SIRIS Technical Training

Prepared by
Nate Trost
Senior Programmer Analyst
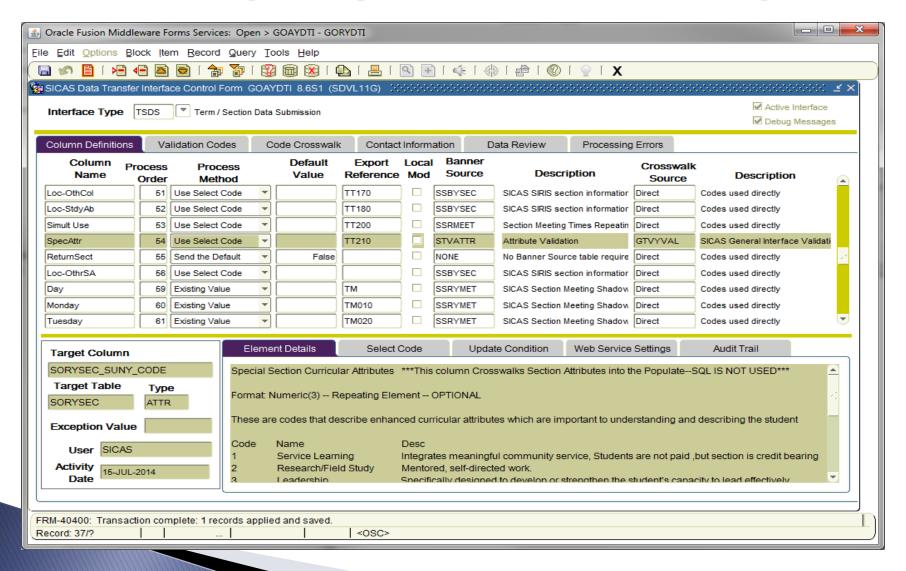
SIRIS Processing Flow--Overview

▸ All SIRIS Interfaces use a similar process flow to collect and transmit the data required.

◦ Insert Key Data values (called the Primary Key) into <u>all rows</u> that will be sent; called the Population. Any elements that are provided for key values can not be altered and have any effect—they are never run.

◦ Process the SQLs on the XOAYDTI form for the interface to assign data to non-primary key columns. Changes made to these locally are respected and run after the population is done.

◦ If running in transmit mode then rows are looped through and XML is generated and sent to System.

SIRIS Table Processing—More Details

- IF FZ_XDS_BuildSomething('BEFORE_THING')=Y
- THEN
  - Insert Key Data values (the Primary Key) into <u>all rows</u> that may be sent to System Admin.
  - FZ_XDS_BuildSomething('AFTER_THING')
- Process the SQLs on the XOAYDTI form for the interface to assign data to non-primary key columns.
- FZ_XDS_BuildSomething('THING_LOOP');
- FZ_XDS_BuildSomething('AFTER_PROCESSING')
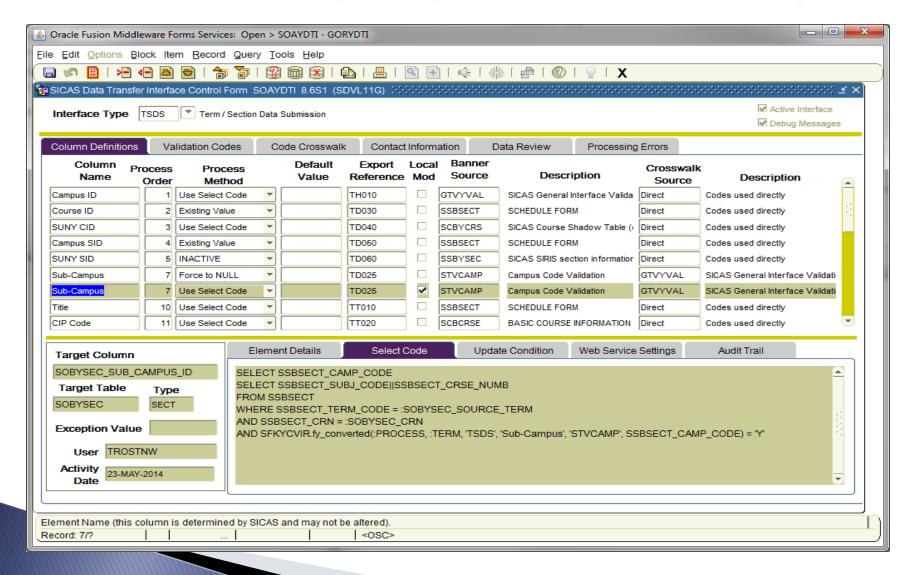- If Transmit Mode then build and send XML.

Primary Key Column Example

Primary Key Column Example

- Changes to this column may prevent the internal process from collecting the Population but certainly <u>will not</u> allow you to alter the set.
- Changes to the Primary Key must be done in application code by a programmer.
- No end user options exist for altering these values—contact your programming staff.
- Use FZ BEFORE_THING pass to take over the population routine; Return N to bypass SICAS.
- Use FZ AFTER_THING pass to "fix" SICAS work.
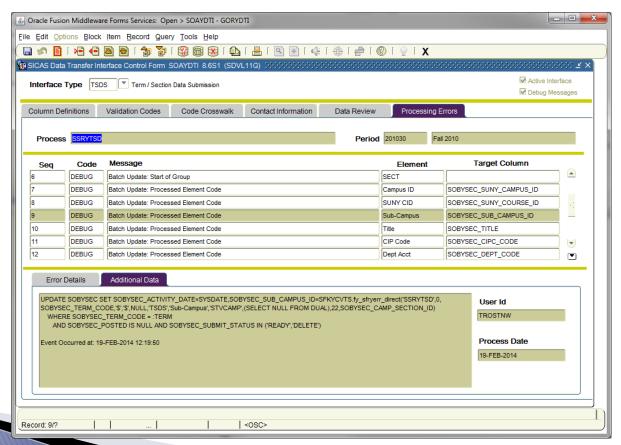
Non-Primary Key Column Example
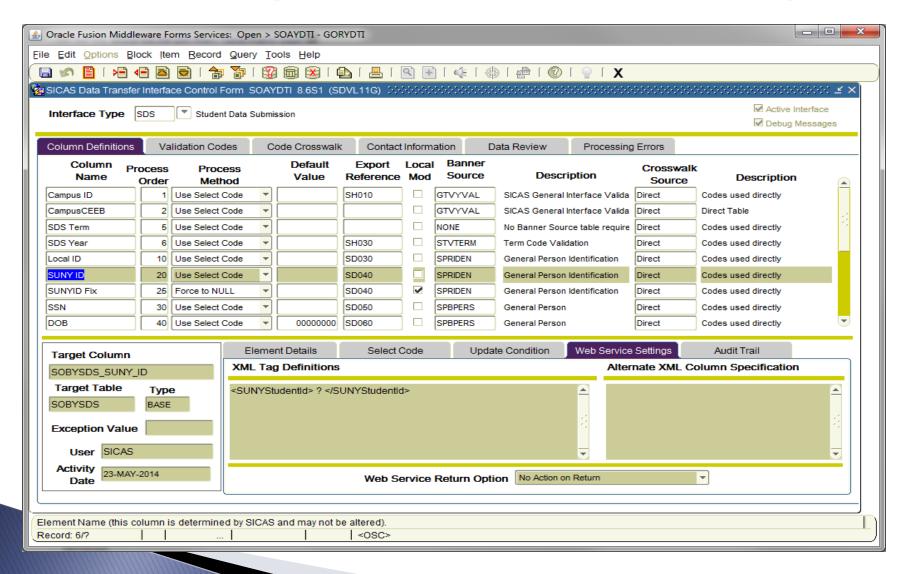
Non-Primary Key Column Example

- These rows provide SQL code to place values into NON-PRIMARY KEY columns.
- This is an end user tool but you may wish to involve your Programmer or SICAS in making changes that are more complex then process method changes.
- The method for changing these values:
  - Select the row you wish to alter.
  - Using the Record Menu: *Insert* and then *Duplicate*.
  - Alter the resulting Local Mod Record.
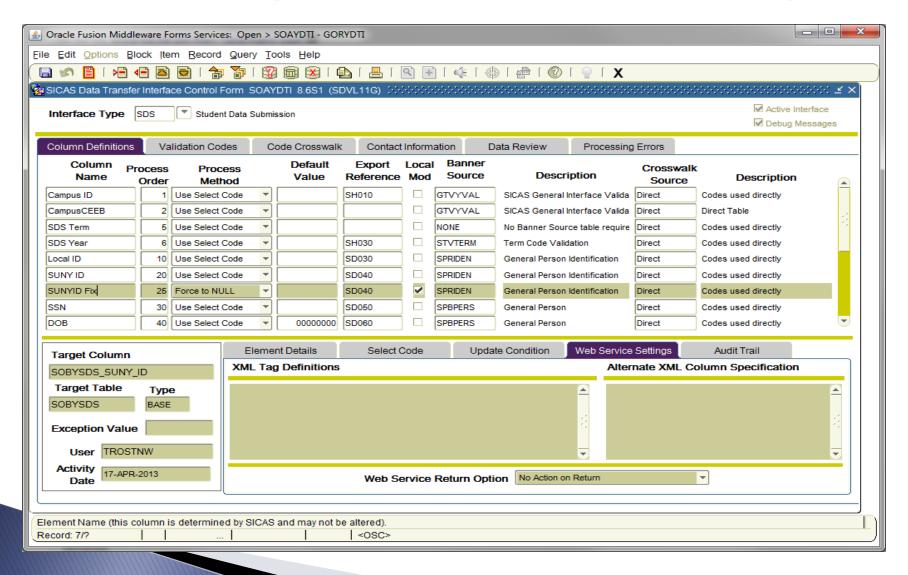 - Your Local Mod Record **REPLACES** the SICAS record.
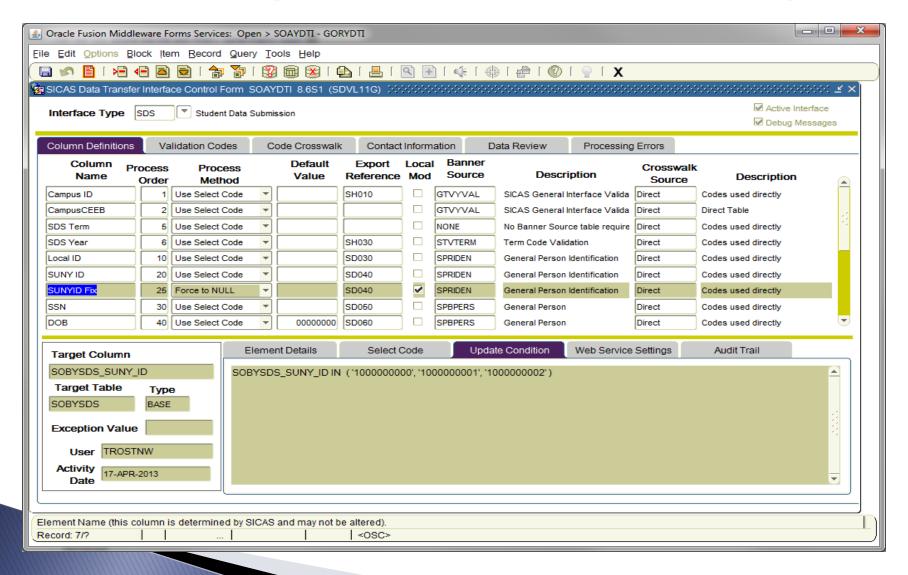
How can I tell if a select is used?

▸ The Processing Errors tab has DEBUG records for all of the rows that run:

Secondary Select Column Example

▸ These Local Mod Records with <u>new unique Column Names;</u> can update Non-Primary Key columns or even Primary Key Fields carefully.

▸ You must change the name to one not in use by the baseline or other local mod records.

▸ You must REMOVE any XML Tag Definitions on the Web Services Tab—only one column is allowed to send data for each target column.

▸ You should modify Process Order to follow baseline and often add an Update Condition.

Secondary Select Column Example

Secondary Select Column Example
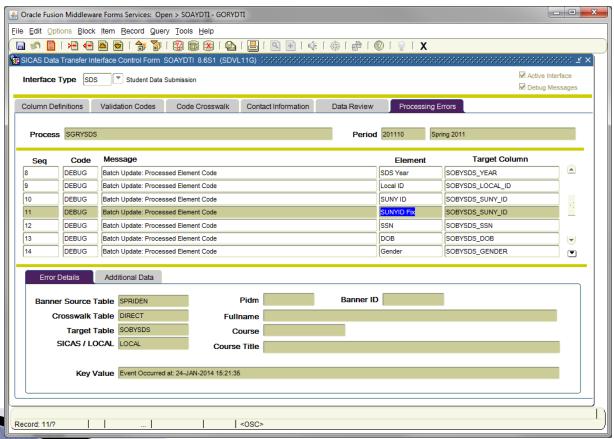
Secondary Select Column Example

SUNYID Fix Record is Common

- This last example is a very common need for all of the Person based Submissions.
- If System Admin changes a Student or Faculty SUNY ID then you will receive an error stating that you sent the wrong ID (you didn't).
- Use this kind of record to FORCE TO NULL the prior SUNY ID (only updates the bad records) and System Admin will send you the new SUNY ID value in the response. We store the new ID.
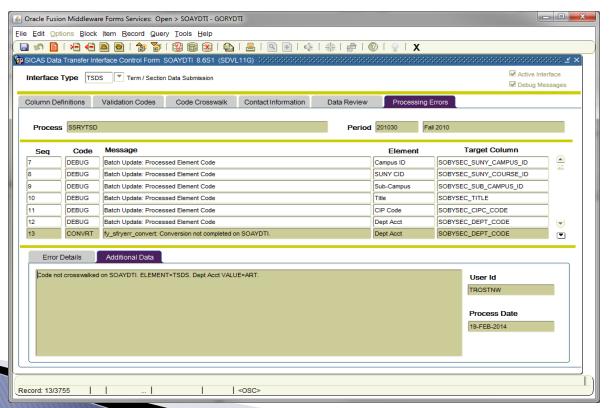- Update Condition should check for old ID(s).

How can I tell if these run?

▸ The Processing Errors tab has DEBUG records for all of the baseline and Secondary rows:

How can I tell if Selects work?

▸ If more than one record exists then errors occurred; many errors keep updates from doing anything; others impact one target row:

So what changes can I make?

- Process Method:
  - **Use Select Code** will "use the SQL to fill the column".
  - **Send the Default** will send the value in the Default column for every record.
  - **Force to NULL** will null out the contents of a column on every row. NULL can be thought of as an intentional empty column.
  - **Existing Value** will leave the Column alone—used to leave values assigned earlier in the stream or to have a Primary Key record for documentation.
  - **INACTIVE** will cause the record to not process—we do not use this often in SIRIS Interfaces.

What is "Use SICAS"?

- Process Method of "Use SICAS" is a very special mode that allows for the following:
  - ◦ Should only be used on LOCAL records (Use SICAS on a SICAS record will confuse the processing).
  - ◦ Allows a Local Mod Record to remain on the form but NOT replace the SICAS Record.
  - ◦ The most common use of this Process Method is to test the SICAS code to see if updates have made your local mod record obsolete.
  - ◦ Always remain on SICAS records when possible.
  - ◦ Test SICAS records after every update.

Other Local Mod Changes

- Default Value can be changed
  - Sent if Process Method is "Send the Default"
  - Sent if no data is found for "Use Select Code".
- Banner Source and Crosswalk Source (to alter or establish a crosswalk for this column).
- Web Service Settings are used only to remove tag settings in Secondary Select cases.
- Select Code can be edited to alter our logic.
- Update Conditions can be altered or added to restrict the update to a subset of target rows.

What can be changed in an FZ?

- Anything.

- These changes are always done in PL/SQL by a campus programmer & allow for complete control over the data in the staging tables which basically gives you full control.

SIRIS Submission Flow

- Run the process for a given Interface in AR mode first (one exception: CDS – AR/TR are only used to replace entire catalog—avoid this)
- Review the data to be sure it is reasonable (CDS running in AU/TU should have mostly Submit Status of CURRENT which don't go).
- Correct data for any issues found in review.
- Transmit Data (repeat until no Fatal Errors)
  - Run in TU mode (TR is only used to force full repeat)
  - Correct Banner data to fix errors returned from DTS.

SIRIS Submission Flow

- Even before you eliminate fatal errors you can start looking at the DTS Submission Dashboards to see if the totals are reasonable.
- Repeat TU and Review to correct problems you find in reviewing the dashboards.
- Once dashboards look good and you still have no Fatal Errors—use DTS to LOCK it.
- System Admin may UNLOCK it and request more changes on your part. Otherwise they APPROVE the submission and EXPORT it.

SIRIS Submission Flow

▸ Once the submission is EXPORTED you should immediately run the process one last time in PO (Posting Mode) to post the Banner side. It is safe to post because no changes can be made to either side once EXPORT has occurred.

Submission Dependencies

- All submissions other than SDS can only have ONE submission in process at a given time.
- SDS can have two active submissions
    - One each of ESS and EOT periods.
    - Error messages for only the last run (ESS or EOT) will exist—each run of SGRYSDS deletes all errors.
- CDS must be run before TSDS is run.
- SDS–EOT and TSDS should both be active until both are ready to LOCK; do not lock one without the other.

Submission Dependencies

- CDS can be UNPOSTED and rerun for a term when it is found that data is missing which is required by TSDS to fix issues.
- CDS can only be run for a term greater than any term run to date. If an older submission must be UNPOSTED then the newer terms must be removed with DB (Delete Batch) mode. I would avoid starting new CDS runs until the SDS-EOT and TSDS submissions are Posted.

A final word to the wise...

- SUNY IR personnel should be asked questions about DTS data received only.
- Never follow their instructions about our process modes (i.e. let us tell you when you need to TR your CDS—they don't understand our processes, only the resulting data).
- Again, do not LOCK TSDS or SDS-EOT separately—even if SUNY IR suggests it.
- All Banner processing questions need an RF.
- Most data issues you can't fix easily do too.

Questions?

Contact Information

SICAS Center

sicascen@oneonta.edu

(607) 436-2029

SICAS Web Site

http://sicas.suny.edu