
{
w2SumStudentFloat += currentAC1010.toFloat()
}
}
int w2SumStudent = w2SumStudentFloat.round()

float w2SumStudentSpouseFloat = 0.0
def w2DocsStudentSpouse = receivedDocuments.getDocumentsForDocCode("W2", "Student Spouse")
for (doc in w2DocsStudentSpouse)
{
String ac1010 = doc.getDocumentField("AC1010")
float ac1010Value = 0.0
if (ac1010 != null && !ac1010.isAllWhitespace())
{
ac1010Value = ac1010.toFloat()
}
w2SumStudentSpouseFloat += ac1010Value
}
if (docOwner == "Student Spouse")
{
String currentAC1010 = currentDocument.getDocumentFieldValue("AC1010")
if (currentAC1010 != null && !currentAC1010.isAllWhitespace())
{
w2SumStudentSpouseFloat += currentAC1010.toFloat()
}
}
int w2SumStudentSpouse = w2SumStudentSpouseFloat.round()

float doc1099StudentFloat = 0.0
def doc1099GDocsStudent = receivedDocuments.getDocumentsForDocCode("1099G", "Student")
for (doc in doc1099GDocsStudent)
{
String ac1013 = doc.getDocumentField("AC1013")
float ac1013Value = 0.0
if (ac1013 != null && !ac1013.isAllWhitespace())
```

```
{
ac1013Value = ac1013.toFloat()
}
doc1099StudentFloat += ac1013Value
}
int doc1099SumStudent = doc1099StudentFloat.round()

float doc1099StudentSpouseFloat = 0.0
def doc1099GDocsStudentSpouse = receivedDocuments.getDocumentsForDocCode("1099G", "Student Spouse")
for (doc in doc1099GDocsStudentSpouse)
{
String ac1013 = doc.getDocumentField("AC1013")
float ac1013Value = 0.0
if (ac1013 != null && !ac1013.isAllWhitespace())
{
ac1013Value = ac1013.toFloat()
}
doc1099StudentSpouseFloat += ac1013Value
}
int doc1099SumStudentSpouse = doc1099StudentSpouseFloat.round()

int totalIncome = 0
int StudentSpouseAge = -1
if (hasVerificationWorksheetIndStudent)
{
if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1044") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1034")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1045") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1035")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1046") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1036")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1047") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1037")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1048") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1038")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1049") == "Student Spouse")
```

```
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1039")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1050") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1040")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1051") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1041")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1052") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1042")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
else if (receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1053") == "Student Spouse")
{
String StudentSpouseAgeString = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1043")
if (StudentSpouseAgeString != null && !StudentSpouseAgeString.isAllWhitespace())
{
StudentSpouseAge = StudentSpouseAgeString.toInteger()
}
}
}

boolean studentAgeUnder65 = true
LocalDate age65Threshold = new LocalDate(1957, 12, 31)
LocalDate studentBirthday = new LocalDate(2022, 01, 01)
if (isirSTUDENT_DATEOFBIRTH != null)
{
studentBirthday = isirSTUDENT_DATEOFBIRTH.secureValue()
}
if (studentBirthday <= age65Threshold)
{
studentAgeUnder65 = false
}
int filingThreshold = 999999
if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "1" && studentAgeUnder65)
{
filingThreshold = 12950
}
if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "1" && !studentAgeUnder65)
{
filingThreshold = 14700
}
if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "2" && studentAgeUnder65 && StudentSpouseAge)
{
filingThreshold = 25900
}
if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "2" && (studentAgeUnder65 || StudentSpouseAge))
```

```
     {
     filingThreshold = 27300
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "2" && !studentAgeUnder65 && StudentSpouseAge)
     {
     filingThreshold = 28700
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "3")
     {
     filingThreshold = 5
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "4" && studentAgeUnder65)
     {
     filingThreshold = 19400
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "4" && !studentAgeUnder65)
     {
     filingThreshold = 21150
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "5" && studentAgeUnder65)
     {
     filingThreshold = 25900
     }
     if (isirSTUDENTFTIM_FILINGSTATUSCODE_FTI == "5" && !studentAgeUnder65)
     {
     filingThreshold = 27300
     }

     if (isirSTUDENT_MARITALSTATUS == "1" || isirSTUDENT_MARITALSTATUS == "2" || isirSTUDENT_MARITALSTATUS ==
     "4" || isirSTUDENT_MARITALSTATUS == null)
     {
     totalIncome = w2SumStudent + doc1099SumStudent + selfEmploymentStudent
     if (totalIncome >= filingThreshold)
     {
     /* supportingDoc.addDocument("IRSForm4868", "Student") */
     supportingDoc.addAnyDocuments(["IRSForm4868", "Student"], ["IRSExtensionApproval", "Student"],
     ["TaxReturnTranscript", "Student"],["1040x", "Student"],["ForeignTaxTranscript", "Student"])
     }
     }
     else if (isirSTUDENT_MARITALSTATUS == "3")
     {
     totalIncome = w2SumStudent + doc1099SumStudent + w2SumStudentSpouse + doc1099SumStudentSpouse +
     selfEmploymentStudent + selfEmploymentStudentSpouse
     if (totalIncome >= filingThreshold)
     {
     /* supportingDoc.addAnyDocument("IRSForm4868", "Student") */
     supportingDoc.addAnyDocuments(["IRSForm4868", "Student"], ["IRSExtensionApproval", "Student"],
     ["TaxReturnTranscript", "Student"],["1040x", "Student"],["ForeignTaxTranscript", "Student"],
    ["TaxReturnTranscript", "Student Spouse"],["1040x", "Student Spouse"],["ForeignTaxTranscript", "Student
     Spouse"])
     }
     }
    }
    else if (docOwner == "Parent" || docOwner == "Parent Spouse")
    {
     boolean ParentUnder65 = true
     boolean ParentSpouseUnder65 = true
     LocalDate age65Threshold = new LocalDate(1957, 12, 31)
     LocalDate ParentBirthday = new LocalDate(2022, 01, 01)
     LocalDate ParentSpouseBirthday = new LocalDate(2022, 01, 01)
     if (isirPARENT_DATEOFBIRTH != null)
     {
     ParentBirthday = isirPARENT_DATEOFBIRTH.secureValue()
     }
     if (isirPARENTSPOUSE_DATEOFBIRTH != null)
     {
```

```
 ParentSpouseBirthday = isirPARENTSPOUSE_DATEOFBIRTH.secureValue()
 }
 if (ParentBirthday <= age65Threshold)
 {
 ParentUnder65 = false
 }
 if (ParentSpouseBirthday <= age65Threshold)
 {
 ParentSpouseUnder65 = false
 }
 int filingThreshold = 999999

 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "1" && ParentUnder65)
 {
 filingThreshold = 12950
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "1" && !ParentUnder65)
 {
 filingThreshold = 14700
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "2" && ParentUnder65 && ParentSpouseUnder65)
 {
 filingThreshold = 25900
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "2" && (ParentUnder65 || ParentSpouseUnder65))
 {
 filingThreshold = 27300
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "2" && !ParentUnder65 && ParentSpouseUnder65)
 {
 filingThreshold = 28700
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "3")
 {
 filingThreshold = 5
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "4" && ParentUnder65)
 {
 filingThreshold = 19400
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "4" && !ParentUnder65)
 {
 filingThreshold = 21150
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "5" && ParentUnder65)
 {
 filingThreshold = 25900
 }
 if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "5" && !ParentUnder65)
 {
 filingThreshold = 27300
 }

 int selfEmploymentParent = 0
 if (receivedDocuments.hasDoc("SelfEmploymentStatement", "Parent") &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent").isAcceptable() &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent").getDocumentField("AC1015") != null && !
receivedDocuments.get("SelfEmploymentStatement", "Parent").getDocumentField("AC1015").isAllWhitespace())
 {
 selfEmploymentParent = receivedDocuments.get("SelfEmploymentStatement",
 "Parent").getDocumentField("AC1015").toFloat().round()
 }

 int selfEmploymentParentSpouse = 0
 if (receivedDocuments.hasDoc("SelfEmploymentStatement", "Parent Spouse") &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent Spouse").isAcceptable() &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent Spouse").getDocumentField("AC1015") !
= null && !receivedDocuments.get("SelfEmploymentStatement", "Parent
 Spouse").getDocumentField("AC1015").isAllWhitespace())
 {
 selfEmploymentParentSpouse = receivedDocuments.get("SelfEmploymentStatement", "Parent
 Spouse").getDocumentField("AC1015").toFloat().round()
 }

 float w2SumParentFloat = 0.0
 def w2DocsParent = receivedDocuments.getDocumentsForDocCode("W2", "Parent")
 for (doc in w2DocsParent)
 {
 String ac1010 = doc.getDocumentField("AC1010")
 float ac1010Value = 0.0
 if (ac1010 != null && !ac1010.isAllWhitespace())
 {
 ac1010Value = ac1010.toFloat()
 }
 w2SumParentFloat += ac1010Value
 }
 if (docOwner == "Parent")
 {
 String currentAC1010 = currentDocument.getDocumentFieldValue("AC1010")
 if (currentAC1010 != null && !currentAC1010.isAllWhitespace())
 {
 w2SumParentFloat += currentAC1010.toFloat()
 }
 }
 int w2SumParent = w2SumParentFloat.round()

 float w2SumParentSpouseFloat = 0.0
 def w2DocsParentSpouse = receivedDocuments.getDocumentsForDocCode("W2", "Parent Spouse")
 for (doc in w2DocsParentSpouse)
 {
 String ac1010 = doc.getDocumentField("AC1010")
 float ac1010Value = 0.0
 if (ac1010 != null && !ac1010.isAllWhitespace())
 {
 ac1010Value = ac1010.toFloat()
 }
 w2SumParentSpouseFloat += ac1010Value
 }
 if (docOwner == "Parent Spouse")
 {
 String currentAC1010 = currentDocument.getDocumentFieldValue("AC1010")
 if (currentAC1010 != null && !currentAC1010.isAllWhitespace())
 {
 w2SumParentSpouseFloat += currentAC1010.toFloat()
 }
 }
 int w2SumParentSpouse = w2SumParentSpouseFloat.round()

 float doc1099ParentFloat = 0.0
 def doc1099GDocsParent = receivedDocuments.getDocumentsForDocCode("1099G", "Parent")
 for (doc in doc1099GDocsParent)
 {
 String ac1013 = doc.getDocumentField("AC1013")
 float ac1013Value = 0.0
 if (ac1013 != null && !ac1013.isAllWhitespace())
 {
 ac1013Value = ac1013.toFloat()
 }
 doc1099ParentFloat += ac1013Value
 }
 int doc1099SumParent = doc1099ParentFloat.round()
```

```
  float doc1099ParentSpouseFloat = 0.0
  def doc1099GDocsParentSpouse = receivedDocuments.getDocumentsForDocCode("1099G", "Parent Spouse")
  for (doc in doc1099GDocsParentSpouse)
  {
  String ac1013 = doc.getDocumentField("AC1013")
  float ac1013Value = 0.0
  if (ac1013 != null && !ac1013.isAllWhitespace())
  {
  ac1013Value = ac1013.toFloat()
  }
  doc1099ParentSpouseFloat += ac1013Value
  }
  int doc1099SumParentSpouse = doc1099ParentSpouseFloat.round()

  int totalIncome = 0

  if (isirPARENTFTIM_FILINGSTATUSCODE_FTI == "3")
  {
  totalIncome = selfEmploymentParent + selfEmploymentParentSpouse + w2SumParent + w2SumParentSpouse +
  doc1099SumParent + doc1099SumParentSpouse
  if (totalIncome >= filingThreshold)
  {
  supportingDoc.addAnyDocuments(["IRSForm4868", "Parent"], ["IRSExtensionApproval", "Parent"],
  ["TaxReturnTranscript", "Parent"],["1040x", "Parent"],["1040", "Parent"],["ForeignTaxTranscript",
  "Parent"],["IRSForm4868", "Parent Spouse"],["IRSExtensionApproval", "Parent Spouse"],
 ["TaxReturnTranscript", "Parent Spouse"],["1040x", "Parent Spouse"],["1040", "Parent Spouse"],
 ["ForeignTaxTranscript", "Parent Spouse"])
  }
  }
  else
  {
  supportingDoc.addAnyDocuments(["IRSForm4868", "Parent"], ["IRSExtensionApproval", "Parent"],
  ["TaxReturnTranscript", "Parent"],["1040x", "Parent"],["1040", "Parent"],["ForeignTaxTranscript",
  "Parent"])
  }
 }
```

enrollmentStatusOverride

Workbook: FAS_FUND_CONFIG.csv

Column: Enrollment_Status_Override_Census_Dates

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.EnrollmentStatusOverrideScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Pell Enrollment Status Override with Census Date - combined Term and Non-Term*/

import org.joda.time.LocalDate;

if (program.isTerm())
{
 LocalDate termStartDate = term.getStartDate();
 LocalDate termEndDate = term.getEndDate();
 LocalDate censusDate;
 LocalDate today = LocalDate.now();

 String programType = primaryProgram.getProgramType();

 censusDate = term.getStartDate().plusDays(10);
```

```
log.debug("CONFIG LOGGING STARTS HERE");
log.debug("censusDate = {}", censusDate);
log.debug("today = {}", today);
log.debug("termStartDate = {}", termStartDate);

def safiTerm = program.getTerms().getOverlappingWith(term).get(0);

if (censusDate != null && today > censusDate && (program.getEnrollmentStatus() == "W" ||
safiTerm.getStudentsTermStatus() == "WITHDRAWN"))
{
return enrollmentStatus;

/*
def countedUnits = 0.0;

for (def course: program.getCourses().getAssociatedTo(period))
{
if (["WITHDRAWN"].contains(course.getSchedulingStatus()))
{
if (course.getLastDateOfAttendance() != null && ((program.getOfficialLastDateOfAttendance() != null &&
course.getLastDateOfAttendance() == program.getOfficialLastDateOfAttendance()) || (safiTerm.getOlda() !=
null && course.getLastDateOfAttendance() == safiTerm.getOlda())))
{
log.debug("GLOBAL ENROLLMENT SCRIPT - WITHDRAWN: courseLDA = {}, courseSchedulingStatus = {},
programOLDA = {}, courseUnitsToAdd = {}", course.getLastDateOfAttendance(), course.getSchedulingStatus(),
program.getOfficialLastDateOfAttendance(), course.getUnits());

countedUnits = countedUnits + course.getUnits();
}
}
else if (["SCHEDULED", "PASSED", "FAILED"].contains(course.getSchedulingStatus()))
{
log.debug("GLOBAL ENROLLMENT SCRIPT - SCHEDULED PASSED FAILED: courseSchedulingStatus = {},
courseUnitsToAdd = {}", course.getSchedulingStatus(), course.getUnits());

countedUnits = countedUnits + course.getUnits();
}
}
def NA = 0.0;
def LTHT = 6.0;
def HT = 9.0;
def TQT = 12.0;

return helper.mapRanges()
.le(NA, "NOT_ATTENDING")
.lt(LTHT, "LESS_THAN_HALF_TIME")
.lt(HT, "HALF_TIME")
.lt(TQT, "THREE_QUARTER_TIME")
.defaultValue("FULL_TIME")
.apply(countedUnits);
*/
}
else if (censusDate != null && today > censusDate)
{
log.debug("FIRST IF PASSED");
def previousFpo;
if (previousFinancialPlan.effectiveAt(censusDate) != null)
{
log.debug("USING CENSUS DATE PREVIOUS FPO");
previousFpo = previousFinancialPlan.effectiveAt(censusDate);
}
else if (financialPlan.getVersion() > 1)
{
log.debug("USING FPO VERSION - 1 PREVIOUS FPO");
previousFpo = previousFinancialPlan.withVersion(financialPlan.getVersion() - 1);
}
```

```
 if (previousFpo != null)
 {
 def pellFundStatuses = previousFpo.getFund("PELL").getInAwardYear(awy).getFundStatuses();
 if (!pellFundStatuses.contains("AWARDED") && !pellFundStatuses.contains("ESTIMATED"))
 {
 return enrollmentStatus;
 }
 log.debug("PREVIOUS FPO IS NOT NULL");
 def previousPell = previousFpo.getFund("PELL").getInAcademicYear(acyNo).getInPaymentPeriod(ppNo);
 if (previousPell.isEligible())
 {
 log.debug("PREVIOUS PELL IS ELIGIBLE");
 if (!previousPell.getEnrollmentStatuses().isEmpty())
 {
 log.debug("PREVIOUS PELL HAS NON NULL ENROLLMENT STATUS");
 log.debug("PREVIOUS PELL ENROLLMENT STATUS = {}", previousPell.getEnrollmentStatuses().get());
 return previousPell.getEnrollmentStatuses().get();
 }
 }
 }
 }

 return enrollmentStatus;
}
return enrollmentStatus;


9. globalEnrollmentStatusCriteria

Workbook: SCHOOL.csv
Column: Enrollment_Status_Determination
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
```

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.GlobalEnrollmentStatusCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Global Enrollment Status - SFP Baseline - Combined Term and Non-Term */

/*
def actualCourses = helper.filterCourses(program)
 .period(term)
 .safiCreatedDate(safiCreatedDate)
 .go();
*/

import org.joda.time.LocalDate;

if (program.isTerm())
{

 // If the student decides to cancel their enrollment and the Program Enrollment Status is X, then return
 NOT_ATTENDING
 if (primaryProgram.getEnrollmentStatus() != null && primaryProgram.getEnrollmentStatus() == "X")
 {
 return "NOT_ATTENDING";
 }

 def countedUnits = 0.0;
 def safiTerm = program.getTerms().getOverlappingWith(term).get(0);

 // Institutions should define their own add drop policy
 def addDropPolicy = term.getStartDate().plusDays(10);

 log.debug ("Term start date = {}, term start date + 10 = {}", term.getStartDate(), addDropPolicy);

 for (def course: program.getCourses().getAssociatedTo(period))
 {
```

```
 log.debug("GLOBAL ENROLLMENT SCRIPT, COURSE IN LOOP: courseEndDate = {}, courseAraIndicator = {},
 courseSchedulingStatus = {}, courseUnitsToAdd = {}", course.getEndDate(), course.getFirstAraIndicator(),
 course.getSchedulingStatus(), course.getUnits());

 // If the course is withdrawn, only count the credits if the course was withdrawn after the add/drop
 period. If the course is not withdrawn, count the course credits.
 if ((["WITHDRAWN"].contains(course.getSchedulingStatus()) && course.getLastDateOfAttendance() != null &&
 course.getLastDateOfAttendance() > addDropPolicy) || course.getSchedulingStatus() != "WITHDRAWN")
 {
 countedUnits = countedUnits + course.getUnits();
 }
 }

 def NA = 0.0;
 def LTHT = 6.0;
 def HT = 9.0;
 def TQT = 12.0;

 return helper.mapRanges()
 .le(NA, "NOT_ATTENDING")
 .lt(LTHT, "LESS_THAN_HALF_TIME")
 .lt(HT, "HALF_TIME")
 .lt(TQT, "THREE_QUARTER_TIME")
 .defaultValue("FULL_TIME")
 .apply(countedUnits);
}
return "FULL_TIME";
```

globalEnrollmentStatusCriteria

Workbook: SCHOOL.csv

Column: Enrollment_Status_Determination

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.GlobalEnrollmentStatusCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Global Enrollment Status - SFP Baseline - Combined Term and Non-Term */

/*
def actualCourses = helper.filterCourses(program)
 .period(term)
 .safiCreatedDate(safiCreatedDate)
 .go();
*/

import org.joda.time.LocalDate;

if (program.isTerm())
{

 // If the student decides to cancel their enrollment and the Program Enrollment Status is X, then return
 NOT_ATTENDING
 if (primaryProgram.getEnrollmentStatus() != null && primaryProgram.getEnrollmentStatus() == "X")
 {
 return "NOT_ATTENDING";
 }

 def countedUnits = 0.0;
 def safiTerm = program.getTerms().getOverlappingWith(term).get(0);
```

```
  // Institutions should define their own add drop policy
  def addDropPolicy = term.getStartDate().plusDays(10);

  log.debug ("Term start date = {}, term start date + 10 = {}", term.getStartDate(), addDropPolicy);

  for (def course: program.getCourses().getAssociatedTo(period))
  {
  log.debug("GLOBAL ENROLLMENT SCRIPT, COURSE IN LOOP: courseEndDate = {}, courseAraIndicator = {},
  courseSchedulingStatus = {}, courseUnitsToAdd = {}", course.getEndDate(), course.getFirstAraIndicator(),
  course.getSchedulingStatus(), course.getUnits());

  // If the course is withdrawn, only count the credits if the course was withdrawn after the add/drop
  period. If the course is not withdrawn, count the course credits.
  if ((["WITHDRAWN"].contains(course.getSchedulingStatus()) && course.getLastDateOfAttendance() != null &&
  course.getLastDateOfAttendance() > addDropPolicy) || course.getSchedulingStatus() != "WITHDRAWN")
  {
  countedUnits = countedUnits + course.getUnits();
  }
  }

  def NA = 0.0;
  def LTHT = 6.0;
  def HT = 9.0;
  def TQT = 12.0;

  return helper.mapRanges()
  .le(NA, "NOT_ATTENDING")
  .lt(LTHT, "LESS_THAN_HALF_TIME")
  .lt(HT, "HALF_TIME")
  .lt(TQT, "THREE_QUARTER_TIME")
  .defaultValue("FULL_TIME")
  .apply(countedUnits);
}
return "FULL_TIME";
```

gradeLevelProgression

Workbook: PKGSCHEDATTRIB.csv

Column: Grade_Level_Progression

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.GradeLevelProgressionScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

def referenceDate = program.isTerm() ? term.getStartDate() : acy.getStartDate();

def totalPrevAcyUnits = acys
 .findAll { it.getNumber() < acyNo }
 .collect { it.getStatus() == "ACTUAL" ? it.getCompletedUnits() : it.getUnits() }
 .sum(0.0);

def totalCurAcyUnits = acy.getLoanPaymentPeriods()
 .findAll { it.getStatus() != "CANCELED" && it.getStartDate() < referenceDate }
 .collect { it.getUnits() }
 .sum(0.0);

def totalCreditsCompletedBeforeProgramStartDate = program.getCourses()
 .findAll {
 it.getSchedulingStatus() == "PASSED" &&
 it.getEndDate() < actualStartDate &&
 (it.getIncompleteResolutionDate() == null || it.getIncompleteResolutionDate() < actualStartDate)
```

```
     }
     .collect { it.getUnits() }
     .sum(0.0);

    def totalCreditUnits = (program.getTotalAcceptedTransferUnits() +
     program.getAssessedUnits() +
     totalPrevAcyUnits +
     totalCurAcyUnits +
     totalCreditsCompletedBeforeProgramStartDate);

    def creditPerAydUnit = totalCreditUnits / program.getAydUnits();

    def baseGradeLevel = (program.isGraduate()
     ? 6
     : program.getProgramType() == "CERTIFICATE_POST_BACCALAUREATE"
     ? 5
     : 1);

    def unboundedGradeLevel = baseGradeLevel + creditPerAydUnit.intValue();

    def isAssociate = program.getProgramType() == "ASSOCIATE";

    def gradeLevelUb = (isAssociate
     ? 2
     : program.isUndergraduate()
     ? 5
     : 7);

    def gradeLevel = [unboundedGradeLevel, gradeLevelUb].min();

    log.debug("actualStartDate={}, referenceDate={}, totalPrevAcyUnits={}, totalCurAcyUnits={},
     totalCreditsCompletedBeforeProgramStartDate={}, totalCreditUnits={}, creditPerAydUnit={},
     baseGradeLevel={}, isAssociate={}, gradeLevelUb={}, gradeLevel={}",
     actualStartDate,
     referenceDate,
     totalPrevAcyUnits,
     totalCurAcyUnits,
     totalCreditsCompletedBeforeProgramStartDate,
     totalCreditUnits,
     creditPerAydUnit,
     baseGradeLevel,
     isAssociate,
     gradeLevelUb,
     gradeLevel
    );

    return gradeLevel;
```

isirAssumptionAutoCodeClearingLogic / isirCCodeAutoCodeClearingLogic / isirHighlightAutoCodeClearingLogic / isirRejectCodeAutoCodeClearingLogic / isirVerificationClearanceRules

Workbook: ISIR_ASSUMPT.csv / ISIR_C_CODES.csv / ISIR_HIGHLIGHT.csv / ISIR_REJECT_CODES.csv / ISIR_VERIFICATION_CODES.csv

Column: Auto_Code_Clearing_Logic

```
    //file:noinspection UnnecessaryQualifiedReference
    @groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.IsirVerificationClearanceRulesScript)
    package CHANGE_ME // this should be updated to your actual package name

    import com.oracle.sfp.scripting.api.*
```

```
import com.oracle.sfp.scripting.api.util.*

/*
Due to the changes to ISIR Verification for 24/25 we are no longer automatically triggering ISIR corrections
 within our
baseline configuration. An ISIR Verification task will be created once all acceptable documentation is
 received for a
Financial Aid Administrator to review. If Oracle determines that there is additional logic that can be added
 to increase
ISIR Verification Automation we will enhance our baseline configuration in future releases.
*/

//Note: if script returns true then auto code clear, if it returns false then create a task

//Student ISIR variables
def isirFPS_INCARCERATEDAPPLICANTFLAG =
 isirRecord.getField(IsirField.FPS_INCARCERATEDAPPLICANTFLAG).getOrNull()
def isirSTUDENT_FILED1040OR1040NR = isirRecord.getField(IsirField.STUDENT_FILED1040OR1040NR).getOrNull()
def isirSTUDENT_FILEDNONUSTAXRETURN = isirRecord.getField(IsirField.STUDENT_FILEDNONUSTAXRETURN).getOrNull()
def isirSTUDENTFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirSTUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirSTUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirSTUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.STUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirSTUDENT_IRAROLLOVER = isirRecord.getField(IsirField.STUDENT_IRAROLLOVER).getOrNull()
def isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.STUDENT_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()

//Parent ISIR variables
def isirPARENT_FILED1040OR1040NR = isirRecord.getField(IsirField.PARENT_FILED1040OR1040NR).getOrNull()
def isirPARENTFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirPARENT_FILEDNONUSTAXRETURN = isirRecord.getField(IsirField.PARENT_FILEDNONUSTAXRETURN).getOrNull()
def isirPARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirPARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirPARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.PARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirPARENT_IRAROLLOVER = isirRecord.getField(IsirField.PARENT_IRAROLLOVER).getOrNull()
def isirPARENT_UPDATEDFAMILYSIZE = isirRecord.getField(IsirField.PARENT_UPDATEDFAMILYSIZE).getOrNull()
def isirFPS_ASSUMEDPARENTFAMILYSIZE = isirRecord.getField(IsirField.FPS_ASSUMEDPARENTFAMILYSIZE).getOrNull()
def isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.PARENT_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()

//Parent Spouse ISIR variables
def isirPARENTSPOUSE_IRAROLLOVER = isirRecord.getField(IsirField.PARENTSPOUSE_IRAROLLOVER).getOrNull()
def isirPARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.PARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirPARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirPARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirPARENTSPOUSE_FILED1040OR1040NR =
 isirRecord.getField(IsirField.PARENTSPOUSE_FILED1040OR1040NR).getOrNull()
def isirPARENTSPOUSEFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirPARENTSPOUSE_FILEDNONUSTAXRETURN =
 isirRecord.getField(IsirField.PARENTSPOUSE_FILEDNONUSTAXRETURN).getOrNull()
def isirPARENTSPOUSE_TAXRETURNFILINGSTATUS =
 isirRecord.getField(IsirField.PARENTSPOUSE_TAXRETURNFILINGSTATUS).getOrNull()
def isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.PARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()
```

```
//Custom documents received variables
boolean IdentificationDocsReceived = false
boolean SOEPDocsReceived = false

//Custom documents received check
if ((receivedDocuments.hasDoc("DriversLicense", "Student") && receivedDocuments.get("DriversLicense",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("NonDriversLicenseID",
 "Student") && receivedDocuments.get("NonDriversLicenseID", "Student").isAcceptable()) ||
 (receivedDocuments.hasDoc("StateIssuedID", "Student") && receivedDocuments.get("StateIssuedID",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("Passport", "Student") &&
 receivedDocuments.get("Passport", "Student").isAcceptable())) {
 IdentificationDocsReceived = true
}
if ((receivedDocuments.hasDoc("SOEP-Campus", "Student") && receivedDocuments.get("SOEP-
Campus", "Student").isAcceptable()) || (receivedDocuments.hasDoc("SOEP-Notary", "Student") &&
 receivedDocuments.get("SOEP-Notary", "Student").isAcceptable())) {
 SOEPDocsReceived = true
}

//If student is incarcerated and has custom documents then exit the script
if ((isirFPS_INCARCERATEDAPPLICANTFLAG == "1" || isirFPS_INCARCERATEDAPPLICANTFLAG == "2" ||
 isirFPS_INCARCERATEDAPPLICANTFLAG == "3") && IdentificationDocsReceived && SOEPDocsReceived) {
 return true
}

//Student auto code clearing logic
if ((isirSTUDENT_FILED1040OR1040NR == "1" || isirSTUDENT_FILED1040OR1040NR == null) && [null, "203", "206",
 "212", "214"].contains(isirSTUDENTFTIM_IRSRESPONSECODE_FTI)) {
 if ((receivedDocuments.hasDoc("1040", "Student") && receivedDocuments.get("1040",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("1040x", "Student") &&
 receivedDocuments.get("1040x", "Student").isAcceptable()) ||
 (receivedDocuments.hasDoc("TaxReturnTranscript", "Student") && receivedDocuments.get("TaxReturnTranscript",
 "Student").isAcceptable())) {
 return false
 }
}
if (isirSTUDENT_FILED1040OR1040NR == "2" && isirSTUDENT_FILEDNONUSTAXRETURN == "1") {
 if (receivedDocuments.hasDoc("ForeignTaxTranscript", "Student") &&
 receivedDocuments.get("ForeignTaxTranscript", "Student").isAcceptable()) {
 return false
 }
}
if (isirSTUDENT_FILED1040OR1040NR == "2" && (isirSTUDENT_FILEDNONUSTAXRETURN == "2" ||
 isirSTUDENT_FILEDNONUSTAXRETURN == null)) {
 if ((receivedDocuments.hasDoc("NonFilingStatement", "Student") &&
 receivedDocuments.get("NonFilingStatement", "Student").isAcceptable()) ||
 (receivedDocuments.hasDoc("W2", "Student") && receivedDocuments.get("W2", "Student").isAcceptable())
 || (receivedDocuments.hasDoc("1099G", "Student") && receivedDocuments.get("1099G",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("SelfEmploymentStatement", "Student") &&
 receivedDocuments.get("SelfEmploymentStatement", "Student").isAcceptable())) {
 return false
 }
}

//Parent auto code clearing logic
if ((isirPARENT_FILED1040OR1040NR == "1" || isirPARENT_FILED1040OR1040NR == null) && [null, "203", "206",
 "212", "214"].contains(isirPARENTFTIM_IRSRESPONSECODE_FTI)) {
 if ((receivedDocuments.hasDoc("1040", "Parent") && receivedDocuments.get("1040", "Parent").isAcceptable())
 || (receivedDocuments.hasDoc("1040x", "Parent") && receivedDocuments.get("1040x",
 "Parent").isAcceptable()) || (receivedDocuments.hasDoc("TaxReturnTranscript", "Parent") &&
 receivedDocuments.get("TaxReturnTranscript", "Parent").isAcceptable())) {
 return false
 }
}
```

```
if ((isirPARENT_FILED1040OR1040NR == "1" || isirPARENT_FILED1040OR1040NR == null) &&
 isirPARENT_FILEDNONUSTAXRETURN == "1") {
 if ((receivedDocuments.hasDoc("1040", "Parent") && receivedDocuments.get("1040", "Parent").isAcceptable())
 || (receivedDocuments.hasDoc("1040x", "Parent") && receivedDocuments.get("1040x",
 "Parent").isAcceptable()) || (receivedDocuments.hasDoc("TaxReturnTranscript", "Parent") &&
 receivedDocuments.get("TaxReturnTranscript", "Parent").isAcceptable())) {
 return false
 }
}
if (isirPARENT_FILED1040OR1040NR == "2" && isirPARENT_FILEDNONUSTAXRETURN == "2") {
 if (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent").isAcceptable()) {
 return false
 }
}
if (isirPARENT_FILED1040OR1040NR == "2" && isirPARENT_FILEDNONUSTAXRETURN == "3") {
 if ((receivedDocuments.hasDoc("NonFilingStatement", "Parent") &&
 receivedDocuments.get("NonFilingStatement", "Parent").isAcceptable()) ||
 (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent").isAcceptable())) {
 return false
 }
}
if (isirPARENT_FILED1040OR1040NR == "2" && ["4", "5", "6"].contains(isirPARENT_FILEDNONUSTAXRETURN)) {
 if ((receivedDocuments.hasDoc("NonFilingStatement", "Parent") &&
 receivedDocuments.get("NonFilingStatement", "Parent").isAcceptable()) ||
 (receivedDocuments.hasDoc("W2", "Parent") && receivedDocuments.get("W2", "Parent").isAcceptable())
 || (receivedDocuments.hasDoc("1099G", "Parent") && receivedDocuments.get("1099G",
 "Parent").isAcceptable()) || (receivedDocuments.hasDoc("SelfEmploymentStatement", "Parent") &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent").isAcceptable())) {
 return false
 }
}

//Parent Spouse auto code clearing logic
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && (isirPARENTSPOUSE_FILED1040OR1040NR
 == "1" || isirPARENTSPOUSE_FILED1040OR1040NR == null) && [null, "203", "206", "212",
 "214"].contains(isirPARENTSPOUSEFTIM_IRSRESPONSECODE_FTI)) {
 if ((receivedDocuments.hasDoc("1040", "Parent Spouse") && receivedDocuments.get("1040",
 "Parent Spouse").isAcceptable()) || (receivedDocuments.hasDoc("1040x", "Parent
 Spouse") && receivedDocuments.get("1040x", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("TaxReturnTranscript", "Parent Spouse") &&
 receivedDocuments.get("TaxReturnTranscript", "Parent Spouse").isAcceptable())) {
 return false
 }
}
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && (isirPARENTSPOUSE_FILED1040OR1040NR == "1" ||
 isirPARENTSPOUSE_FILED1040OR1040NR == null) && isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "1") {
 if ((receivedDocuments.hasDoc("1040", "Parent Spouse") && receivedDocuments.get("1040",
 "Parent Spouse").isAcceptable()) || (receivedDocuments.hasDoc("1040x", "Parent
 Spouse") && receivedDocuments.get("1040x", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("TaxReturnTranscript", "Parent Spouse") &&
 receivedDocuments.get("TaxReturnTranscript", "Parent Spouse").isAcceptable())) {
 return false
 }
}
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" &&
 isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "2") {
 if (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent Spouse") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent Spouse").isAcceptable()) {
 return false
 }
}
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" &&
 isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "3") {
```

```
 if ((receivedDocuments.hasDoc("NonFilingStatement", "Parent Spouse") &&
 receivedDocuments.get("NonFilingStatement", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent Spouse") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent Spouse").isAcceptable())) {
 return false
 }
}
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" && ["4", "5",
 "6"].contains(isirPARENTSPOUSE_FILEDNONUSTAXRETURN)) {
 if ((receivedDocuments.hasDoc("NonFilingStatement", "Parent Spouse") &&
 receivedDocuments.get("NonFilingStatement", "Parent Spouse").isAcceptable()) ||
 (receivedDocuments.hasDoc("W2", "Parent Spouse") && receivedDocuments.get("W2",
 "Parent Spouse").isAcceptable()) || (receivedDocuments.hasDoc("1099G", "Parent
 Spouse") && receivedDocuments.get("1099G", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("SelfEmploymentStatement", "Parent Spouse") &&
 receivedDocuments.get("SelfEmploymentStatement", "Parent Spouse").isAcceptable())) {
 return false
 }
}

//Student rollover information auto code clearing logic
if (isirSTUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 || isirSTUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 ||
 isirSTUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS > 0 || isirSTUDENT_IRAROLLOVER > 0) {
 if (receivedDocuments.hasDoc("RolloverStatement", "Student") && receivedDocuments.get("RolloverStatement",
 "Student").isAcceptable()) {
 return false
 }
}

//Parent rollover information auto code clearing logic
if (isirPARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 || isirPARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 ||
 isirPARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS > 0 || isirPARENT_IRAROLLOVER > 0) {
 if (receivedDocuments.hasDoc("RolloverStatement", "Parent") && receivedDocuments.get("RolloverStatement",
 "Parent").isAcceptable()) {
 return false
 }
}

//Parent Spouse rollover information auto code clearing logic
if (isirPARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 ||
 isirPARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 || isirPARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS
 > 0 || isirPARENTSPOUSE_IRAROLLOVER > 0) {
 if (receivedDocuments.hasDoc("RolloverStatement", "Parent Spouse") &&
 receivedDocuments.get("RolloverStatement", "Parent Spouse").isAcceptable()) {
 return false
 }
}

//Student Foreign Income Exclusion auto code clearing logic
if (isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION > 0 ||
 isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 if ((receivedDocuments.hasDoc("1040", "Student") && receivedDocuments.get("1040",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("1040x", "Student") &&
 receivedDocuments.get("1040x", "Student").isAcceptable()) ||
 (receivedDocuments.hasDoc("TaxReturnTranscript", "Student") && receivedDocuments.get("TaxReturnTranscript",
 "Student").isAcceptable()) || (receivedDocuments.hasDoc("ForeignTaxTranscript", "Student") &&
 receivedDocuments.get("ForeignTaxTranscript", "Student").isAcceptable())) {
 return false
 }
}

//Parent Foreign Income Exclusion auto code clearing logic
if (isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION > 0 ||
 isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 if ((receivedDocuments.hasDoc("1040", "Parent") && receivedDocuments.get("1040", "Parent").isAcceptable())
 || (receivedDocuments.hasDoc("1040x", "Parent") && receivedDocuments.get("1040x",
 "Parent").isAcceptable()) || (receivedDocuments.hasDoc("TaxReturnTranscript",
 "Parent") && receivedDocuments.get("TaxReturnTranscript", "Parent").isAcceptable()) ||
 (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent").isAcceptable())) {
 return false
 }
}

//Parent Spouse Foreign Income Exclusion auto code clearing logic
if (isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION
 > 0 || isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 if ((receivedDocuments.hasDoc("1040", "Parent Spouse") && receivedDocuments.get("1040",
 "Parent Spouse").isAcceptable()) || (receivedDocuments.hasDoc("1040x", "Parent
 Spouse") && receivedDocuments.get("1040x", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("TaxReturnTranscript", "Parent Spouse") &&
 receivedDocuments.get("TaxReturnTranscript", "Parent Spouse").isAcceptable())
 || (receivedDocuments.hasDoc("ForeignTaxTranscript", "Parent Spouse") &&
 receivedDocuments.get("ForeignTaxTranscript", "Parent Spouse").isAcceptable())) {
 return false
 }
}

//Family size auto code clearing logic
if (isirPARENT_UPDATEDFAMILYSIZE != null || isirFPS_ASSUMEDPARENTFAMILYSIZE != null) {
 if (receivedDocuments.hasDoc("VW-Dep", "Student") && receivedDocuments.get("VW-Dep",
 "Student").isAcceptable()) {
 return false
 }
}

return true
```

isirAssumptionDocumentsRequired / isirCCodeDocumentsRequired / isirHighlightsDocumentsRequired / isirRejectCodeDocumentsRequired / isirVerificationDocumentsRequired

Workbook: ISIR_ASSUMPT.csv / ISIR_C_CODES.csv / ISIR_HIGHLIGHT.csv / ISIR_REJECT_CODES.csv / ISIR_VERIFICATION_CODES.csv

Column: Documents_Required

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.IsirVerificationDocumentsRequiredScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

//Student ISIR variables
def isirFPS_INCARCERATEDAPPLICANTFLAG =
 isirRecord.getField(IsirField.FPS_INCARCERATEDAPPLICANTFLAG).getOrNull()
def isirSTUDENT_FILED1040OR1040NR = isirRecord.getField(IsirField.STUDENT_FILED1040OR1040NR).getOrNull()
def isirSTUDENT_FILEDNONUSTAXRETURN = isirRecord.getField(IsirField.STUDENT_FILEDNONUSTAXRETURN).getOrNull()
def isirSTUDENTFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirSTUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirSTUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.STUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirSTUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.STUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirSTUDENT_IRAROLLOVER = isirRecord.getField(IsirField.STUDENT_IRAROLLOVER).getOrNull()
def isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.STUDENT_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()

//Parent ISIR variables
def isirPARENT_FILED1040OR1040NR = isirRecord.getField(IsirField.PARENT_FILED1040OR1040NR).getOrNull()
def isirPARENTFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirPARENT_FILEDNONUSTAXRETURN = isirRecord.getField(IsirField.PARENT_FILEDNONUSTAXRETURN).getOrNull()
def isirPARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirPARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.PARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirPARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.PARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirPARENT_IRAROLLOVER = isirRecord.getField(IsirField.PARENT_IRAROLLOVER).getOrNull()
def isirPARENT_UPDATEDFAMILYSIZE = isirRecord.getField(IsirField.PARENT_UPDATEDFAMILYSIZE).getOrNull()
def isirFPS_ASSUMEDPARENTFAMILYSIZE = isirRecord.getField(IsirField.FPS_ASSUMEDPARENTFAMILYSIZE).getOrNull()
def isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.PARENT_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()

//Parent Spouse ISIR variables
def isirPARENTSPOUSE_IRAROLLOVER = isirRecord.getField(IsirField.PARENTSPOUSE_IRAROLLOVER).getOrNull()
def isirPARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS =
 isirRecord.getField(IsirField.PARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS).getOrNull()
def isirPARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI).getOrNull()
def isirPARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI).getOrNull()
def isirPARENTSPOUSE_FILED1040OR1040NR =
 isirRecord.getField(IsirField.PARENTSPOUSE_FILED1040OR1040NR).getOrNull()
def isirPARENTSPOUSEFTIM_IRSRESPONSECODE_FTI =
 isirRecord.getField(IsirField.PARENTSPOUSEFTIM_IRSRESPONSECODE_FTI).getOrNull()
def isirPARENTSPOUSE_FILEDNONUSTAXRETURN =
 isirRecord.getField(IsirField.PARENTSPOUSE_FILEDNONUSTAXRETURN).getOrNull()
def isirPARENTSPOUSE_TAXRETURNFILINGSTATUS =
 isirRecord.getField(IsirField.PARENTSPOUSE_TAXRETURNFILINGSTATUS).getOrNull()
def isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION =
 isirRecord.getField(IsirField.PARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION).getOrNull()

//Custom documents request logic
//documentRequest.addAnyDocuments(["HSDiploma", "Student"],["HSTranscript", "Student"],
["GEDCert", "Student"],["GEDTranscript", "Student"],["StateHSEquivalencyCert", "Student"],
["SecondarySchoolLeavingCert", "Student"],["HomeSchoolCert", "Student"])
documentRequest.addAnyDocuments(["DriversLicense", "Student"],["NonDriversLicenseID", "Student"],
["StateIssuedID", "Student"],["Passport", "Student"])
documentRequest.addAnyDocuments(["SOEP-Campus", "Student"],["SOEP-Notary", "Student"])

//If student is incarcerated then exit the script
if (isirFPS_INCARCERATEDAPPLICANTFLAG == "1" || isirFPS_INCARCERATEDAPPLICANTFLAG == "2" ||
 isirFPS_INCARCERATEDAPPLICANTFLAG == "3") {
 return
}

//Student document request logic
if ((isirSTUDENT_FILED1040OR1040NR == "1" || isirSTUDENT_FILED1040OR1040NR == null) && [null, "203", "206",
 "212", "214"].contains(isirSTUDENTFTIM_IRSRESPONSECODE_FTI)) {
 documentRequest.addAnyDocuments(["1040", "Student"], ["1040x", "Student"], ["TaxReturnTranscript",
 "Student"])
}
else if (isirSTUDENT_FILED1040OR1040NR == "2" && isirSTUDENT_FILEDNONUSTAXRETURN == "1") {
 documentRequest.addDocument("ForeignTaxTranscript", "Student")
}
else if (isirSTUDENT_FILED1040OR1040NR == "2" && (isirSTUDENT_FILEDNONUSTAXRETURN == "2" ||
 isirSTUDENT_FILEDNONUSTAXRETURN == null)) {
 documentRequest.addAnyDocuments(["NonFilingStatement", "Student"], ["W2", "Student"], ["1099G", "Student"],
 ["SelfEmploymentStatement", "Student"])
```

```
}

//Parent document request logic
if ((isirPARENT_FILED1040OR1040NR == "1" || isirPARENT_FILED1040OR1040NR == null) && [null, "203", "206",
 "212", "214"].contains(isirPARENTFTIM_IRSRESPONSECODE_FTI)) {
 documentRequest.addAnyDocuments(["1040", "Parent"], ["1040x", "Parent"], ["TaxReturnTranscript", "Parent"])
}
else if ((isirPARENT_FILED1040OR1040NR == "1" || isirPARENT_FILED1040OR1040NR == null) &&
 isirPARENT_FILEDNONUSTAXRETURN == "1") {
 documentRequest.addAnyDocuments(["1040", "Parent"], ["1040x", "Parent"], ["TaxReturnTranscript", "Parent"])
}
else if (isirPARENT_FILED1040OR1040NR == "2" && isirPARENT_FILEDNONUSTAXRETURN == "2") {
 documentRequest.addDocument("ForeignTaxTranscript", "Parent")
}
else if (isirPARENT_FILED1040OR1040NR == "2" && isirPARENT_FILEDNONUSTAXRETURN == "3") {
 documentRequest.addAnyDocuments(["NonFilingStatement", "Parent"], ["ForeignTaxTranscript", "Parent"])
}
else if (isirPARENT_FILED1040OR1040NR == "2" && ["4", "5", "6"].contains(isirPARENT_FILEDNONUSTAXRETURN)) {
 documentRequest.addAnyDocuments(["NonFilingStatement", "Parent"], ["W2", "Parent"], ["1099G", "Parent"],
 ["SelfEmploymentStatement", "Parent"])
}

//Parent Spouse document request logic
if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && (isirPARENTSPOUSE_FILED1040OR1040NR
 == "1" || isirPARENTSPOUSE_FILED1040OR1040NR == null) && [null, "203", "206", "212",
 "214"].contains(isirPARENTSPOUSEFTIM_IRSRESPONSECODE_FTI)) {
 documentRequest.addAnyDocuments(["1040", "Parent Spouse"], ["1040x", "Parent Spouse"],
 ["TaxReturnTranscript", "Parent Spouse"])
}
else if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && (isirPARENTSPOUSE_FILED1040OR1040NR == "1" ||
 isirPARENTSPOUSE_FILED1040OR1040NR == null) && isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "1") {
 documentRequest.addAnyDocuments(["1040", "Parent Spouse"], ["1040x", "Parent Spouse"],
 ["TaxReturnTranscript", "Parent Spouse"])
}
else if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" &&
 isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "2") {
 documentRequest.addDocument("ForeignTaxTranscript", "Parent Spouse")
}
else if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" &&
 isirPARENTSPOUSE_FILEDNONUSTAXRETURN == "3") {
 documentRequest.addAnyDocuments(["NonFilingStatement", "Parent Spouse"], ["ForeignTaxTranscript", "Parent
 Spouse"])
}
else if (isirPARENTSPOUSE_TAXRETURNFILINGSTATUS == "3" && isirPARENTSPOUSE_FILED1040OR1040NR == "2" && ["4",
 "5", "6"].contains(isirPARENTSPOUSE_FILEDNONUSTAXRETURN)) {
 documentRequest.addAnyDocuments(["NonFilingStatement", "Parent Spouse"], ["W2", "Parent Spouse"], ["1099G",
 "Parent Spouse"], ["SelfEmploymentStatement", "Parent Spouse"])
}

//Student rollover information document request logic
if (isirSTUDENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 || isirSTUDENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 ||
 isirSTUDENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS > 0 || isirSTUDENT_IRAROLLOVER > 0) {
 documentRequest.addDocument("RolloverStatement", "Student")
}

//Parent rollover information document request logic
if (isirPARENTFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 || isirPARENTFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 ||
 isirPARENT_UNTAXEDPORTIONSOFIRADISTRIBUTIONS > 0 || isirPARENT_IRAROLLOVER > 0) {
 documentRequest.addDocument("RolloverStatement", "Parent")
}

//Parent Spouse rollover information document request logic
if (isirPARENTSPOUSEFTIM_UNTAXEDIRADISTRIBUTIONS_FTI > 0 ||
 isirPARENTSPOUSEFTIM_IRADEDUCTIBLEANDPAYMENTS_FTI > 0 || isirPARENTSPOUSE_UNTAXEDPORTIONSOFIRADISTRIBUTIONS
 > 0 || isirPARENTSPOUSE_IRAROLLOVER > 0) {
 documentRequest.addDocument("RolloverStatement", "Parent Spouse")
```

```
}

//Student Foreign Income Exclusion document request logic
if (isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION > 0 ||
 isirSTUDENT_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 documentRequest.addAnyDocuments(["1040", "Student"], ["1040x", "Student"], ["TaxReturnTranscript",
 "Student"], ["ForeignTaxTranscript", "Student"])
}

//Parent Foreign Income Exclusion document request logic
if (isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION > 0 ||
 isirPARENT_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 documentRequest.addAnyDocuments(["1040", "Parent"], ["1040x", "Parent"], ["TaxReturnTranscript", "Parent"],
 ["ForeignTaxTranscript", "Parent"])
}

//Parent Spouse Foreign Income Exclusion document request logic
if (isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION != null && (isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION
 > 0 || isirPARENTSPOUSE_FOREIGNEARNEDINCOMEEXCLUSION < 0)) {
 documentRequest.addAnyDocuments(["1040", "Parent Spouse"], ["1040x", "Parent Spouse"],
 ["TaxReturnTranscript", "Parent Spouse"], ["ForeignTaxTranscript", "Parent Spouse"])
}

//Family size document request logic
if (isirPARENT_UPDATEDFAMILYSIZE != null || isirFPS_ASSUMEDPARENTFAMILYSIZE != null) {
 documentRequest.addDocument("VW-Dep", "Student")
}

return
```

isirDiscrepancyEvaluation

Workbook: ISIR_DISCREP.csv

Column: Discrepancy_Evaluation_Script

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.IsirDiscrepancyEvaluationScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

//Variable definitions
def isirSTUDENT_MARITALSTATUS = isirRecord.getField(IsirField.STUDENT_MARITALSTATUS).getOrNull()

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1117") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1117").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1117")
 if (documentValue == "Single (Never married)" && isirSTUDENT_MARITALSTATUS != "1")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "1"
 return
 }
 if (documentValue == "Married (Not separated)" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
 if (documentValue == "Remarried" && isirSTUDENT_MARITALSTATUS != "3")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "3"
 return
 }
 if (documentValue == "Divorced" && isirSTUDENT_MARITALSTATUS != "4")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "4"
 return
 }
 if (documentValue == "Separated" && isirSTUDENT_MARITALSTATUS != "5")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "5"
 return
 }
 if (documentValue == "Widowed" && isirSTUDENT_MARITALSTATUS != "6")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "6"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1044") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1044").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1044")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1045") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1045").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1045")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1046") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1046").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1046")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1047") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1047").isAllWhitespace())
```

```
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1047")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1048") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1048").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1048")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1049") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1049").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1049")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1050") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1050").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1050")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1051") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1051").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1051")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1052") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1052").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1052")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}

if (receivedDocuments.hasDoc("VW-Ind","Student") && receivedDocuments.get("VW-Ind","Student").isAcceptable()
 && receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1053") != null && !
receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1053").isAllWhitespace())
{
 def documentValue = receivedDocuments.get("VW-Ind", "Student").getDocumentField("AC1053")
 if (documentValue == "Spouse" && isirSTUDENT_MARITALSTATUS != "2")
 {
 discrepancyEvaluationResult.hasDiscrepancy = true
 discrepancyEvaluationResult.correctedValue = "2"
 return
 }
}
```

isirUseSubsequentIsir

Workbook: ISIR_MGMT.csv

Column: Use_Subsequent_ISIR

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.IsirUseSubsequentIsirScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

def isirTXID_TRANSACTIONSOURCE = latestIsirRecord.getField(IsirField.TXID_TRANSACTIONSOURCE).getOrNull()

if (isirTXID_TRANSACTIONSOURCE != null) {
 return true
}
```

isirUsedInPackaging

Workbook: ISIR_MGMT.csv

Column: ISIR_Used_in_Packaging

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.IsirUsedInPackagingScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

//get all compatible ISIRs for the Award Year where no reject codes exist and a SAI exists
def awardYearISIRs =
 allAwardYearIsirs.forAwardYear(awy).getCompatibleWithPackaging().findAll{it.getRejectCodes().isEmpty() &&
 it.getField(IsirField.TXID_SAI).getOrNull() != null}

//if a Valid or Unverified ISIR exists, then return it
def validOrUnverifiedISIR = awardYearISIRs.find{["Valid", "Unverified"].contains(it.isirStatus)}
if (validOrUnverifiedISIR != null) {
 return validOrUnverifiedISIR
}
```

```
//if a Corrected ISIR exists, then return it if the ISIR had been previously Valid or Unverified
def previouslyValidOrUnverifiedCorrectedISIR = awardYearISIRs.find{it.isirStatus == "Corrected" && ["Valid",
 "Unverified"].contains(it.getIsirStatusBeforeCorrection())}
if (previouslyValidOrUnverifiedCorrectedISIR != null) {
 return previouslyValidOrUnverifiedCorrectedISIR
}

//if a Corrected ISIR exists, then return it
def correctedISIR = awardYearISIRs.find{it.isirStatus == "Corrected"}
if (correctedISIR != null) {
 return correctedISIR
}

//if a Pending ISIR exists, then return it
def pendingISIR = awardYearISIRs.find{it.isirStatus == "Pending"}
if (pendingISIR != null) {
 return pendingISIR
}

//if no ISIRs exist in the Award Year that fit the criteria above, then no ISIR will be returned
return null
```

letterCodeCriteria

Workbook: LETTER.csv

Column: Letter_Code_Criteria

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.LetterCodeCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

log.info("ProvisionalAidOfferLetter params={}", letterCode.params)

def letterCodeToReturn = "PO1"

//The letter event should only trigger for academic years where the ISIR award year is open
def academicYearNumber = letterCode.params.academicYearNumber.toInteger()
def academicYear = acys.getWithNumber(academicYearNumber)
def isAwardYearOpen = helper.awardYear.forPeriod(academicYear).find{awy->
 fafsaAwardYearInfoList.find{info->
 awy==info.getAwardYear()}?.isOpen()} != null
if (!isAwardYearOpen){
 log.info("ProvisionalAidOfferLetter return #1, isAwardYearOpen=false: null")
 return null
}
else{
 log.info("ProvisionalAidOfferLetter isAwardYearOpen=true")
}

//The letter event should be triggered for a student when they only have one FPO and it contains at least 1
 Projected or Estimated fund status in their academic year
def fundStatuses = financialPlan.getInAcademicYear(academicYearNumber).getFundStatuses()
log.info("ProvisionalAidOfferLetter fundStatuses={}", fundStatuses)
if ((fundStatuses.contains("ESTIMATED") || fundStatuses.contains("PROJECTED")) && financialPlan.getVersion()
 == 1){
 log.info("ProvisionalAidOfferLetter return #2: {}", letterCodeToReturn)
 return letterCodeToReturn
}

//Once all funds are in an Awarded status the letter event should no longer trigger for that academic year
```

```
def financialPlanInAcademicYear = financialPlan.getInAcademicYear(academicYearNumber)
if (financialPlanInAcademicYear.getFundStatuses().find{it != "AWARDED"}==null){
 log.info("ProvisionalAidOfferLetter fundStatuses, all funds awarded={}",
 financialPlanInAcademicYear.getFundStatuses())
 log.info("ProvisionalAidOfferLetter return #3: null")
 return null
}

//The letter event should be generated when there are updates from the prior version of the FPO V2
if (financialPlan.getVersion() > 1){
 def prevFinancialPlanInAcy = previousFinancialPlan.withVersion(financialPlan.getVersion() -
 1).getInAcademicYear(academicYearNumber)

 //The letter should generate if Cost of Attendance is updated from the prior version of the FPO V2
 if (prevFinancialPlanInAcy.getAcyCoa() != financialPlanInAcademicYear.getAcyCoa()){
 log.info("ProvisionalAidOfferLetter preAcyCoa={}, currentAcyCoa={}", prevFinancialPlanInAcy.getAcyCoa(),
 financialPlanInAcademicYear.getAcyCoa())
 log.info("ProvisionalAidOfferLetter return #4: {}", letterCodeToReturn)
 return letterCodeToReturn
 }

 //If current FPO has different funds in it than prior FPO, then generate letter
 def funds = financialPlanInAcademicYear.getFundCodes()
 def previousFunds = prevFinancialPlanInAcy.getFundCodes()
 log.info("ProvisionalAidOfferLetter previousFunds={}", previousFunds)
 log.info("ProvisionalAidOfferLetter funds={}", funds)
 if (funds != previousFunds) {
 log.info("ProvisionalAidOfferLetter return #5: {}", letterCodeToReturn)
 return letterCodeToReturn
 }

 //The letter should generate if a fund's status or award amount changes
 def differentFund = funds.find { fc ->
 def fund = financialPlanInAcademicYear.getFund(fc)
 def prevFund = prevFinancialPlanInAcy.getFund(fc)
 fund.getFundStatuses() != prevFund.getFundStatuses() || fund.getMaxAmount() != prevFund.getMaxAmount()
 }
 if (differentFund) {
 log.info("ProvisionalAidOfferLetter return #6: {}", letterCodeToReturn)
 return letterCodeToReturn
 }
}

log.info("ProvisionalAidOfferLetter return #7: null")
return null
```

nfrDisbursementDates

Workbook: NFR_ATTRIB.csv

Column: NFR_Fund_Disbursement_Dates

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.NfrDisbursementDatesScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

// Schedule disbursement 6 days after the period started.
return period.startDate.plusDays(6)
```

nfrEligibility

Workbook: NFR_ATTRIB.csv

Column: Fund_Eligibility

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.NfrEligibilityScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

return [new NfrEligibilityField("Is eligible", !nfrConfig.evalPeriods().empty)]
```

nfrMaximumProjectedAwardAmount

Workbook: NFR_ATTRIB.csv

Column: Fund_Maximum_Projected_Award_Amount

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.NfrMaximumProjectedAwardAmountScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

import org.joda.time.LocalDate

static BigDecimal nthIntPart(BigDecimal dividend, int divisor, int partIdx) {
 int truncatedDividend = dividend.intValue();
 BigDecimal fraction = dividend - truncatedDividend;
 int roundedAmount = (dividend / divisor).intValue()
 return partIdx < truncatedDividend % divisor
 ? roundedAmount + 1
 : partIdx == divisor - 1
 ? roundedAmount + fraction
 : roundedAmount
}

def now = LocalDate.now()

def condDebug = { NfrPeriod p, String msg ->
 if (p == period) {
 log.debug(msg)
 }
}

def loan = period.commonLineLoan
def periodsAssociatedToLoan = nfrConfig.evalPeriods().filter { p -> p.commonLineLoan.equals(loan) }
 .indexed(1)
int nbPeriods = periodsAssociatedToLoan.size()
def getLoanPeriodNumber = { NfrPeriod p ->
 periodsAssociatedToLoan.find { it.value == p }.key
}
def isAlreadyPackaged = { NfrPeriod p ->
 p.pairedTerm.acyNo < acyNo
}
def alreadyPackaged = financialPlan.filterAcademicYears { curAcyNo -> curAcyNo < acyNo }
 .getWithCommonLineUniqueId(loan.commonLineUniqueId)
 .reach();
def thisInfo = "for fund $fundCode for acy $acy.number in period $period.number using CLUID
 $loan.commonLineUniqueId"
```

```
def isDecrease = loan.requestedLoanAmount < loan.ppAmountTotal

def frozenValue = { NfrPeriod p ->
 int clPpNo = getLoanPeriodNumber(p)
 int periodIndex = clPpNo - 1

 if (periodIndex == -1) {
 condDebug(p, "Period index can't be found $thisInfo, periodsAssociatedToLoan=$periodsAssociatedToLoan")
 return 0.0
 }

 if (loan.recordStatus == "LOAN_TERMINATED") {
 condDebug(p, "Setting zero amount for terminated loan $thisInfo, periodIndex=$periodIndex, nbPeriods=
$nbPeriods")
 return 0.0
 }

 def thisTerm = p.pairedTerm
 if (thisTerm == null) {
 condDebug(p, "Setting zero amount for not found current term $thisInfo, periodIndex=$periodIndex,
 nbPeriods=$nbPeriods")
 return 0.0
 }

 if (thisTerm.isNotAttending()) {
 condDebug(p, "Setting zero amount for not attending term $thisInfo, periodIndex=$periodIndex, nbPeriods=
$nbPeriods")
 return 0.0
 }

 if (isAlreadyPackaged(p)) {
 condDebug(p, "Using value calculated in previous acy $thisInfo")
 return alreadyPackaged.getOverlappingWith(p).maxAmount
 }

 if (!isDecrease && loan.isDisbursed(clPpNo) && loan.endDate < now) {
 condDebug(p, "Using disbursed amount $thisInfo")
 return loan.getPpAmount(clPpNo)
 }
 return null;
}.memoize()

def isFrozen = { NfrPeriod p -> frozenValue(p) != null }

if (isFrozen(period)) {
 return frozenValue(period)
}

def allFrozen = periodsAssociatedToLoan.findAll { isFrozen(it.value) }
def frozenTotal = allFrozen.collect { frozenValue(it.value) }
 .inject(0.0, { a, b -> a + b })

def amountLeft = loan.requestedLoanAmount - frozenTotal;
def periodsLeft = periodsAssociatedToLoan - allFrozen

if (periodsLeft.isEmpty()) {
 log.debug("No periods left for $thisInfo")
 return 0
}

def acyScopedFundCodes = ["DISCOUNT"].toSet()
def acyScopedTotal = { int curAcyNo ->
 financialPlan.getInAcademicYear(curAcyNo).getFunds(acyScopedFundCodes).maxAmount
}.memoize();
def higherPriorityLoans = nfrConfig.evalPeriods()
```

```
 .takeWhile { p -> p.commonLineLoan != loan }
 .collect { p -> p.commonLineLoan }
 .unique()
def higherPriorityPeriods = nfrConfig.evalPeriods().filter { p ->
 higherPriorityLoans.contains(p.commonLineLoan) }

def orderedByNeed = periodsLeft.collectEntries([:] as LinkedHashMap<NfrPeriod, BigDecimal>, { clPpNo, p ->
 def curAcy = acys.getWithNumber(p.pairedTerm.acyNo)
 def curAcyNo = curAcy.number
 def termFundingSoFar = financialPlan.filterFunds { code -> !acyScopedFundCodes.contains(code) }
 .getOverlappingWith(p)
 .maxAmount
 def higherPriorityPeriodTotalInThisTerm = higherPriorityPeriods
 .getOverlappingWith(p)
 .collect(nfrConfig.&evalMaximumProjectedAwardAmount)
 .inject(0.0, { a, b -> a + b })
 def nbTerms = curAcy.terms.size()
 def pairedTermIdx = curAcy.terms.indexOf(p.pairedTerm)
 def acyScopedInThisTerm = nthIntPart(acyScopedTotal(curAcyNo), nbTerms, pairedTermIdx);
 def termCoa = coa.in(p.pairedTerm).activeCoa
 def termRemainingNeed = [termCoa - termFundingSoFar - higherPriorityPeriodTotalInThisTerm -
 acyScopedInThisTerm, 0].max()
 [(p): termRemainingNeed]
}).toSorted { it.value }

def computedFirst = orderedByNeed.takeWhile { it.key != period }
 .keySet();

def computedFirstTotal = computedFirst
 .collect(nfrConfig.&evalMaximumProjectedAwardAmount)
 .inject(0.0, { a, b -> a + b })

def noPeriodsLeftToCompute = orderedByNeed.size() - computedFirst.size();

def curMaxAmount = nthIntPart(amountLeft - computedFirstTotal, noPeriodsLeftToCompute, 0)

def amount = [curMaxAmount, orderedByNeed[period]].min();

log.debug("Setting amount for fund {} for acy {} period {} using CLUID {}, amount={}, nbPeriods={}",
 fundCode, acy.number, period.number, loan.commonLineUniqueId, amount, nbPeriods)

return amount
```

nfrPeriods

Workbook: NFR_ATTRIB.csv

Column: NFR_Periods

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.NfrPeriodsScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

// Replace this code with your code
def alternativeLoanProgramTypeCodes = ["091"]

static <T> Closure<T> lazy(Closure<T> init) {
 T value
 boolean isInitDue = true
 return {
 if (isInitDue) {
 isInitDue = false
```

```
  value = init()
  }
  value
  }
 }

 def cluidsDisbursedInOtherFunds = lazy {
  Set<ICommonLineUniqueId> otherFundsDisbursedCLUIDs = [];
  if (financialPlan.version > 1) {
  otherFundsDisbursedCLUIDs = previousFinancialPlan
  .withVersion(financialPlan.version - 1)
  .nfr
  .filterCommonLineUniqueIds { cluid -> cluid != null }
  .filterFunds { f -> f != fundCode && isDisbursed(f) }
  .commonLineUniqueIds
  }
  otherFundsDisbursedCLUIDs
 }

 /**
  * Indicate if the fund has already been disbursed across all academic years
  */
 def isDisbursed(String fundCode) {
  disbursements
  .getWithFundCode(fundCode)
  .getWithStatus("DISBURSED")
  .getTotalDisbursementAmount() > 0
 }

 /**
  * Get the list of terms that can be associated with the given loan.
  * All loan / term matching logic should happen here.
  */
 IAcademicTermPeriodsAPI getLoanTerms(IAcademicTermPeriodsAPI terms, ICommonLineLoanAPI loan) {
  terms.getOverlappingWith(loan)
 }

 /**
  * Get the list of terms that can be associated to the given loan across all academic years of the current
  package.
  */
 IAcademicTermPeriodsAPI getAllLoanTermsAcrossPackage(ICommonLineLoanAPI loan) {
  getLoanTerms(acys.getTerms(), loan)
 }

 /**
  * Indicate if the loan can be associated with the current academic year.
  */
 boolean hasTermsInCurrentAcademicYear(ICommonLineLoanAPI loan) {
  !getLoanTerms(acy.terms, loan).isEmpty()
 }

 return commonLineLoans
  .filter { loan ->
  alternativeLoanProgramTypeCodes.contains(loan.alternativeLoanProgramTypeCode) &&
  loan.getRequestedLoanAmount() > 0 &&
  loan.processingStatus != "COMMON_LINE_UNIQUE_ID_CHANGED" &&
  hasTermsInCurrentAcademicYear(loan) &&
  !cluidsDisbursedInOtherFunds().contains(loan.commonLineUniqueId)
  }
  .orderedBy(new OrderBy([{ it.startDate }, { it.commonLineUniqueId }]))
  .withIndex()
  .collectMany { loan, loanIndex ->
  def loanTerms = getAllLoanTermsAcrossPackage(loan)
  loanTerms.withIndex(1).collect { period, periodIndex ->
  int periodNumber = loanIndex * 10 + period.number;
  // should be:
  // periodNumber = loanIndex * 10 + periodIndex;
  log.debug("Associating fund {} for period {} starting on {} with CLUID {}",
  fundCode, periodNumber, period.startDate, loan.commonLineUniqueId)
  new NfrPeriod(periodNumber, period)
  .withCommonLineLoan(loan)
  .withPairedPeriod(period)
  }
  }
```

nonTermAcyMonths

Workbook: PKGSCHEDATTRIB.csv

Column: Non_Term_ACY_Months

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.NonTermAcyMonthsScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Non-Term Baseline ACY Months Calculation, return type int */

if (acy.getUnits() == null || program.getAydUnits() == null)
{
 log.debug("ACY MONTHS CALCULATION CONFIG: ERROR, null values passed from APIs");
 return 0;
}

log.debug("ACY MONTHS CALCULATION CONFIG: Academic Year Schedule Units = {}, Program AYD Units = {}",
 acy.getUnits(), program.getAydUnits());

if ((acy.getUnits() / program.getAydUnits()) >= 1)
{
 return 9;
}
else
{
 int returnValue = Math.round(((acy.getUnits()/program.getAydUnits()) * 9));
 return returnValue;
}
```

pellEnrollmentIntensity

Workbook: FAS_FUND_CONFIG.csv

Column: Pell_Enrollment_Intensity

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.PellEnrollmentIntensityScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*
import org.joda.time.LocalDate

def countedUnits = 0.0
def enrollmentIntensity = 0.0
def creditsRequiredForFullTimeEnrollment = 12
LocalDate today = LocalDate.now()
LocalDate censusDate
```

```
// if enrollment status is X return 0 (0%)
if (primaryProgram.getEnrollmentStatus() != null && primaryProgram.getEnrollmentStatus() == "X") {
 return 0
}

// if not term, return 1 (100%)
if (!program.isTerm()) {
 return 1
}

//term program logic
censusDate = term.getStartDate().plusDays(10)

//count credits and calculate enrollmentIntensity
for (def course: program.getCourses().getAssociatedTo(term)) {
 log.debug("PellEnrollmentIntensity script, COURSE IN LOOP: courseEndDate = {}, courseAraIndicator = {},
 courseSchedulingStatus = {}, courseUnitsToAdd = {}", course.getEndDate(), course.getFirstAraIndicator(),
 course.getSchedulingStatus(), course.getUnits())
 // If the course is withdrawn, only count the credits if the course was withdrawn after the census date. If
 the course is not withdrawn, count the course credits.
 if ((["WITHDRAWN"].contains(course.getSchedulingStatus()) && course.getLastDateOfAttendance() != null &&
 course.getLastDateOfAttendance() > censusDate) || course.getSchedulingStatus() != "WITHDRAWN") {
 countedUnits = countedUnits + course.getUnits()
 }
 enrollmentIntensity = countedUnits / creditsRequiredForFullTimeEnrollment
}

//census date logic
if (censusDate != null && today <= censusDate) {
 log.debug("PellEnrollmentIntensity script if statement final values: enrollmentIntensity = {},
 countedUnits = {}, creditsRequiredForFullTimeEnrollment = {}", enrollmentIntensity, countedUnits,
 creditsRequiredForFullTimeEnrollment)
 return enrollmentIntensity
}
else {

 //define previousFpo
 log.debug("PellEnrollmentIntensity script else statement begin")
 def previousFpo
 if (previousFinancialPlan.effectiveAt(censusDate) != null) {
 log.debug("PellEnrollmentIntensity script using census date previous FPO")
 previousFpo = previousFinancialPlan.effectiveAt(censusDate)
 } else if (financialPlan.getVersion() > 1) {
 log.debug("PellEnrollmentIntensity script using (FPO version - 1) previous FPO")
 previousFpo = previousFinancialPlan.withVersion(financialPlan.getVersion() - 1)
 }

 //previousFpo is not null logic
 if (previousFpo != null) {
 log.debug("PellEnrollmentIntensity script previousFpo is not null")
 def prevEnrollmentIntensities =
 previousFpo.getInAwardYear(awy).getOverlappingWith(term).getEnrollmentIntensities()
 if (prevEnrollmentIntensities.isEmpty()) {
 log.debug("PellEnrollmentIntensity script, prevEnrollmentIntensities is empty, final values:
 enrollmentIntensity = {}, countedUnits = {}, creditsRequiredForFullTimeEnrollment = {}",
 enrollmentIntensity, countedUnits, creditsRequiredForFullTimeEnrollment)
 return enrollmentIntensity
 }
 enrollmentIntensity = prevEnrollmentIntensities.get()
 log.debug("PellEnrollmentIntensity script, prevEnrollmentIntensities is NOT empty, final
 values: enrollmentIntensity = {}, countedUnits = {}, creditsRequiredForFullTimeEnrollment = {}",
 enrollmentIntensity, countedUnits, creditsRequiredForFullTimeEnrollment)
 }
 else {
 //previousFpo is null message
 log.debug("PellEnrollmentIntensity script previousFpo is null")
```

```
     }
    }

    return enrollmentIntensity
```

plusCreditDecisionMatchingCriteria

Workbook: FAS_FUND_CONFIG.csv

Column: Plus_Credit_Decision_Matching_Criteria

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.PlusCreditDecisionMatchingCriteriaScript)
package CHANGE_ME // this should be updated to your actual package name

// Define utility methods
static BigDecimal nthIntPart(BigDecimal dividend, int divisor, int partIdx) {
 int truncatedDividend = dividend.intValue()
 BigDecimal fraction = dividend - truncatedDividend
 int roundedAmount = (dividend / divisor).intValue()
 return partIdx < truncatedDividend % divisor
 ? roundedAmount + 1
 : partIdx == divisor - 1
 ? roundedAmount + fraction
 : roundedAmount
}

// Actual script
boolean applicableLoansExist

if (program.isTerm()) {
 // Acy-scoped fund codes the client may need. Setting it as empty works too but in this case
 // the script can be significantly simplified. As this script only applies to (G)PLUS adding
 // fund codes with lower priority than (G)PLUS will make no difference.
 def acyScopedFundCodes = ["DISCOUNT"] as Set<String>

 def acyScopedTotal =
 financialPlan.getInAcademicYear(acyNo).filterFunds(acyScopedFundCodes.&contains).maxAmount

 def termsInNeed = acy.terms.indexed().findAll { tidx, t ->
 def termFundingSoFar = financialPlan.filterFunds { code -> !acyScopedFundCodes.contains(code) }
 .getOverlappingWith(t)
 .maxAmount
 def acyScopedInThisTerm = nthIntPart(acyScopedTotal, acy.terms.size(), tidx)
 def termCoa = coa.in(t).activeCoa
 def termRemainingNeed = [termCoa - termFundingSoFar - acyScopedInThisTerm, 0].max()
 termRemainingNeed > 0
 }.values()

 applicableLoansExist = termsInNeed.any { t ->
 !plusLoans.getOverlappingWith(t).isEmpty()
 }
} else {
 applicableLoansExist = acy.loanPaymentPeriods.any { pp ->
 pp.status != "CANCELED" && !plusLoans.getOverlappingWith(pp).isEmpty()
 }
}

if (!applicableLoansExist && plusOverrides.isEmpty() && deniedPlusLoans.isEmpty()) {
 return [projectedPlusLoan]
} else {
 return plusLoans.getOverlappingWith(pp).collect { loan ->
 loan.withNoPriority() // removing default priority queue here and rebuilding it below:
 .addReversedPriority(disbursements.getInScope(scope).isDisbursed())
 .addPriority(loan.getLatestPackagePriority(acyNo))
 .addPriority(loan.getProcessDate())
 .addReversedPriority(loan.getAcceptedLoanAmount())
 }
}
```

summerTermType

Workbook: PKGSCHEDATTRIB.csv

Column: Summer_Term_Script

```
return "header";
```

termPaymentPeriodMonths

Workbook: PKGSCHEDATTRIB.csv

Column: Term_PP_Months

```
//file:noinspection UnnecessaryQualifiedReference
@groovy.transform.BaseScript(com.oracle.sfp.scripting.api.sdk.autocomplete.TermPaymentPeriodMonthsScript)
package CHANGE_ME // this should be updated to your actual package name

import com.oracle.sfp.scripting.api.*
import com.oracle.sfp.scripting.api.util.*

/* Term Baseline PP Months Calculation, return type BigDecimal */

import org.joda.time.LocalDate;

def returnValue = 1.0;

def programTermType = program.getTermType();

log.debug("PP MONTHS CALCULATION CONFIG: Program Term Type = {}", program.getTermType());

def safiTerm = program.getTerms().getOverlappingWith(term).get(0);

if (program.getTermType() == null)
{
 log.debug("PP MONTHS CALCULATION CONFIG: ERROR, null value passed from API");
 returnValue = 1.0;
}
else if (safiTerm.isSummer())
{
 returnValue = 2.0;
}
// Standard from this point on
else if (programTermType == "Semester")
{
 returnValue = 4.5;
}
else if (programTermType == "Trimester")
{
 returnValue = 3.0;
}
else if (programTermType == "Quarter")
{
 returnValue = 3.0;
}

return returnValue;
```

What is a group synchronization in Student Self-Service?

You can manually synchronize groups from the Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) integration within Student Self-Service portal. This updates the group listings and displays any changes, like if a group was removed from the OCI Cloud Console but is still linked to a role in Student Financial Aid.

Every night, a synchronization process automatically checks and validates the connections between groups and roles to ensure everything works correctly.

In this scenario, if a conflict arises, the role is highlighted in red within the Student Self-Service portal to alert administrators and prompt action. The SFA administrator can resolve the issue by:

- Mapping the role to an existing or new OCI IAM Group
- Leaving the role unmapped to an OCI IAM Group

If a user is assigned, only the affected role and attempts to log into the SFA administration interface before the issue is resolved, they will receive an error message indicating that an issue has occurred while signing in and to contact the financial aid office for assistance.

Related Topics

- Manage Users and Groups in OCI IAM

What SAIG files does Student Financial Aid import?

When a student or borrower completes documentation required by the US Department of Education, a response file is generated. When a response file is generated with a designation to be routed to the SAIG mailbox that Oracle Student Financial Aid Cloud Service (SFA) has access to, SFA imports the file from the destination point mailbox according to configured intervals. SFA recognizes and accepts each response file, matches it to a student record, removes any associated flags, and updates the student record.

When ED makes changes to any of the record file layouts, Oracle updates SFA to support the new file layout.

Here's the list of files imported into SFA.

FAFSA Processing System (FPS) Files

FPS is formerly known as Central Processing System (CPS).

CORRXXIN	Outbound	ISIR Corrections
IDAPXXOP	Inbound	Daily Electronic Application ISIRs
IDSAXXOP	Inbound	Daily ISIR

CORRXXIN	Outbound	ISIR Corrections
IGCOXXOP	Inbound	FPS/CPS Daily ISIRs – ISIRs generated by institutional corrections
IGSGXXOP	Inbound	FPS/CPS Pushed ISIRs – System-Generated
IGSAXXOP	Inbound	FPS/CPS Pushed ISIRs
COREXXOP	Inbound	ISIR Errors - Electronic Correction Errors
EAPRXXOP	Inbound	ISIR Errors - Electronic Application Errors (not a valid inbound file - do not support associated outbound file) Related outbound file is FAFSA Application Export Record Layout (EAPSXXIN), which is currently not supported.
SIGAXXOP	Inbound	ISIR Errors - Signature Record Errors (not a valid inbound file - do not support associated outbound file) Related outbound file is Signature Record Export Record Layout (SIGSXXIN), which is currently not supported.

National Student Loan Data System (NSLDS) Files (SFP only)

CORRXXIN	Outbound	ISIR Corrections
TRNINFIN	Outbound	TSM/FAH Batch Inform File
TRNINFOP	Inbound	TSM/FAH Error/Acknowledgment File
FAHEXTOP	Inbound	Financial Aid History File
TRALRTOP	Inbound	TSM Alert File
EXTC05	Inbound	Scheduled Completion Report File (NSLDS/ Direct Loans)
EXTDP1	Inbound	Scheduled Completion Report File (TEACH)

Common Origination and Disbursement (COD) Files (SFP only)

CORRXXIN	Outbound	ISIR Corrections
CRDLXXIN	Outbound	Export Common Record for Direct Loans
CRPNXXOP	Inbound	Master Promissory Note Response

CORRXXIN	Outbound	ISIR Corrections
CRECMYOP	Inbound	Entrance Counseling Response
CRSPXXOP	Inbound	PLUS Application acknowledgment
CRCOXXOP	Inbound	Credit Decision Override
CRCSXXOP	Inbound	Credit Status Response
PGLEXXOP	Inbound	Pell Lifetime Eligibility Used 20XX-20XX Report (CommanDelimited)
PGMRXXOP	Inbound	Pell Multiple Reporting Record (Fixed-length, Flat File Format)
PGVRXXOP	Inbound	Pell Grant Verification Status Report (Comma, Delimited)
CRACXXOP	Inbound	TEACH Grant Counseling acknowledgment
CRATXXOP	Inbound	ATS Note acknowledgment
COMRECOP	Inbound	Common Record for Pell and Teach awards
CRDLMYOP	Inbound	Common Record for Direct Loans
PGRCXXOP	Inbound	Pell Reconciliation Report
DSDFXXOP	Inbound	Direct Loan School Account Statement (Fixed-Length, Disbursement Level Loan Detail)
DSRFXXOP	Inbound	Direct Loan School Account Statement Disbursement Detail On-Demand (Date Range, Fixed Length)
THSMXXYYOP	Inbound	TEACH Grant School Account Statement (Monthly)
PGASXXOP	Inbound	Pell Grant Electronic Statement of Account (ESOA), YTD
CRIBXXOP	Inbound	Annual Student Loan acknowledgment (Informed Borrower)

What's the 2025-2026 baseline configuration for Student Financial Aid Documents?

2025-2026 Baseline Configuration for the Documents Configuration Workbook

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
1099G	1099G	Aid Year	No	Your 2023 1099-G is required to complete your financial aid verification process. Form 1099-G is requested to report unemployment compensation as well as any state or local income tax refunds you received that year.
Amended Tax Return Form 1040x	1040x	Aid Year	No	Your 2023 tax return is required to complete your financial aid verification process. The Amended US Individual Income Tax Return is used for taxpayers who needed to correct mistakes made on Tax Form 1040.
Foreign Tax Transcript	ForeignTaxTranscript	Aid Year	No	Your 2023 tax return is required to complete your financial aid verification process. A tax return transcript should be requested from the IRS, which shows most line items from your tax return (Form 1040, 1040A, or 1040EZ) as it was originally filed, including any accompanying forms and schedules. It does not reflect any changes you, your representative, or the IRS made after the return was filed.
High School Diploma Equivalency Statement	HighSchoolStatement	Aid Year	Yes	Your high school diploma equivalency statement is required to complete your financial aid verification process. A high school diploma equivalency statement should be used to verify secondary education if other document types are not available due to COVID 19.
Housing	Housing	Aid Year	No	Your housing status declaration is required to update your Financial Aid Cost of Attendance to align with your housing plans.
IRS Tax Extension Approval	IRSExtensionApproval	Aid Year	No	Your 2023 extension approval form is required

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				to complete your financial aid verification process. This form is used to verify the IRS's approval of an extension beyond the automatic six-month extension for the appropriate tax year.
IRS Tax Extension Form 4868	IRSForm4868	Aid Year	No	Your 2023 extension form is required to complete your financial aid verification process. IRS Extension Form 4868 is used for taxpayers who are not able to file their federal individual income tax return by the due date, and are not able to get an automatic 6-month extension of time to file.
IRS Tax Return 1040	1040	Aid Year	No	Your 2023 tax return is required to complete your financial aid verification process. Your 2023 Individual Income Tax Return is the annual income tax return filed by citizens or residents of the United States and must be include all signatures of the tax filers.
Non-filing Statement	NonFilingStatement	Aid Year	Yes	Your non-filing statement is required to complete your financial aid verification process. A Verification of Non-filing (VNF) should be requested from the IRS stating that you have not filed an IRS income tax return for the requested tax year. If you are not able to obtain a VNF from the IRS or relevant tax authority, you may submit a statement certifying your attempt. If you are a dependent student, you can provide a statement certifying your non filing status, and are not required to request a VNF from the IRS or relevant tax authority.
PJ Provisional ISIR Status	PJProvisional	Aid Year	Yes	Your ISIR was received with a Provisional SAI. A Professional Judgment is required to complete your financial aid process.

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
Professional Judgment Application	PJApp	Aid Year	Yes	A professional judgment application may be requested to allow a financial aid administrator to review your unique situation and adjust the cost of attendance or data in the FAFSA used to calculate your SAI.
Rollover Statement	RolloverStatement	Aid Year	No	Your 2023 statement is required to complete your financial aid verification process. Your FAFSA indicates that an IRS rollover was reported on your federal tax return for the specified tax year. Please confirm the amount of the IRS-authorized rollover amount reported on the return. For your reference, a rollover is described as the following: Untaxed portions of IRA distributions and portions are reported as lines 4a minus 4b on the 1040. Sometimes, these amounts are "rolled over" into another qualified IRA, pension, or annuity plan, so these rollover amounts are not actually received as untaxed income. The rollover amount is verified and subtracted from the untaxed IRA distribution amount or untaxed pension and annuity distribution amount, as applicable. The rollover amount cannot be a negative number.
SAP Appeals Request	SAPAppealRequest	Aid Year	Yes	A SAP appeal may be requested to allow a financial aid administrator to review your unique situation and adjust your recent satisfactory academic progress record for the latest period.
Self Employment Statement	SelfEmploymentStatement	Aid Year	No	A 2023 self employment statement is required to complete your financial aid verification process. If self-employed, please provide a signed statement with the amounts of your AGI and

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				US income taxes paid for the tax year.
Statement of Education Purpose - Campus	SOEP-Campus	Aid Year	No	Your statement is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this document, and request additional information if necessary. Students must sign a statement of educational purpose that certifies who you are and that the federal student aid that you may receive will only be used for educational purposes and for the cost of attending the school for this Aid Year. Please complete the Statement of Education Purpose letter you received in person at your school and present a valid, unexpired, government-issued photo identification (ID) such as a passport or a driver's license or other state-issued ID, then submit the signed document.
Statement of Education Purpose - Notary	SOEP-Notary	Aid Year	Yes	Your statement is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this document, and request additional information if necessary. Students must sign a statement of educational purpose that certifies who you are and that the federal

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				student aid that you may receive will only be used for educational purposes and for the cost of attending the school for this Aid Year. Please complete the Statement of Education Purpose letter you received in person at your school and present a valid, unexpired, government-issued photo identification (ID) such as a passport or a driver's license or other state-issued ID. If you are unable to appear at your school, you must go to a notary public and sign the statement of educational purpose, then submit the signed document, including the certification stamp from the notary.
Student Statement	StudentStatement	Aid Year	Yes	A student statement is required to complete your financial aid verification process. Please provide a signed statement with information about your specific situation to be reviewed by a financial aid advisor.
US Tax Return Transcript	TaxReturnTranscript	Aid Year	No	Your 2023 tax return is required to complete your financial aid verification process. A tax return transcript should be requested from the IRS, which shows most line items from your tax return (Form 1040) as it was originally filed, including any accompanying forms and schedules. It does not reflect any changes you, your representative, or the IRS made after the return was filed.
Verification Worksheet Independent	VW-Ind	Aid Year	No	Your verification worksheet is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				information you reported on your Free Application for Federal Student Aid (FAFSA) to this worksheet, and request additional information if necessary.
Verification Worksheet Dependent	VW-Dep	Aid Year	No	Your verification worksheet is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this worksheet, and request additional information if necessary.
W2	W2	Aid Year	No	Your 2023 W2 is required to complete your financial aid verification process. Form W-2's are provided to employees from every employer engaged in a trade or business who pays for work or a service, including non-cash payments of $600 or more for the year, from whom: Income, social security, or Medicare tax was withheld. Income tax would have been withheld if the employee had claimed no more than one withholding allowance or had not claimed exemption from withholding on Form W-4, Employee's Withholding Allowance Certificate.
Non-Driver's License Identification Card	NonDriversLicenseID	Expiration Date	No	Your identification card is required to complete your financial aid verification process. DMV offers identification cards to residents who need an official form of identification, but do not want or need a driver license.
Passport	Passport	Expiration Date	No	Your passport is required to complete your financial aid verification process.

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				A passport is an official document issued by a government, certifying the holder's identity and citizenship and entitling them to travel under its protection to and from foreign countries and should contain the holder's name, place and date of birth, an issuing agency and an expiration date.
State-issued Driver's License	DriversLicense	Expiration Date	No	Your driver's license is required to complete your financial aid verification process. A driver's license is an official document used for identification that permits a specific individual to operate one or more types of motorized vehicles, such as a motorcycle, car, truck, or bus on a public road.
State-issued Identification Card	StateIssuedID	Expiration Date	No	Your identification card is required to complete your financial aid verification process. DMV offers identification cards to residents who need an official form of identification, but do not want or need a driver license.
Death Certificate	DeathCertificate	Lifetime	Yes	Your death certificate is required to complete your financial aid verification process. The death certificate should be the official statement, signed by a physician, indicating the cause, date, and place of the person's death.
GED Certificate	GEDCert	Lifetime	No	Your GED Certificate is required to complete your financial aid verification process. The General Education Development/ General Education Diploma (GED) or High School Equivalency Certificate, shows that you have a level of knowledge equivalent to a high school graduate and is used to verify secondary education.

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
GED Transcript	GEDTranscript	Lifetime	No	Your GED Transcript is required to complete your financial aid verification process. General Education Development/ General Education Diploma (GED) transcripts provide a substantial verification of your academic history. Refer to the GED Testing Service to find specific directions for your individual state and follow the requirements to request a copy of your transcripts.
High School Diploma	HSDiploma	Lifetime	No	Your high school diploma is required to complete your financial aid verification process. An official high school diploma should be received upon successful graduation from high school and is used to verify secondary education.
High School Transcript	HSTranscript	Lifetime	No	Your high school transcript is required to complete your financial aid verification process. High school transcripts provide a substantial verification of your academic history. Refer to your high school to find specific directions and follow the requirements to request a copy of your transcripts.
Home School Certificate	HomeSchoolCert	Lifetime	No	Your home school certificate is required to complete your financial aid verification process. A home school certificate can be received upon successful completion of a home school program and is used to verify secondary education.
Legal Name Change Document	LegalNameChange	Lifetime	Yes	Your legal name change document is required to complete your financial aid verification process. The government-issued document evidencing your legal name change under federal or state law.
Marriage Certificate	MarriageCertificate	Lifetime	Yes	Your marriage certificate is required to complete your financial aid verification

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				process. An original or certified copy of your marriage certificate evidencing your legal name change under federal or state law.
Secondary School Leaving Certificate	SecondarySchoolLeavingCe	Lifetime	No	Your certificate is required to complete your financial aid verification process. The Secondary School Leaving Certificate is a certification obtained by a student on successful completion of an examination at the end of study at the secondary schooling level.
State High School Equivalency Certificate	StateHSEquivalencyCert	Lifetime	No	Your certificate is required to complete your financial aid verification process. For students who left high school before graduation may complete an examination to secure a high school equivalency credential and will
Identity Verification Method	IdentityVerificationMethod	Aid Year	No	Identity Verification Method

What's the 2025-2026 baseline configuration for Student Financial Aid Doc Metadata?

2025-2026 Baseline Configuration for the Doc Metadata Configuration Workbook

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
1040	AC1084	Tax Form Type	1	Enumeration 1040, Foreign Tax Return, U.S. Territory Tax Return	Yes	N/A
1040	AC1015	Adjusted Gross Income	2	Double	Yes	N/A
1040	AC1120	Income earned from work	3	N/A	Yes	N/A
1040	AC1016	Taxes Paid	4	Double	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
1040	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	N/A
1040	AC1019	IRA Deduction	6	Double	Yes	N/A
1040	AC1020	Tax-Exempt Interest	7	Double	Yes	N/A
1040	AC1021	Education Credits	8	Double	Yes	N/A
1040	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	N/A
1040	AZ1118	Filing Status	9	Enumeration Single, Married-Filed Joint Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents from Parent 2: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTransc
1040	AC1007	Signature Date	10	Date	Yes	N/A
1040x	AC1084	Tax Form Type	1	Enumeration 1040, 1040A, 1040EZ, 1040NR, Foreign Tax Return, U.S. Territory Tax Return	Yes	N/A
1040x	AC1015	Adjusted Gross Income	2	Double	Yes	N/A
1040x	AC1120	Income earned from work	3	Double	Yes	N/A
1040x	AC1016	Taxes Paid	4	Double	Yes	N/A
1040x	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
1040x	AC1019	IRA Deduction	6	Double	Yes	N/A
1040x	AC1020	Tax-Exempt Interest	7	Double	Yes	N/A
1040x	AC1021	Education Credits	8	Double	Yes	N/A
1040x	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	N/A
1040x	AC1118	Filing Status	9	Enumeration Single, Married-Filed Joint Return, Married-Filed Separate Return, Head of Household, Qualifying Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents from Parent 2: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTranscript)
1040x	AC1007	Signature Date	11	Date	Yes	N/A
1099G	AC1008	Social Security Number	1	String	Yes	N/A
1099G	AC1064	Payer's Federal Identification Number	2	String	Yes	N/A
1099G	AC1013	Box 1 Amount	3	Double	Yes	Sums all 1099G, W2 and Self Employed Statement Income amounts to determine if Student was required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
1099G	AC1014	Box 4 Amount	4	Double	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
1099G	AC1012	Tax Calendar Year	5	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023	Yes	N/A
DriversLicense	AC1001	First Name	1	String	Yes	N/A
DriversLicense	AC1002	Last Name	2	String	Yes	N/A
DriversLicense	AC1003	Date of Birth	3	Date	Yes	N/A
DriversLicense	AC1004	Gender	4	Enumeration Male, Female	Yes	N/A
DriversLicense	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
DriversLicense	AC1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico,	Yes	Value "Other" requires Manually Reviewed

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
DriversLicense	AC1070	ID Number	7	String	Yes	N/A
ForeignTaxTranscrip	AC1015	Adjusted Gross Income	1	Double	Yes	N/A
ForeignTaxTranscrip	AC1084	Tax Form Type	2	Enumeration 1040, 1040A,1040EZ, 1040NR,Foreign Tax Return,U.S. Territory Tax Return	Yes	N/A
ForeignTaxTranscrip	AC1122	Income earned from work	3	Double	Yes	N/A
ForeignTaxTranscrip	AC1016	Taxes Paid	4	Double	Yes	N/A
ForeignTaxTranscrip	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	N/A
ForeignTaxTranscrip	AC1018	Untaxed Pensions Total	6	Double	Yes	N/A
ForeignTaxTranscrip	AC1019	IRA Deduction	7	Double	Yes	N/A
ForeignTaxTranscrip	AC1020	Tax-Exempt Interest	8	Double	Yes	N/A
ForeignTaxTranscrip	AC1021	Education Credits	9	Double	Yes	N/A
ForeignTaxTranscrip	AC1123	Foreign income exempt from federal taxation	10	Double	N/A	N/A
ForeignTaxTranscrip	AC1118	Filing Status	11	Enumeration Single,Married-Filed Joint Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse:

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
						(TaxReturnTranscript OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents from Parent 2: (TaxReturnTranscript OR 1040x OR ForeignTaxTranscript)
GEDCert	AC1001	First Name	1	String	Yes	N/A
GEDCert	AC1002	Last Name	2	String	Yes	N/A
GEDCert	AC1069	City	3	String	Yes	N/A
GEDCert	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	N/A
GEDCert	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
GEDTranscript	AC1001	First Name	1	String	Yes	N/A
GEDTranscript	AC1002	Last Name	2	String	Yes	N/A
GEDTranscript	AC1069	City	3	String	Yes	N/A
GEDTranscript	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
GEDTranscript	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
HomeSchoolCert	AC1001	First Name	1	String	Yes	N/A
HomeSchoolCert	AC1002	Last Name	2	String	Yes	N/A
HomeSchoolCert	AC1069	City	3	String	Yes	N/A
HomeSchoolCert	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	N/A
HomeSchoolCert	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
HSDiploma	AC1001	First Name	1	String	Yes	N/A
HSDiploma	AC1002	Last Name	2	String	Yes	N/A
HSDiploma	AC1068	School Name	3	String	Yes	N/A
HSDiploma	AC1069	City	4	String	Yes	N/A
HSDiploma	AC1006	Graduation Date	6	Date	Yes	Future Graduation Date Requires Manual Review
HSTranscript	AC1001	First Name	1	String	Yes	N/A
HSTranscript	AC1002	Last Name	2	String	Yes	N/A
HSTranscript	AC1068	School Name	3	String	Yes	N/A
HSTranscript	AC1069	City	4	String	Yes	N/A
HSTranscript	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC,	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC		
HSTranscript	AC1006	Graduation Date	6	Date	Yes	Future Graduation Date Requires Manual Review
IRSExtensionApprov	AC1012	Tax Calendar Year	1	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023	Yes	N/A
IRSExtensionApprov	AC1007	Signature Date	2	Date	N/A	N/A
IRSForm4868	AC1007	Signature Date	1	Date	Yes	N/A
IRSForm4868	AC1012	Tax Calendar Year	1	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023	Yes	N/A
LegalNameChange	AC1095	Previous Last Name	1	String	Yes	N/A
LegalNameChange	AC1096	Current Last Name	2	String	Yes	N/A
MarriageCertificate	AC1095	Previous Last Name	1	String	Yes	N/A
MarriageCertificate	AC1096	Current Last Name	2	String	Yes	N/A
NonDriversLicenseI	AC1001	First Name	1	String	Yes	N/A
NonDriversLicenseI	AC1002	Last Name	2	String	Yes	N/A
NonDriversLicenseI	AC1003	Date of Birth	3	Date	Yes	N/A
NonDriversLicenseI	AC1004	Gender	4	Enumeration Male, Female	Yes	N/A
NonDriversLicenseI	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
NonDriversLicenseI	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan,	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
NonDriversLicenseI	AC1070	ID Number	7	String	Yes	N/A
NonFilingStatement	AC1007	Signature Date	1	Date	Yes	N/A
Passport	AC1001	First Name	1	String	Yes	N/A
Passport	AC1002	Last Name	2	String	Yes	N/A
Passport	AC1003	Date of Birth	3	Date	Yes	N/A
Passport	AC1004	Gender	4	Enumeration Male, Female	Yes	N/A
Passport	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
						Updated Document
Passport	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian	N/A	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				or Alaska Native Tribe, Other		
Passport	AC1070	ID Number	7	String	Yes	N/A
RolloverStatement	AC1076	Rollover Amount	1	Double	Yes	N/A
RolloverStatement	AC1007	Signature Date	2	Date	Yes	N/A
SecondarySchoolLe	AC1001	First Name	1	String	Yes	N/A
SecondarySchoolLe	AC1002	Last Name	2	String	Yes	N/A
SecondarySchoolLe	AC1069	City	3	String	Yes	N/A
SecondarySchoolLe	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	N/A
SecondarySchoolLe	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
SelfEmploymentSta	AC1015	Adjusted Gross Income	1	Double	Yes	Sums all Self Employed Statement, 1099G and W2 Income amounts to determine if Student was required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
SelfEmploymentSta	AC1016	Taxes Paid	2	Double	Yes	N/A
SelfEmploymentSta	AC1007	Signature Date	3	Date	Yes	N/A
SOEP-Campus	AC1001	First Name	1	String	Yes	N/A
SOEP-Campus	AC1002	Last Name	2	String	Yes	N/A
SOEP-Campus	AC1068	School Name	3	String	Yes	N/A
SOEP-Campus	AC1071	School Representative First Name	4	String	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
SOEP-Campus	AC1072	School Representative Last Name	5	String	Yes	N/A
SOEP-Campus	AC1007	Signature Date	6	Date	Yes	N/A
SOEP-Notary	AC1001	First Name	1	String	Yes	N/A
SOEP-Notary	AC1002	Last Name	2	String	Yes	N/A
SOEP-Notary	AC1068	School Name	3	String	Yes	N/A
SOEP-Notary	AC1073	Contains Notary Seal?	4	String Yes, No	Yes	N/A
SOEP-Notary	AC1074	Notary First Name	5	String	Yes	N/A
SOEP-Notary	AC1075	Notary Last Name	6	String	Yes	N/A
SOEP-Notary	AC1007	Signature Date	7	Date	Yes	N/A
StateHSEquivalency	AC1001	First Name	1	String	Yes	N/A
StateHSEquivalency	AC1002	Last Name	2	String	Yes	N/A
StateHSEquivalency	AC1069	City	3	String	Yes	N/A
StateHSEquivalency	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	N/A
StateHSEquivalency	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
StateIssuedID	AC1001	First Name	1	String	Yes	N/A
StateIssuedID	AC1002	Last Name	2	String	Yes	N/A
StateIssuedID	AC1003	Date of Birth	3	Date	Yes	N/A
StateIssuedID	AC1004	Gender	4	Enumeration Male, Female	Yes	N/A
StateIssuedID	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
StateIssuedID	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other	N/A	N/A
StateIssuedID	AC1070	ID Number	7	String	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
TaxReturnTranscript	AC1084	Tax Form Type	1	Enumeration 1040, Foreign Tax Return, U.S. Territory Tax Return	yes	N/A
TaxReturnTranscript	AC1015	Adjusted Gross Income	2	Double	Yes	N/A
TaxReturnTranscript	AC1120	Income earned from work	3	Double	Yes	N/A
TaxReturnTranscript	AC1016	Taxes Paid	4	Double	Yes	N/A
TaxReturnTranscript	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	N/A
TaxReturnTranscript	AC1019	IRA Deduction	6	Double	Yes	N/A
TaxReturnTranscript	AC1020	Tax-Exempt Interest	7	Double	Yes	N/A
TaxReturnTranscript	AC1021	Education Credits	8	Double	Yes	N/A
TaxReturnTranscript	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	N/A
TaxReturnTranscript	AC1065	Filing Status	10	Enumeration Single,Married-Filed Joint Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse	Yes	N/A
TaxReturnTranscript	AC1007	Signature Date	11	Date	No	N/A
VW-Dep	AC1119	Parent's Marital Status	1	Enumeration Never married, Unmarried living together,Married (not separated), Remarried, Divorced, Separated, Widowed	Yes	N/A
VW-Dep	AC1098	Parent's Number of Family Members	2	Integer	Yes	N/A
VW-Dep	AC1024	Household Member Name - 1	3	String	Yes	N/A
VW-Dep	AC1034	Household Member Age - 1	4	Integer	Yes	N/A
VW-Dep	AC1044	Household Member	5	Enumeration Aunt, Cousin, Daughter,	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
		Relationships to student - 1		Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Dep	AC1025	Household Member Name - 2	6	String	No	N/A
VW-Dep	AC1035	Household Member Age - 2	7	Integer	No	N/A
VW-Dep	AC1045	Household Member Relationships to student - 2	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	N/A
VW-Dep	AC1026	Household Member Name - 3	9	String	No	N/A
VW-Dep	AC1036	Household Member Age - 3	10	Integer	No	N/A
VW-Dep	AC1046	Household Member Relationships to student - 3	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	N/A
VW-Dep	AC1027	Household Member Name - 4	12	String	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Dep	AC1037	Household Member Age - 4	13	Integer	No	N/A
VW-Dep	AC1047	Household Member Relationships to student - 4	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	N/A
VW-Dep	AC1028	Household Member Name - 5	15	String	No	N/A
VW-Dep	AC1038	Household Member Age - 5	16	Integer	No	N/A
VW-Dep	AC1048	Household Member Relationships to student - 5	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	N/A
VW-Dep	AC1029	Household Member Name - 6	18	String	No	N/A
VW-Dep	AC1039	Household Member Age - 6	19	Integer	No	N/A
VW-Dep	AC1049	Household Member Relationships to student - 6	20	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent,	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Step-Sibling, Unborn Child, Uncle		
VW-Dep	AC1030	Household Member Name - 7	21	String	No	N/A
VW-Dep	AC1040	Household Member Age - 7	22	Integer	No	N/A
VW-Dep	AC1050	Household Member Relationships to student - 7	23	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Dep	AC1031	Household Member Name - 8	24	String	No	N/A
VW-Dep	AC1041	Household Member Age - 8	25	Integer	No	N/A
VW-Dep	AC1051	Household Member Relationships to student - 8	26	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Dep	AC1032	Household Member Name - 9	27	String	No	N/A
VW-Dep	AC1042	Household Member Age - 9	28	Integer	No	N/A
VW-Dep	AC1052	Household Member Relationships to student - 9	29	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew,	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Dep	AC1033	Household Member Name - 10	30	String	No	N/A
VW-Dep	AC1043	Household Member Age - 10	31	Integer	No	N/A
VW-Dep	AC1053	Household Member Relationships to student - 10	32	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Dep	AC1007	Signature Date	33	Date	Yes	N/A
VW-Dep	AC1113	Parent Signature Date	34	Date	Yes	If value does not equal ISIR."Parent 1 Last Name" or ISIR."Parent 2 Last Name", request one of (LegalNameChange, MarriageCertificate, DriversLicense, Passport, NonDriversLicenseID) from Parent 1
VW-Dep	AC1114	Parent Signature Last Name	35	String	Yes	N/A
VW-Ind	AC1117	Student's Marital Status	1	Enumeration Single (Never married),Married (Not separated), Remarried, Divorced, Separated, Widowed	Yes	N/A
VW-Ind	AC1022	Student's Number of Family Members	2	Integer	Yes	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Ind	AC1024	Household Member Name - 1	3	String	Yes	N/A
VW-Ind	AC1034	Household Member Age - 1	4	Integer	Yes	N/A
VW-Ind	AC1044	Household Member Relationships to student - 1	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	N/A
VW-Ind	AC1025	Household Member Name - 2	6	String	No	N/A
VW-Ind	AC1035	Household Member Age - 2	7	Integer	No	N/A
VW-Ind	AC1045	Household Member Relationships to student - 2	8	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1026	Household Member Name - 3	9	String	No	N/A
VW-Ind	AC1036	Household Member Age - 3	10	Integer	No	N/A
VW-Ind	AC1046	Household Member Relationships to student - 3	11	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Ind	AC1027	Household Member Name - 4	12	String	No	N/A
VW-Ind	AC1037	Household Member Age - 4	13	Integer	No	N/A
VW-Ind	AC1047	Household Member Relationships to student - 4	14	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1028	Household Member Name - 5	15	String	No	N/A
VW-Ind	AC1038	Household Member Age - 5	16	Integer	No	N/A
VW-Ind	AC1048	Household Member Relationships to student - 5	17	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1029	Household Member Name - 6	18	String	No	N/A
VW-Ind	AC1039	Household Member Age - 6	19	Integer	No	N/A
VW-Ind	AC1049	Household Member Relationships to student - 6	20	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Ind	AC1030	Household Member Name - 7	21	String	No	N/A
VW-Ind	AC1040	Household Member Age - 7	22	Integer	No	N/A
VW-Ind	AC1050	Household Member Relationships to student - 7	23	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1031	Household Member Name - 8	24	String	No	N/A
VW-Ind	AC1041	Household Member Age - 8	25	Integer	No	N/A
VW-Ind	AC1051	Household Member Relationships to student - 8	26	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1032	Household Member Name - 9	27	String	No	N/A
VW-Ind	AC1042	Household Member Age - 9	28	Integer	No	N/A

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Ind	AC1052	Household Member Relationships to student - 9	29	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1033	Household Member Name - 10	30	String	No	N/A
VW-Ind	AC1043	Household Member Age - 10	31	Integer	No	N/A
VW-Ind	AC1053	Household Member Relationships to student - 10	32	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	N/A
VW-Ind	AC1007	Signature Date	33	Date	Yes	If value does not equal ISIR.Last Name, request one of (LegalNameChange, MarriageCertificate, DriversLicense, Passport, NonDriversLicenseID)
VW-Ind	AC1094	Signature Last Name	34	String	Yes	N/A
W2	AC1008	Social Security Number	`	String	Yes	N/A
W2	AC1009	EIN	2	String	Yes	Sums all W2, Self Employed Statement and 1099G Income amounts to determine if Student was

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
						required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
W2	AC1010	Box 1 Amount	3	Double	Yes	N/A
W2	AC1011	Box 2 Amount	4	Double	Yes	N/A
W2	AC1085	Box 12a Code	5	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	N/A
W2	AC1089	Box 12a Amount	6	Double	No	N/A
W2	AC1086	Box 12b Code	7	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	N/A
W2	AC1090	Box 12b Amount	8	Double	No	N/A
W2	AC1087	Box 12c Code	9	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	N/A
W2	AC1091	Box 12c Amount	10	Double	No	N/A
W2	AC1088	Box 12d Code	11	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	N/A
W2	AC1092	Box 12d Amount	12	Double	No	N/A
W2	AC1012	Tax Calendar Year	13	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023	Yes	N/A
IdentityVerificationN	AC1126	N/A	1	Enumeration In-Person, Conference Call, Third Party Verification	N/A	N/A

What's the 2026-2027 baseline configuration for the Doc Metadata configuration workbook?

2026-2027 Baseline Configuration for the Doc Metadata Configuration Workbook

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
1040	AC1084	Tax Form Type	1	Enumeration 1040, Foreign Tax Return, U.S. Territory Tax Return	Yes	NA
1040	AC1015	Adjusted Gross Income	2	Double	Yes	NA
1040	AC1120	Income earned from work	3	NA	Yes	NA
1040	AC1016	Taxes Paid	4	Double	Yes	NA
1040	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	NA
1040	AC1019	IRA Deduction	6	Double	Yes	NA
1040	AC1020	Tax-Exempt Interest	7	Double	Yes	NA
1040	AC1021	Education Credits	8	Double	Yes	NA
1040	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	NA
1040	AZ1118	Filing Status	9	Enumeration Single, Married-Filed Joint Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
						from Parent 2: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTransc
1040	AC1007	Signature Date	10	Date	Yes	NA
1040x	AC1084	Tax Form Type	1	Enumeration 1040, 1040A, 1040EZ, 1040NR, Foreign Tax Return, U.S. Territory Tax Return	Yes	NA
1040x	AC1015	Adjusted Gross Income	2	Double	Yes	NA
1040x	AC1120	Income earned from work	3	Double	Yes	NA
1040x	AC1016	Taxes Paid	4	Double	Yes	NA
1040x	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	NA
1040x	AC1019	IRA Deduction	6	Double	Yes	NA
1040x	AC1020	Tax-Exempt Interest	7	Double	Yes	NA
1040x	AC1021	Education Credits	8	Double	Yes	NA
1040x	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	NA
1040x	AC1118	Filing Status	9	Enumeration Single, Married-Filed Joint Return, Married-Filed Separate Return, Head of Household, Qualifying Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse: (TaxReturnTranscript OR 1040 OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents from Parent 2: (TaxReturnTranscript OR 1040 OR

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
						1040x OR ForeignTaxTranscript)
1040x	AC1007	Signature Date	11	Date	Yes	NA
1099G	AC1008	Social Security Number	1	String	Yes	NA
1099G	AC1064	Payer's Federal Identification Number	2	String	Yes	NA
1099G	AC1013	Box 1 Amount	3	Double	Yes	Sums all 1099G, W2 and Self Employed Statement Income amounts to determine if Student was required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
1099G	AC1014	Box 4 Amount	4	Double	Yes	NA
1099G	AC1012	Tax Calendar Year	5	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024	Yes	NA
DriversLicense	AC1001	First Name	1	String	Yes	NA
DriversLicense	AC1002	Last Name	2	String	Yes	NA
DriversLicense	AC1003	Date of Birth	3	Date	Yes	NA
DriversLicense	AC1004	Gender	4	Enumeration Male, Female	Yes	NA
DriversLicense	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
DriversLicense	AC1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland,	Yes	Value "Other" requires Manually Reviewed

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
DriversLicense	AC1070	ID Number	7	String	Yes	NA
ForeignTaxTranscrip	AC1015	Adjusted Gross Income	1	Double	Yes	NA
ForeignTaxTranscrip	AC1084	Tax Form Type	2	Enumeration 1040, 1040A,1040EZ, 1040NR,Foreign Tax Return,U.S. Territory Tax Return	Yes	NA
ForeignTaxTranscrip	AC1122	Income earned from work	3	Double	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
ForeignTaxTranscrip	AC1016	Taxes Paid	4	Double	Yes	NA
ForeignTaxTranscrip	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	NA
ForeignTaxTranscrip	AC1018	Untaxed Pensions Total	6	Double	Yes	NA
ForeignTaxTranscrip	AC1019	IRA Deduction	7	Double	Yes	NA
ForeignTaxTranscrip	AC1020	Tax-Exempt Interest	8	Double	Yes	NA
ForeignTaxTranscrip	AC1021	Education Credits	9	Double	Yes	NA
ForeignTaxTranscrip	AC1123	Foreign income exempt from federal taxation	10	Double	NA	NA
ForeignTaxTranscrip	AC1118	Filing Status	11	Enumeration Single,Married-Filed Joint Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse	Yes	If doc received from Student and value = Married Filing Separately, request the following documents from Spouse: (TaxReturnTranscript OR 1040x OR ForeignTaxTranscript) If doc received from Parent 1 and value = Married Filing Separately, request the following documents from Parent 2: (TaxReturnTranscript OR 1040x OR ForeignTaxTranscript)
GEDCert	AC1001	First Name	1	String	Yes	NA
GEDCert	AC1002	Last Name	2	String	Yes	NA
GEDCert	AC1069	City	3	String	Yes	NA
GEDCert	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC,	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC		
GEDCert	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
GEDTranscript	AC1001	First Name	1	String	Yes	NA
GEDTranscript	AC1002	Last Name	2	String	Yes	NA
GEDTranscript	AC1069	City	3	String	Yes	NA
GEDTranscript	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	NA
GEDTranscript	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
HomeSchoolCert	AC1001	First Name	1	String	Yes	NA
HomeSchoolCert	AC1002	Last Name	2	String	Yes	NA
HomeSchoolCert	AC1069	City	3	String	Yes	NA
HomeSchoolCert	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	NA
HomeSchoolCert	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
HSDiploma	AC1001	First Name	1	String	Yes	NA
HSDiploma	AC1002	Last Name	2	String	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
HSDiploma	AC1068	School Name	3	String	Yes	NA
HSDiploma	AC1069	City	4	String	Yes	NA
HSDiploma	AC1006	Graduation Date	6	Date	Yes	Future Graduation Date Requires Manual Review
HSTranscript	AC1001	First Name	1	String	Yes	NA
HSTranscript	AC1002	Last Name	2	String	Yes	NA
HSTranscript	AC1068	School Name	3	String	Yes	NA
HSTranscript	AC1069	City	4	String	Yes	NA
HSTranscript	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	NA
HSTranscript	AC1006	Graduation Date	6	Date	Yes	Future Graduation Date Requires Manual Review
IRSExtensionApprov	AC1012	Tax Calendar Year	1	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024	Yes	NA
IRSExtensionApprov	AC1007	Signature Date	2	Date	NA	NA
IRSForm4868	AC1007	Signature Date	1	Date	Yes	NA
IRSForm4868	AC1012	Tax Calendar Year	1	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024	Yes	NA
LegalNameChange	AC1095	Previous Last Name	1	String	Yes	NA
LegalNameChange	AC1096	Current Last Name	2	String	Yes	NA
MarriageCertificate	AC1095	Previous Last Name	1	String	Yes	NA
MarriageCertificate	AC1096	Current Last Name	2	String	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
NonDriversLicenseI	AC1001	First Name	1	String	Yes	NA
NonDriversLicenseI	AC1002	Last Name	2	String	Yes	NA
NonDriversLicenseI	AC1003	Date of Birth	3	Date	Yes	NA
NonDriversLicenseI	AC1004	Gender	4	Enumeration Male, Female	Yes	NA
NonDriversLicenseI	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
NonDriversLicenseI	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
NonDriversLicenseI	AC1070	ID Number	7	String	Yes	NA
NonFilingStatement	AC1007	Signature Date	1	Date	Yes	NA
Passport	AC1001	First Name	1	String	Yes	NA
Passport	AC1002	Last Name	2	String	Yes	NA
Passport	AC1003	Date of Birth	3	Date	Yes	NA
Passport	AC1004	Gender	4	Enumeration Male, Female	Yes	NA
Passport	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
Passport	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota,	NA	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Tennessee, Texas, Utah, Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
Passport	AC1070	ID Number	7	String	Yes	NA
RolloverStatement	AC1076	Rollover Amount	1	Double	Yes	NA
RolloverStatement	AC1007	Signature Date	2	Date	Yes	NA
SecondarySchoolLe	AC1001	First Name	1	String	Yes	NA
SecondarySchoolLe	AC1002	Last Name	2	String	Yes	NA
SecondarySchoolLe	AC1069	City	3	String	Yes	NA
SecondarySchoolLe	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA, ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC	Yes	NA
SecondarySchoolLe	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
SelfEmploymentSta	AC1015	Adjusted Gross Income	1	Double	Yes	Sums all Self Employed Statement, 1099G and W2 Income amounts to determine if Student was required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
SelfEmploymentSta	AC1016	Taxes Paid	2	Double	Yes	NA
SelfEmploymentSta	AC1007	Signature Date	3	Date	Yes	NA
SOEP-Campus	AC1001	First Name	1	String	Yes	NA
SOEP-Campus	AC1002	Last Name	2	String	Yes	NA
SOEP-Campus	AC1068	School Name	3	String	Yes	NA
SOEP-Campus	AC1071	School Representative First Name	4	String	Yes	NA
SOEP-Campus	AC1072	School Representative Last Name	5	String	Yes	NA
SOEP-Campus	AC1007	Signature Date	6	Date	Yes	NA
SOEP-Notary	AC1001	First Name	1	String	Yes	NA
SOEP-Notary	AC1002	Last Name	2	String	Yes	NA
SOEP-Notary	AC1068	School Name	3	String	Yes	NA
SOEP-Notary	AC1073	Contains Notary Seal?	4	String Yes, No	Yes	NA
SOEP-Notary	AC1074	Notary First Name	5	String	Yes	NA
SOEP-Notary	AC1075	Notary Last Name	6	String	Yes	NA
SOEP-Notary	AC1007	Signature Date	7	Date	Yes	NA
StateHSEquivalency	AC1001	First Name	1	String	Yes	NA
StateHSEquivalency	AC1002	Last Name	2	String	Yes	NA
StateHSEquivalency	AC1069	City	3	String	Yes	NA
StateHSEquivalency	AC1067	State	4	Enumeration AK, AL, AR, AS, AZ, CA, CO, CT, DC, DE, FL, FM, GA, GU, HI, IA,	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				ID, IL, IN, KS, KY, LA, MA, MD, ME, MH, MI, MN, MO, MS, MT, NC, ND, NE, NH, NJ, NM, NV, NY, OH, OK, OR, PA, PR, RI, SC, SD, TN, TX, UT, VA, VI, VT, WA, WI, WV, WY, FC		
StateHSEquivalency	AC1006	Graduation Date	5	Date	Yes	Future Graduation Date Requires Manual Review
StateIssuedID	AC1001	First Name	1	String	Yes	NA
StateIssuedID	AC1002	Last Name	2	String	Yes	NA
StateIssuedID	AC1003	Date of Birth	3	Date	Yes	NA
StateIssuedID	AC1004	Gender	4	Enumeration Male, Female	Yes	NA
StateIssuedID	AC1005	Expiration Date	5	Date	Yes	Checks Expiration Date, Request Updated Document
StateIssuedID	AZ1066	Issuing Agency	6	Enumeration Alabama, Alaska, Arizona, Arkansas, California, Colorado, Connecticut, Delaware, Florida, Georgia, Hawaii, Idaho, Illinois, Indiana, Iowa, Kansas, Kentucky, Louisiana, Maine, Maryland, Massachusetts, Michigan, Minnesota, Mississippi, Missouri, Montana, Nebraska, Nevada, New Hampshire, New Jersey, New Mexico, New York, North Carolina, North Dakota, Ohio, Oklahoma, Oregon, Pennsylvania, Rhode Island, South Carolina, South Dakota, Tennessee, Texas, Utah,	NA	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Vermont, Virginia, Washington, West Virginia, Wisconsin, Wyoming, U.S. government, District of Columbia, The Commonwealth of Puerto Rico, American Samoa, Guam, The Virgin Islands, The Commonwealth of the Northern Mariana Islands, The Republic of the Marshall Islands, The Federated States of Micronesia, The Republic of Palau,Federally Recognized American Indian or Alaska Native Tribe, Other		
StateIssuedID	AC1070	ID Number	7	String	Yes	NA
TaxReturnTranscript	AC1084	Tax Form Type	1	Enumeration 1040, Foreign Tax Return, U.S. Territory Tax Return	yes	NA
TaxReturnTranscript	AC1015	Adjusted Gross Income	2	Double	Yes	NA
TaxReturnTranscript	AC1120	Income earned from work	3	Double	Yes	NA
TaxReturnTranscript	AC1016	Taxes Paid	4	Double	Yes	NA
TaxReturnTranscript	AC1017	Untaxed IRA Distribution and Pension total	5	Double	Yes	NA
TaxReturnTranscript	AC1019	IRA Deduction	6	Double	Yes	NA
TaxReturnTranscript	AC1020	Tax-Exempt Interest	7	Double	Yes	NA
TaxReturnTranscript	AC1021	Education Credits	8	Double	Yes	NA
TaxReturnTranscript	AC1121	Foreign income exempt from federal taxation	9	Double	Yes	NA
TaxReturnTranscript	AC1065	Filing Status	10	Enumeration Single,Married-Filed Joint	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Return,Married-Filed Separate Return,Head of Household,Qualifyir Surviving Spouse		
TaxReturnTranscript	AC1007	Signature Date	11	Date	No	NA
VW-Dep	AC1119	Parent's Marital Status	1	Enumeration Never married, Unmarried living together,Married (not separated), Remarried, Divorced, Separated, Widowed	Yes	NA
VW-Dep	AC1098	Parent's Number of Family Members	2	Integer	Yes	NA
VW-Dep	AC1024	Household Member Name - 1	3	String	Yes	NA
VW-Dep	AC1034	Household Member Age - 1	4	Integer	Yes	NA
VW-Dep	AC1044	Household Member Relationships to student - 1	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	NA
VW-Dep	AC1025	Household Member Name - 2	6	String	No	NA
VW-Dep	AC1035	Household Member Age - 2	7	Integer	No	NA
VW-Dep	AC1045	Household Member Relationships to student - 2	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent,	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Step-Sibling, Unborn Child, Uncle		
VW-Dep	AC1026	Household Member Name - 3	9	String	No	NA
VW-Dep	AC1036	Household Member Age - 3	10	Integer	No	NA
VW-Dep	AC1046	Household Member Relationships to student - 3	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	NA
VW-Dep	AC1027	Household Member Name - 4	12	String	No	NA
VW-Dep	AC1037	Household Member Age - 4	13	Integer	No	NA
VW-Dep	AC1047	Household Member Relationships to student - 4	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	NA
VW-Dep	AC1028	Household Member Name - 5	15	String	No	NA
VW-Dep	AC1038	Household Member Age - 5	16	Integer	No	NA
VW-Dep	AC1048	Household Member Relationships to student - 5	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law	Yes	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Dep	AC1029	Household Member Name - 6	18	String	No	NA
VW-Dep	AC1039	Household Member Age - 6	19	Integer	No	NA
VW-Dep	AC1049	Household Member Relationships to student - 6	20	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Dep	AC1030	Household Member Name - 7	21	String	No	NA
VW-Dep	AC1040	Household Member Age - 7	22	Integer	No	NA
VW-Dep	AC1050	Household Member Relationships to student - 7	23	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Dep	AC1031	Household Member Name - 8	24	String	No	NA
VW-Dep	AC1041	Household Member Age - 8	25	Integer	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Dep	AC1051	Household Member Relationships to student - 8	26	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Dep	AC1032	Household Member Name - 9	27	String	No	NA
VW-Dep	AC1042	Household Member Age - 9	28	Integer	No	NA
VW-Dep	AC1052	Household Member Relationships to student - 9	29	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Dep	AC1033	Household Member Name - 10	30	String	No	NA
VW-Dep	AC1043	Household Member Age - 10	31	Integer	No	NA
VW-Dep	AC1053	Household Member Relationships to student - 10	32	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Dep	AC1007	Signature Date	33	Date	Yes	NA
VW-Dep	AC1113	Parent Signature Date	34	Date	Yes	If value does not equal ISIR."Parent 1 Last Name" or ISIR."Parent 2 Last Name", request one of (LegalNameChange, MarriageCertificate, DriversLicense, Passport, NonDriversLicenseID) from Parent 1
VW-Dep	AC1114	Parent Signature Last Name	35	String	Yes	NA
VW-Ind	AC1117	Student's Marital Status	1	Enumeration Single (Never married),Married (Not separated), Remarried, Divorced, Separated, Widowed	Yes	NA
VW-Ind	AC1022	Student's Number of Family Members	2	Integer	Yes	NA
VW-Ind	AC1024	Household Member Name - 1	3	String	Yes	NA
VW-Ind	AC1034	Household Member Age - 1	4	Integer	Yes	NA
VW-Ind	AC1044	Household Member Relationships to student - 1	5	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	Yes	NA
VW-Ind	AC1025	Household Member Name - 2	6	String	No	NA
VW-Ind	AC1035	Household Member Age - 2	7	Integer	No	NA
VW-Ind	AC1045	Household Member	8	Enumeration Aunt, Cousin, Daughter,	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
		Relationships to student - 2		Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Ind	AC1026	Household Member Name - 3	9	String	No	NA
VW-Ind	AC1036	Household Member Age - 3	10	Integer	No	NA
VW-Ind	AC1046	Household Member Relationships to student - 3	11	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1027	Household Member Name - 4	12	String	No	NA
VW-Ind	AC1037	Household Member Age - 4	13	Integer	No	NA
VW-Ind	AC1047	Household Member Relationships to student - 4	14	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1028	Household Member Name - 5	15	String	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
VW-Ind	AC1038	Household Member Age - 5	16	Integer	No	NA
VW-Ind	AC1048	Household Member Relationships to student - 5	17	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1029	Household Member Name - 6	18	String	No	NA
VW-Ind	AC1039	Household Member Age - 6	19	Integer	No	NA
VW-Ind	AC1049	Household Member Relationships to student - 6	20	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1030	Household Member Name - 7	21	String	No	NA
VW-Ind	AC1040	Household Member Age - 7	22	Integer	No	NA
VW-Ind	AC1050	Household Member Relationships to student - 7	23	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling,	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Unborn Child, Uncle		
VW-Ind	AC1031	Household Member Name - 8	24	String	No	NA
VW-Ind	AC1041	Household Member Age - 8	25	Integer	No	NA
VW-Ind	AC1051	Household Member Relationships to student - 8	26	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1032	Household Member Name - 9	27	String	No	NA
VW-Ind	AC1042	Household Member Age - 9	28	Integer	No	NA
VW-Ind	AC1052	Household Member Relationships to student - 9	29	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent, Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle	No	NA
VW-Ind	AC1033	Household Member Name - 10	30	String	No	NA
VW-Ind	AC1043	Household Member Age - 10	31	Integer	No	NA
VW-Ind	AC1053	Household Member Relationships to student - 10	32	Enumeration Aunt, Cousin, Daughter, Fiance, God Child, Grandchild, Grandparent, In-law Parent, In-law Sibling, Nephew, Niece, Parent,	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Self, Sibling, Significant Other, Son, Spouse, Step-Child, Step-Parent, Step-Sibling, Unborn Child, Uncle		
VW-Ind	AC1007	Signature Date	33	Date	Yes	If value does not equal ISIR.Last Name, request one of (LegalNameChange, MarriageCertificate, DriversLicense, Passport, NonDriversLicenseID)
VW-Ind	AC1094	Signature Last Name	34	String	Yes	NA
W2	AC1008	Social Security Number	`	String	Yes	NA
W2	AC1009	EIN	2	String	Yes	Sums all W2, Self Employed Statement and 1099G Income amounts to determine if Student was required to file per Threshold amount rules. If Student is required to file supporting Docs are requested.
W2	AC1010	Box 1 Amount	3	Double	Yes	NA
W2	AC1011	Box 2 Amount	4	Double	Yes	NA
W2	AC1085	Box 12a Code	5	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	NA
W2	AC1089	Box 12a Amount	6	Double	No	NA
W2	AC1086	Box 12b Code	7	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	NA
W2	AC1090	Box 12b Amount	8	Double	No	NA
W2	AC1087	Box 12c Code	9	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W,	No	NA

Document Code	Attribute Code	Attribute Name	Display Order	Data Type	Required?	Additional Resolution Actions
				Y,Z,AA,BB,DD,EE, FF		
W2	AC1091	Box 12c Amount	10	Double	No	NA
W2	AC1088	Box 12d Code	11	Enumeration A,B, C,D,E,F,G,H,J,K,L, M,N,P,Q,R,S,T,V,W, Y,Z,AA,BB,DD,EE, FF	No	NA
W2	AC1092	Box 12d Amount	12	Double	No	NA
W2	AC1012	Tax Calendar Year	13	Enumeration 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021, 2022, 2023, 2024	Yes	NA
IdentityVerificationN	AC1126	NA	1	Enumeration In-Person, Conference Call, Third Party Verification	NA	NA

What's the 2026-2027 baseline configuration for the Documents configuration workbook?

2026-2027 Baseline Configuration for the Documents Configuration Workbook

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
1099G	1099G	Aid Year	No	Your 2024 1099-G is required to complete your financial aid verification process. Form 1099-G is requested to report unemployment compensation as well as any state or local income tax refunds you received that year.
Amended Tax Return Form 1040x	1040x	Aid Year	No	Your 2024 tax return is required to complete your financial aid verification process. The Amended US Individual Income Tax Return is used for taxpayers who needed to correct

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				mistakes made on Tax Form 1040.
Foreign Tax Transcript	ForeignTaxTranscript	Aid Year	No	Your 2024 tax return is required to complete your financial aid verification process. A tax return transcript should be requested from the IRS, which shows most line items from your tax return (Form 1040, 1040A, or 1040EZ) as it was originally filed, including any accompanying forms and schedules. It does not reflect any changes you, your representative, or the IRS made after the return was filed.
High School Diploma Equivalency Statement	HighSchoolStatement	Aid Year	Yes	Your high school diploma equivalency statement is required to complete your financial aid verification process. A high school diploma equivalency statement should be used to verify secondary education if other document types are not available due to COVID 19.
Housing	Housing	Aid Year	No	Your housing status declaration is required to update your Financial Aid Cost of Attendance to align with your housing plans.
IRS Tax Extension Approval	IRSExtensionApproval	Aid Year	No	Your 2024 extension approval form is required to complete your financial aid verification process. This form is used to verify the IRS's approval of an extension beyond the automatic six-month extension for the appropriate tax year.
IRS Tax Extension Form 4868	IRSForm4868	Aid Year	No	Your 2024 extension form is required to complete your financial aid verification process. IRS Extension Form 4868 is used for taxpayers who are not able to file their federal individual income tax return by the due date, and are not able to get an automatic 6-month extension of time to file.

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
IRS Tax Return 1040	1040	Aid Year	No	Your 2024 tax return is required to complete your financial aid verification process. Your 2024 Individual Income Tax Return is the annual income tax return filed by citizens or residents of the United States and must be include all signatures of the tax filers.
Non-filing Statement	NonFilingStatement	Aid Year	Yes	Your non-filing statement is required to complete your financial aid verification process. A Verification of Non-filing (VNF) should be requested from the IRS stating that you have not filed an IRS income tax return for the requested tax year. If you are not able to obtain a VNF from the IRS or relevant tax authority, you may submit a statement certifying your attempt. If you are a dependent student, you can provide a statement certifying your non filing status, and are not required to request a VNF from the IRS or relevant tax authority.
PJ Provisional ISIR Status	PJProvisional	Aid Year	Yes	Your ISIR was received with a Provisional SAI. A Professional Judgment is required to complete your financial aid process.
Professional Judgment Application	PJApp	Aid Year	Yes	A professional judgment application may be requested to allow a financial aid administrator to review your unique situation and adjust the cost of attendance or data in the FAFSA used to calculate your SAI.
Rollover Statement	RolloverStatement	Aid Year	No	Your 2024 statement is required to complete your financial aid verification process. Your FAFSA indicates that an IRS rollover was reported on your federal tax return for the specified tax year. Please confirm the amount of the IRS-authorized rollover amount reported

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				on the return. For your reference, a rollover is described as the following: Untaxed portions of IRA distributions and portions are reported as lines 4a minus 4b on the 1040. Sometimes, these amounts are "rolled over" into another qualified IRA, pension, or annuity plan, so these rollover amounts are not actually received as untaxed income. The rollover amount is verified and subtracted from the untaxed IRA distribution amount or untaxed pension and annuity distribution amount, as applicable. The rollover amount cannot be a negative number.
SAP Appeals Request	SAPAppealRequest	Aid Year	Yes	A SAP appeal may be requested to allow a financial aid administrator to review your unique situation and adjust your recent satisfactory academic progress record for the latest period.
Self Employment Statement	SelfEmploymentStatement	Aid Year	No	A 2024 self employment statement is required to complete your financial aid verification process. If self-employed, please provide a signed statement with the amounts of your AGI and US income taxes paid for the tax year.
Statement of Education Purpose - Campus	SOEP-Campus	Aid Year	No	Your statement is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this document, and request additional information if necessary. Students must sign a statement of educational purpose that certifies who you

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				are and that the federal student aid that you may receive will only be used for educational purposes and for the cost of attending the school for this Aid Year. Please complete the Statement of Education Purpose letter you received in person at your school and present a valid, unexpired, government-issued photo identification (ID) such as a passport or a driver's license or other state-issued ID, then submit the signed document.
Statement of Education Purpose - Notary	SOEP-Notary	Aid Year	Yes	Your statement is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this document, and request additional information if necessary. Students must sign a statement of educational purpose that certifies who you are and that the federal student aid that you may receive will only be used for educational purposes and for the cost of attending the school for this Aid Year. Please complete the Statement of Education Purpose letter you received in person at your school and present a valid, unexpired, government-issued photo identification (ID) such as a passport or a driver's license or other state-issued ID. If you are unable to appear at your school, you must go to a notary public and sign the statement of educational purpose, then submit the signed document, including the certification stamp from the notary.

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
Student Statement	StudentStatement	Aid Year	Yes	A student statement is required to complete your financial aid verification process. Please provide a signed statement with information about your specific situation to be reviewed by a financial aid advisor.
US Tax Return Transcript	TaxReturnTranscript	Aid Year	No	Your 2024 tax return is required to complete your financial aid verification process. A tax return transcript should be requested from the IRS, which shows most line items from your tax return (Form 1040) as it was originally filed, including any accompanying forms and schedules. It does not reflect any changes you, your representative, or the IRS made after the return was filed.
Verification Worksheet Independent	VW-Ind	Aid Year	No	Your verification worksheet is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this worksheet, and request additional information if necessary.
Verification Worksheet Dependent	VW-Dep	Aid Year	No	Your verification worksheet is required to complete your financial aid verification process. Your application for financial aid has been selected for verification. Verification is a process mandated by the US Department of Education that requires the school to compare the information you reported on your Free Application for Federal Student Aid (FAFSA) to this worksheet,

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				and request additional information if necessary.
W2	W2	Aid Year	No	Your 2024 W2 is required to complete your financial aid verification process. Form W2s are provided to employees from every employer engaged in a trade or business who pays for work or a service, including non-cash payments of $600 or more for the year, from whom: Income, social security, or Medicare tax was withheld. Income tax would have been withheld if the employee had claimed no more than one withholding allowance or had not claimed exemption from withholding on Form W-4, Employee's Withholding Allowance Certificate.
Non-Driver's License Identification Card	NonDriversLicenseID	Expiration Date	No	Your identification card is required to complete your financial aid verification process. DMV offers identification cards to residents who need an official form of identification, but do not want or need a driver license.
Passport	Passport	Expiration Date	No	Your passport is required to complete your financial aid verification process. A passport is an official document issued by a government, certifying the holder's identity and citizenship and entitling them to travel under its protection to and from foreign countries and should contain the holder's name, place and date of birth, an issuing agency and an expiration date.
State-issued Driver's License	DriversLicense	Expiration Date	No	Your driver's license is required to complete your financial aid verification process. A driver's license is an official document used for identification that permits a specific individual to operate one or more types of motorized vehicles,

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				such as a motorcycle, car, truck, or bus on a public road.
State-issued Identification Card	StateIssuedID	Expiration Date	No	Your identification card is required to complete your financial aid verification process. DMV offers identification cards to residents who need an official form of identification, but do not want or need a driver license.
Death Certificate	DeathCertificate	Lifetime	Yes	Your death certificate is required to complete your financial aid verification process. The death certificate should be the official statement, signed by a physician, indicating the cause, date, and place of the person's death.
GED Certificate	GEDCert	Lifetime	No	Your GED Certificate is required to complete your financial aid verification process. The General Education Development/ General Education Diploma (GED) or High School Equivalency Certificate, shows that you have a level of knowledge equivalent to a high school graduate and is used to verify secondary education.
GED Transcript	GEDTranscript	Lifetime	No	Your GED Transcript is required to complete your financial aid verification process. General Education Development/ General Education Diploma (GED) transcripts provide a substantial verification of your academic history. Refer to the GED Testing Service to find specific directions for your individual state and follow the requirements to request a copy of your transcripts.
High School Diploma	HSDiploma	Lifetime	No	Your high school diploma is required to complete your financial aid verification process. An official high school diploma should be received upon successful graduation from high

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				school and is used to verify secondary education.
High School Transcript	HSTranscript	Lifetime	No	Your high school transcript is required to complete your financial aid verification process. High school transcripts provide a substantial verification of your academic history. Refer to your high school to find specific directions and follow the requirements to request a copy of your transcripts.
Home School Certificate	HomeSchoolCert	Lifetime	No	Your home school certificate is required to complete your financial aid verification process. A home school certificate can be received upon successful completion of a home school program and is used to verify secondary education.
Legal Name Change Document	LegalNameChange	Lifetime	Yes	Your legal name change document is required to complete your financial aid verification process. The government-issued document evidencing your legal name change under federal or state law.
Marriage Certificate	MarriageCertificate	Lifetime	Yes	Your marriage certificate is required to complete your financial aid verification process. An original or certified copy of your marriage certificate evidencing your legal name change under federal or state law.
Secondary School Leaving Certificate	SecondarySchoolLeavingCe	Lifetime	No	Your certificate is required to complete your financial aid verification process. The Secondary School Leaving Certificate is a certification obtained by a student on successful completion of an examination at the end of study at the secondary schooling level.
State High School Equivalency Certificate	StateHSEquivalencyCert	Lifetime	No	Your certificate is required to complete your financial aid verification process. For students who left high school before

Document Name	Document Code	Active Document Dates	Manual Review Required	Document Request Message
				graduation may complete an examination to secure a high school equivalency credential and will
Identity Verification Method	IdentityVerificationMethod	Aid Year	No	Identity Verification Method

What's the 2026-2027 baseline configuration for the ISIR Discrepancy configuration workbook?

Note: For the following document codes, some of the tax return-related attribute names displayed in this reference aren't the full attribute names that are in DOCMETADATA.csv. The extended attribute names in DOCMETADATA.csv include additional information about where the information can be found on the appropriate tax form.

- 1040
- 1040X
- SelfEmploymentStatement
- TaxReturnTranscript

The baseline configuration standard logic is that for any document metadata field that has a coordinating ISIR field (for example, ISIR.Student's Date of Birth equal to DriversLicense.Date of Birth),where the document metadata value is different from the ISIR value, replace the ISIR value with the document metadata value. This triggers an ISIR correction.

2026-2027 Baseline Configuration for the ISIR Discrepancy Configuration Workbook

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
HS Diploma or E +A2:G65quivalent	STUDENT_ HIGHSCHOOLCOMI	HSDiploma	High School Diploma	N/A	N/A	If Doc Exists and is Acceptable and ISIR field HS Diploma or Equivalent is Not = 1, then Update ISIR field to 1
HS Diploma or Equivalent	STUDENT_ HIGHSCHOOLCOMI	HSTranscript	High School Transcript	N/A	N/A	If Doc Exists and is Acceptable

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						and ISIR field HS Diploma or Equivalent is Not = 1, then Update ISIR field to 1
HS Diploma or Equivalent	STUDENT_ HIGHSCHOOLCOMI	GEDCert	GED Certificate	N/A	N/A	If Doc Exists and is Acceptable and ISIR field HS Diploma or Equivalent is Not = 2, then Update ISIR field to 2
HS Diploma or Equivalent	STUDENT_ HIGHSCHOOLCOMI	GEDTranscript	GED Transcript	N/A	N/A	If Doc Exists and is Acceptable and ISIR field HS Diploma or Equivalent is Not = 2, then Update ISIR field to 2
HS Diploma or Equivalent	STUDENT_ HIGHSCHOOLCOMI	HomeSchoolCert	Home School Certificate	N/A	N/A	If Doc Exists and is Acceptable and ISIR field HS Diploma or Equivalent is Not = 3, then Update ISIR field to 3
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1093 AC1117	Student's Marital Status	Where Metadata value is different from ISIR value, replace with Metadata value.
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1044	Household Member Relationship(s) to student - 1	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1045	Household Member Relationship(s) to student - 2	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1046	Household Member Relationship(s) to student - 3	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1047	Household Member Relationship(s) to student - 4	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						"2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1048	Household Member Relationship(s) to student - 5	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1049	Household Member Relationship(s) to student - 6	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1050	Household Member Relationship(s) to student - 7	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1051	Household Member Relationship(s) to student - 8	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1052	Household Member Relationship(s) to student - 9	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Marital Status	STUDENT_ MARITALSTATUS	VW-Ind	Verification Worksheet Independent	AC1053	Household Member Relationship(s) to student - 10	If Meta Data Value = Spouse and ISIR value not 2, updated ISIR to "2 = Married/ remarried"
Student's Adjusted Gross Income from IRS form	STUDENT_ ADJUSTEDGROSSIN	SelfEmploymentSta	Self Employment Statement	N/A	Adjusted Gross Income (Line 11- 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Adjusted Gross Income from IRS form	STUDENT_ ADJUSTEDGROSSIN	TaxReturnTranscript	US Tax Return Transcript	N/A	Adjusted Gross Income (Line 11-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Adjusted Gross Income from IRS form	STUDENT_ ADJUSTEDGROSSIN	1040	IRS Tax Return 1040	N/A	Adjusted Gross Income (Line 11-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
Student's Adjusted Gross Income from IRS form	STUDENT_ ADJUSTEDGROSSIN	1040x	US Tax Return Amended	N/A	Adjusted Gross Income (Line 11-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Adjusted Gross Income from IRS form	STUDENT_ ADJUSTEDGROSSIN	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Adjusted Gross Income	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Educational Credits	STUDENT_ EDUCATIONCREDIT	TaxReturnTranscrip	US Tax Return Transcript	N/A	Education Credits (Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Educational Credits	STUDENT_ EDUCATIONCREDIT	1040	IRS Tax Return 1040	N/A	Education Credits (Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Educational Credits	STUDENT_ EDUCATIONCREDIT	1040x	US Tax Return Amended	N/A	Education Credits ((Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						this ISIR field we will enhance our baseline configuration in future releases.
Student's Educational Credits	STUDENT_ EDUCATIONCREDIT	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Education Credits	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Interest Income	STUDENT_ TAXEXEMPTINTERE	TaxReturnTranscrip	US Tax Return Transcript	N/A	Tax-Exempt Interest (Line 2a - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Interest Income	STUDENT_ TAXEXEMPTINTERE	1040	IRS Tax Return 1040	N/A	Tax-Exempt Interest (Line 2a - 1040)	Due to the ISIR field related to this discrepancy being related to an

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Interest Income	STUDENT_ TAXEXEMPTINTERE	1040x	US Tax Return Amended	N/A	Tax-Exempt Interest (Line 2a - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Interest Income	STUDENT_ TAXEXEMPTINTERE	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Tax-Exempt Interest	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Untaxed Portions of IRA Distributions and Pensions	STUDENT_ UNTAXEDPORTION	TaxReturnTranscript	US Tax Return Transcript	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Untaxed Portions of IRA Distributions and Pensions	STUDENT_ UNTAXEDPORTION	1040	IRS Tax Return 1040	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						configuration in future releases.
Student's Untaxed Portions of IRA Distributions and Pensions	STUDENT_ UNTAXEDPORTION	1040x	US Tax Return Amended	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Untaxed Portions of IRA Distributions and Pensions	STUDENT_ UNTAXEDPORTION	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Untaxed IRA Distribution and Pension total	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Number of Family Members	STUDENT_ UPDATEDFAMILYSIZ	VW-Ind	Verification Worksheet Independent	AC1022	Student's Number of Family Members	Where Metadata value is different from ISIR value, replace with Metadata value.

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
Student's Tax Return Completed	STUDENT_ FILED1040OR1040N	W2 1099G SelfEmployment Statement IRSForm4868 NonFilingStatement	W2 1099G Self Employment Statement Extension Form IRS Form 4868 Non-filing Student Statement	N/A	N/A	IF TaxReturnTranscript, 1040x, ForeignTaxTranscript received && ISIR value = BLANK,3,2 Set ISIR Correction = 1 IF IRSForm4868 received && ISIR Value = BLANK,3 Set ISIR Correction = 2 IF NonFilingStatement received && threshold not exceeded && ISIR Value = BLANK Set ISIR Correction = 3
Student's Tax Return Filing Status	STUDENT_ TAXRETURNFILING	TaxReturnTranscript 1040 1040x ForeignTaxTranscrip	US Tax Return Transcript IRS Tax Return 1040 Amended Tax Return Form 1040x Foreign Tax Transcript	N/A	Filing Status	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Type of Tax Form Used?	STUDENT_ FILED1040OR1040N	TaxReturnTranscript 1040 1040x ForeignTaxTranscrip	US Tax Return Transcript IRS Tax Return 1040 Amended Tax Return Form 1040x Foreign Tax Transcript	AC1084	Tax Form Type	Where Metadata value is different from ISIR value, Replace with Metadata value.
Student's Untaxed Pensions	STUDENT_ UNTAXEDPORTION	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Untaxed Pensions Total	Due to the ISIR field related to

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's U.S. Income Tax Paid	STUDENT_ INCOMETAXPAID	SelfEmploymentSta	Self Employment Statement	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's U.S. Income Tax Paid	STUDENT_ INCOMETAXPAID	TaxReturnTranscript	US Tax Return Transcript	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's U.S. Income Tax Paid	STUDENT_ INCOMETAXPAID	1040	IRS Tax Return 1040	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's U.S. Income Tax Paid	STUDENT_ INCOMETAXPAID	1040x	US Tax Return Amended	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						our baseline configuration in future releases.
Student's U.S. Income Tax Paid	STUDENT_ INCOMETAXPAID	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Taxes Paid	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parent's Marital Status	PARENT_ MARITALSTATUS	VW-Dep	Verification Worksheet Dependent	AC1097 AC1119	Parent's Marital Status	Where Metadata value is different from ISIR value, replace with Metadata value.
Parents' Number of Family Members	PARENT_ UPDATEDFAMILYSIZ	VW-Dep	Verification Worksheet Dependent	AC1098	Parent's Number of Family Members	Where Metadata value is different from ISIR value, replace with Metadata value.
Parents' Adjusted Gross Income from IRS form	PARENT_ ADJUSTEDGROSSIN	SelfEmploymentSta	Self Employment Statement	N/A	Adjusted Gross Income (Line 11- 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						this ISIR field we will enhance our baseline configuration in future releases.
Parents' Adjusted Gross Income from IRS form	PARENT_ ADJUSTEDGROSSIN	TaxReturnTranscript	US Tax Return Transcript	N/A	Adjusted Gross Income (Line 11- 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Adjusted Gross Income from IRS form	PARENT_ ADJUSTEDGROSSIN	1040	IRS Tax Return 1040	N/A	Adjusted Gross Income (Line 11- 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Adjusted Gross Income from IRS form	PARENT_ ADJUSTEDGROSSIN	1040x	US Tax Return Amended	N/A	Adjusted Gross Income (Line 11- 1040)	Due to the ISIR field related to this discrepancy being related to an

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Adjusted Gross Income from IRS form	PARENT_ ADJUSTEDGROSSIN	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Adjusted Gross Income	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Exemptions Claimed	PARENT_ TAXEXEMPTINTERE	N/A	N/A	N/A	N/A	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Educational Credits	PARENT_ EDUCATIONCREDIT	TaxReturnTranscript	US Tax Return Transcript	N/A	Education Credits (Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Educational Credits	PARENT_ EDUCATIONCREDIT	1040	IRS Tax Return 1040	N/A	Education Credits (Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						configuration in future releases.
Parents' Educational Credits	PARENT_ EDUCATIONCREDIT	1040x	US Tax Return Amended	N/A	Education Credits (Line 3 of Schedule 3-1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Educational Credits	PARENT_ EDUCATIONCREDIT	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Education Credits	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Untaxed Portions of IRA Distributions and Pensions	PARENT_ UNTAXEDPORTION	TaxReturnTranscrip	US Tax Return Transcript	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Untaxed Portions of IRA Distributions and Pensions	PARENT_ UNTAXEDPORTION	1040	IRS Tax Return 1040	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Untaxed Portions of IRA Distributions and Pensions	PARENT_ UNTAXEDPORTION	1040x	US Tax Return Amended	N/A	Untaxed IRA Distribution and Pension total (Lines (4a + 4c) minus (4b + 4d) - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Untaxed Portions of IRA Distributions and Pensions	PARENT_ UNTAXEDPORTION	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Untaxed IRA Distribution and Pension total	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' IRA Payments	PARENT_ DEDUCTIBLEPAYME	TaxReturnTranscrip	US Tax Return Transcript	N/A	IRA Deduction (Lines 15 + 19 of Schedule 1 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
Parents' IRA Payments	PARENT_ DEDUCTIBLEPAYME	1040	IRS Tax Return 1040	N/A	IRA Deduction (Lines 15 + 19 of Schedule 1 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' IRA Payments	PARENT_ DEDUCTIBLEPAYME	1040x	US Tax Return Amended	N/A	IRA Deduction (Lines 15 + 19 of Schedule 1 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' IRA Payments	PARENT_ DEDUCTIBLEPAYME	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	IRA Deduction	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Tax Return Filing Status	PARENT_ FILEDNONUSTAXRE	TaxReturnTranscript 1040 1040x ForeignTaxTranscrip	US Tax Return Transcript IRS Tax Return 1040 Amended Tax Return Form 1040x Foreign Tax Transcript	N/A	Filing Status	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' Type of Tax Form Used?	PARENT_ FILED1040OR1040N	TaxReturnTranscript 1040 1040x ForeignTaxTranscrip	US Tax Return Transcript IRS Tax Return 1040 Amended Tax Return Form 1040x Foreign Tax Transcript	AC1084	Tax Form Type	Where Metadata value is different from ISIR value, replace with Metadata value.
Parents' Untaxed Pensions	PARENT_ UNTAXEDPORTION	ForeignTaxTranscrip	Foreign Tax Transcript	AC1018	N/A	Due to the ISIR field related to this discrepancy being related to an FTI value within

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' U.S. Income Tax Paid	PARENT_ INCOMETAXPAID	SelfEmploymentSta	Self Employment Statement	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' U.S. Income Tax Paid	PARENT_ INCOMETAXPAID	TaxReturnTranscript	US Tax Return Transcript	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
						discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' U.S. Income Tax Paid	PARENT_ INCOMETAXPAID	1040	IRS Tax Return 1040	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Parents' U.S. Income Tax Paid	PARENT_ INCOMETAXPAID	1040x	US Tax Return Amended	N/A	Taxes Paid (Line 22 minus Schedule 2, Line 2 - 1040)	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.

Tech Reference ISIR Field	SFA ISIR Field Reference	Document Code	Document Name	Attribute Code	Attribute Name / Meta Data Field	ISIR Discrepancy Rules
Parents' U.S. Income Tax Paid	PARENT_ INCOMETAXPAID	ForeignTaxTranscrip	Foreign Tax Transcript	N/A	Taxes Paid	Due to the ISIR field related to this discrepancy being related to an FTI value within the 25/26 ISIR we are no longer automatically triggering this ISIR discrepancy within our baseline configuration. If Oracle determines that there is additional discrepancy logic that can be added for the manually entered field related to this ISIR field we will enhance our baseline configuration in future releases.
Student's Permanent State	STUDENT_STATE	N/A	N/A	N/A	N/A	IF ISIR.PERMANENTSTATE == "FC" & SFP.ADDRESS_ STATE != "FC" Trigger discrepancy Trigger manual review