ISCN data staging

2015-12-10

This working document contains information on the ISCN ArchiveSoilCarbon archive database and the staging processing that occurs in the ISCNStage database. Sections may contain open design issues, documentation on closed design issues, programming details, and instructions for staging data.

Table of Contents

INFO: Datasets

Central to the ISCN database is the concept of a dataset. A dataset is a named collection of data values at one or more spatial locators. Each new data submission generates a dataset. Data submissions include standard variable templates, overlay dataset templates, or special migration from a database such as NRCS.

Each dataset has a submission type:

- A *new* dataset contains only new spatial locators and new data. The dataset uses the standard ISCN template.
- An *overlay* dataset contains no new spatial locators and new data. An example of an overlay dataset is the GPP-ET dataset from MPI. The dataset used the then existing site lat/longs to perform a reverse lookup in the MPI gridded data product to generate GPP, ET and other flux variables at ISCN sites.
- A *migration* dataset follows the same reporting protocol embodied in the standard ISCN template. Here the data are pulled by SQL query from one database (Access or SQL). Data migration also includes transformation of the original data format to the ISCN data format. An example is converting lat/long from degrees/minutes/seconds to decimal degrees.
- An *analysis* dataset may contain both old and new spatial locators with new data values. Example is the Ping profiles in the Permafrost templates. These have different soil_taxon than the NRCS data.
- A *correction* dataset is associated with exactly one parent (previously processed) dataset. Correction datasets may contain both old and new spatial locators. For example, additional profiles and samples may be added. Data values may be either additions or replacements. Data are added when no such data exists in the parent dataset; data are replaced if the data exists in the parent dataset. For an example, consider a dataset that includes landscape at some, but not all, sites. The values contained in the correction dataset will be added when no such value existed in the parent and will replace the parent value whenever the parent value differs from the correction value.

Each dataset has a parent dataset.

- The parent dataset for a new, migration, or analysis dataset is the ISCN database dataset, "ISCN database".
- The parent for a correction dataset is specified at the time of data submission.
- The parent for overlay datasets by ISCN curators is the ISCN curation dataset, "ISCN curation".
- The parent for overlay datasets by other collaborators is "ISCN database".

The distinction for overlay datasets is less about the person that actually submitted the dataset but rather intended to indicate how such data might be updated after additional data submissions (new sites) or corrections (updated lat/longs).

Datasets are assumed to be new unless otherwise identified by the data submitter or ISCN curator.

- Only an ISCN curator can indicate a migration dataset. Such datasets require additional special processing code and creation of a mapping document specifying how the data are to be migrated.
- Only an ISCN curator can indicate an overlay dataset. Such datasets may require defining additional variables. When uploading such a dataset, the upload comment should indicate that the file is not a standard template and include any information necessary to decipher the file contents.
- The data submitter should indicate that the uploaded file is an analysis dataset at upload by appropriate text in the upload comment. An ISCN curator can also indicate that the dataset is an analysis dataset when the template is staged. Again, note that analysis datasets use the standard submission template.
- Similarly, the data submitter should indicate that the uploaded file is a correction dataset at upload by appropriate text in the upload comment. The text should include the parent dataset name. An ISCN curator can also indicate that the dataset is a correction dataset and the parent dataset name when the template is staged.

Datasets are tracked in the database in two ways:

- The submissions data are tracked in the dataset table. This table contains the submissions metadata (submissions file, name, email, date; (ISCN) curator id; processdate). The primary key is used as a foreign key in the variable, group, and site data tables to track data submissions, parent dataset relationships, and data retirement.
- The dataset name is tracked in the site table. This is primarily to enable flexible metadata such as citations and dataset_description to be associated with datasets.

The minimum metadata for each dataset is as follows:

- unique dataset name (dataset_name)
- short description of the dataset (dataset_description)
- submission date (modification_date)
- submission name, email, and organization. (curator_name, curator_email, and curator_organization)

INFO: Spatial locators

The ISCN database imposes a spatial locator hierarchy. A spatial locator is a named element of the hierarchy. At each level in the hierarchy, the spatial locator name must be unique. In other words, the (level, name) pair is used as *the unique identifier* for a spatial locator.

In descending order, the spatial locator levels are:

- site: sites are characterized by lat, long, and datum.
- profile: profiles are characterized by observation_date.
- sample: a sample may be a soil layer sample, gas sample, or other sample such as a biological sample. Soil layers are characterized by layer_top, layer_bot, and hzn_desgn (horizon). Gas samples are characterized by gas_type; other samples are characterized by other_type. Both gas_type and other_type are elements of controlled vocabularies.
- fraction: fractions are subsets of a soil layer sample and characterized by fraction_scheme and other metadata. Fraction_scheme is an element of a controlled vocabulary.

Again, note that while a spatial locator may be characterized by key metadata, the name is the unique identifier. In particular, key metadata may change across datasets. For example, the lat/long associated with a given site may have different values in the NRCS dataset and Permafrost dataset.

Note that there is currently no mechanism to delete (retire) or merge spatial locators. Note also that the NSCN database included an optional cluster level between site and profile. See the design issues section for more information.

INFO: Variable groups

Similar to the variable groupings for the AmeriFlux/Fluxnet BADM data, ISCN variables are grouped. In many ways, this is no different from the variable folding in the older database. The primary differences are the grouping is more visible to data submitters and data users and that the grouping is more flexible. For example, there is no numeric limit on the number of variables in a group.

Variable group changes are transactional – all the values for all variables in the group are updated together. This specifically means *any correction dataset must include all variables in the group*; the correction data are not merged with existing parent data in the same group. Consider the grp_location that contains lat, long, datum, and elevation. Suppose that the existing data are lat, long, and datum without the (optional) elevation. To add an elevation, the existing lat, long, and datum must be included in the submission. Similarly, to correct the long value, the lat and datum must be included in the submission.

Variables within a group may be:

- Primary. Each variable group contains at least one primary variable. When a group is submitted, at least one primary variable must be present.
- Required. Variable groups may also include required parameter variables. Required parameters must be included in the submission whenever the associated owner variable is submitted. An example is the required parameter variable long that must be submitted with the primary variable lat.
- Optional. These variables may be optionally included in the group. An example is datum that has a defined default of WGS84.

For grp_location, lat is the primary variable, long is required and datum and elevation are optional

Grouping variables must be done with insight into how potential corrections may occur. While it was tempting to group all site level variables into one group, separating the MPI flux, LBNL landuse, geography (county, state, country), location (lat, long, datum), elevation, and vegclass from other variables such as aspect_cl, stand_age or parent proved to be prudent when the existing data was examined.

The current variable groups represent a "best simple guess" approach. The guess was modified slightly after experience staging and checking the NRCS, existing NSCN and submitted templates in flight. When in doubt, The Truth is in the ancdecode table in the ArchiveSoilCarbon database on lbl.gov.eddy.

A few details:

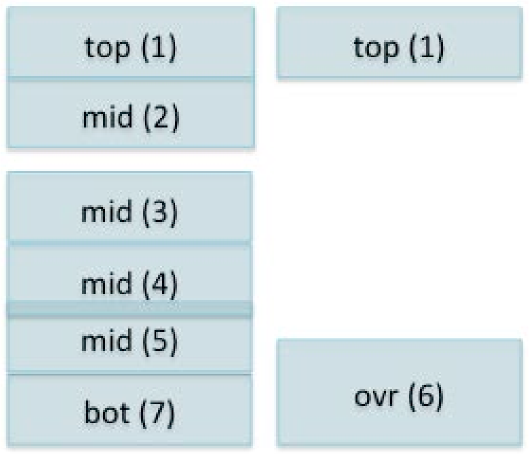
- As noted above, elevation is grouped with lat/long in grp_location. A special, one-time, merge was performed when migrating the NASIS data. The NRCS data contains only lat, long and datum; the NASIS data contains only elevation. When migrating NSCN sites with no lat/long but valid elevation, a special "ignore the missing lat/long" was done.
- The grp_geography contains the country, state, province, and county. Only the country is required. The staging code performs special checks to require state or province if the site does not have a valid lat/long. In other words, at the end of a valid push to archive, the site will have at least one of a valid state/province or valid lat/long.
- NSCN migration of the USGS_S3C (Buell) data ignores c_method without c_tot or oc. (The values are "L", "T", and "A"; not sure what "A" might be.) This appears to be a cut-overenthusiastic paste issue.

INFO: Sequencing layers

The ISCN database implements an algorithm for sequencing layers within a profile. That algorithm also detects gaps and overlaps between layers in the profile as well as extra layers. Note that the algorithm has changed from the older NSCN database.

Each layer is characterized by a layer_top and layer_bot. Profiles may use either the top of the organic layer or the bottom of the organic layer (the top of the mineral layer) as the zero reference. In other words, a 10 cm organic layer at the top of the profile may have [layer_top, layer_bot] = [0,10], [10, 0] or [15,5].

An example profile is shown at right. The "top", "mid", "bot", "ovr" text strings are the layer_indicators. The numbers in parenthesis are the layer_seq (sequence number). The profile has a pair of "clone" top layers with the same layer_top and layer_bot. There is a gap between the second layer and the third layer. There is an overlap between the fourth and fifth layers. The seventh layer is contiguous with the fifth layer and its layer_bot is the bottom of the profile. There is also an extra sixth layer that spans the fifth and seventh layers.



The sequencing code begins by tagging all layers with missing or obviously erroneous layer information. Examples include layers with layer_top = layer_bot, negative values of layer_top or layer_bot and missing layer_top or layer_bot. These layers are ignored in all subsequent computations.

The sequencing code next determines the profile_zero_ref variable value of "topO" or "botO". By default, profiles are assumed to be topO; this is true even if there is no layer_top = 0. Candidate botO profiles are identified by looking for layers with layer_top > layer_bot.

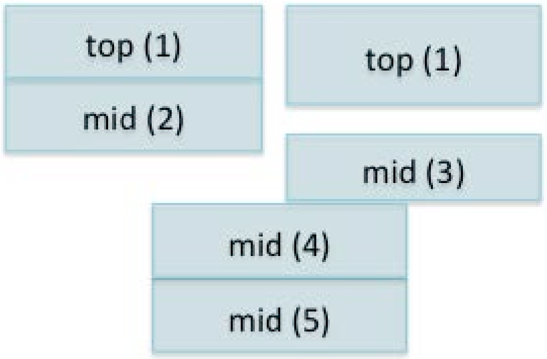
- If the hzn_desgn is available and either L or O*, the profile_zero_ref is set to botO.
- If the hzn_desgn is available and not L or O*, the layer is tagged as having bad layer information and the profile_zero_ref remains topO. A specific example of this is a layer at the bottom of a profile which has layer_top which is contiguous with a valid layer above but layer_bot = 0.
- If there is no horizon information, the profile is tagged as botO whenever the layer_depth is less than 40; that gives some protection against the layer_bot=0 case or data entry typos.

Internally, all computations use the topO reference for simplicity (layer_top always < layer_bot). botO layers are shifted using the uppermost layer as the 0 reference independent of the (optional) horizon.

The sequencing code then assigns layer sequence numbers by ordering the topO referenced layer_top values. Layer sequence numbers are used internally to simplify various queries, but are otherwise not used or exported.

The sequencing code then choses a representative layer for all clones - layers that share layer_top and layer_bot values. Having exactly one layer associated with a layer_top and layer_bot simplifies the subsequent code. At the end of sequencing, the sequencing information for the representative layer is shared with the other clones.

The "best" top layer is chosen from layers with layer_seq = 1 with consideration of the next two closest adjoining layers and their adjoining layers. That allows the code to minimize the total gap/overlap between layers in cases like that shown at right. Choosing the right path (the thicker top(1) + mid(3)) has a smaller total gap than the left path (the thinner top(1) + mid(2)) even though mid(2) is contiguous with the thinner top.



Note that the layer_top and layer_bot of a given named sample layer may differ across datasets. For example, a subsequent dataset may correct a typo in layer_top or add a horizon. Similarly, an analysis dataset might choose to reference all profiles as botO. As such, layers are sequenced for each parent dataset containing layer_top, layer_bot and hzn_desgn values. Layer sequencing impacts the computation of the layer carbon (soc_predict) aggregation into the computed profile carbon via computed layer weights. See the next section below.

INFO: Computing layer carbon

The ISCN database implements a non-gapfill computation of layer carbon. "Non-gapfill" means that there is no algorithm to compensate for missing layer carbon data, missing bulk density data or layer top and bottom. The algorithm selects the first of:

- Non-zero c_tot, oc or zero c_tot
- bd_samp, bd_tot, bd_whole. A check for the NRCS practice of filling the profile with 1.45 is made and any such values are ignored.

The internal soc_flag tracks the available data as a mask. For example "c-st-" indicates that c_tot, bd_samp, and bd_tot were available.

Datasets introduce a number of potential combinations here. At a minimum, the carbon computation has a dependency on:

- The grp_layer_note variables layer_top, layer_bot and derived sequencing information such as the layer weight factor.
- The grp_sample variables c_tot and/or oc.
- The grp_soil variables bd_samp, bd_tot, or bd_whole.

Each of the above has a parent dataset that may be different from the parent dataset of the others. For example, the current dataset may apply a correction to c_tot, but leave the grp_layer_note and grp_soil variables unchanged. Similarly, the current analysis dataset may supply new values for bd_samp, but leave all others unchanged.

Internally, the database computes a layer soc value only when the grp_layer_note, grp_sample and grp_soil variables can be sourced from a single parent dataset. In other words, the database does not merge data across parent datasets but does merge child and parent datasets. This was done to:

- limit the combinatorics to something that can be explained to a human.
- reduce the complexity of the code to simplify testing and long term maintenance.

A consequence of this is that a dataset that includes only grp_layer_note and grp_sample cannot be merged with a dataset that contains the same grp_layer_note values and grp_soil to generate a soil carbon computation. Instead, the existing grp_sample values should be merged with the second dataset prior to staging. This can be done by seeding a template with grp_layer_note and grp_sample values and having the data submitter add the grp_soil variables or by special merge at template stage. The latter is the only option near term.

The NRCS prep_code further complicates the combinations. The source tables for carbon (c_tot and oc) and bulk density (bd_samp and bd_tot) contain (relatively undocumented) "prep_code" columns. The source table for bd_whole has no prep_code. As such, same layer may be associated with one of 12 different carbon prep_codes and one of two different bulk density prep_codes. The good news is that the ~96% of samples are associated with a carbon prep_code of "S" and ~60% are associated with only one prep_code combination. The bad news is that when there are multiple prep_codes, the difference in the values is usually more than 10% and may be factors of 4 or more. The approach taken is to compute all prep_code combinations and apply any selection policy when computing profile carbon or generating a layer BWT data product.

Lastly, a change from the previous calculation has to do with the precision of the calculation. The older NSCN calculation simply CAST the stored character strings as REAL single precision floating-point numbers. The new ISCN calculation CASTs the stored character strings as REAL(DECIMAL(10,4)) and then formatting the result as DECIMAL(10,2) (the current soc export format). While this can cause some changes in the result, it also means that the results are closer to what a user of a BWT data product would obtain when attempting to do the ISCN computation.

INFO: Computing profile carbon

Profile carbon is computed by aggregating the layer carbon computation. The computation is done twice:

- A 1m aggregation is done to a depth of 1m from the surface. The layers in the profile must have no more than an accumulated gap of 5cm over that depth.
- A total aggregation of all layers. Note that this computation will include profiles that have large gaps, may be very short, may have layer_top > 1m, and other less desirable characteristics.

The profile carbon computation is associated with two quality flags:

- soc_spatial_flag. This flag indicates the completeness of the profile. Values include "Complete","Complete(5cm)", "Short", "Short(5cm)", "Gap"
- soc_carbon_flag. This flag indicates the completeness of the soc computation. Values include "no_soc","soc_total", "soc_1m", "soc_total_gap", "soc_1m_gap".

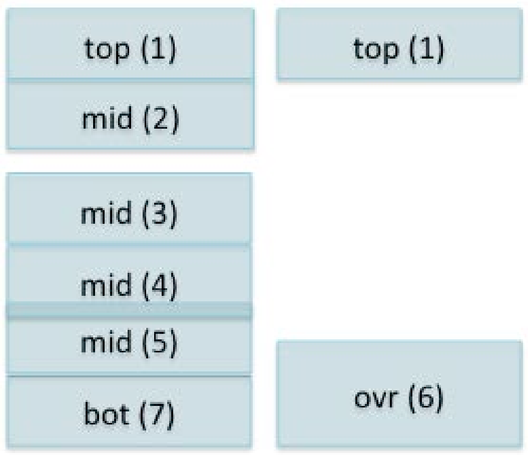
Note that investigator computed soc can also define soc_spatial_flag and soc_carbon flag.

Layer gaps, layer overlaps, layer clones, NRCS prep_codes, and data cruft all mean that the aggregations are not always simple sums over the contributing layers. The approach taken is to compute:

- A carbon weight for each layer. The carbon weight handles NRCS prep_code policies and data cruft.
- A layer weight for each layer for each aggregation. The layer weight handles layer overlaps, layer clones and aggregation depth.
- A layer gap for each layer for each aggregation. This is the gap between the current layer and the minimum of the next lower layer or the aggregation depth.
- The profile carbon for each aggregation. This is the sum of the computed layer soc * carbon weight * layer weight across all layers in the profile.
- The profile gap for each aggregation. This is the sum of the gaps to the aggregation depth.

The carbon weight is computed by weighting the layer subsets with valid layer soc. An example profile is shown at right. For non-NRCS layers:

- If the layer soc is invalid, the carbon weight = 0.
- If the layer is unique without clone or overlap, the carbon weight = 1. If mid(2) has valid soc, the carbon weight = 1.
- If both top(1) layers have valid soc, both have carbon weight = ½.



- If only one of the top(1) layers has valid soc, the layer with valid soc has carbon weight = 1 and the other has carbon weight = 0.
- If both mid(4) and mid(5) have valid soc and the overlap is 20%, the mid(4) carbon weight = 0.8 + 0.2/2 = 0.9

The NRCS prep_code acts as an additional factor in determining "valid soc".

- As of this writing, there is no "invalid" or "ignored" NRCS prep_code.
- If the NRCS prep_code for the carbon (c_tot or oc) is "S", the carbon weight follows the above rules. In other words, layers with "S" carbon prep_codes are chosen in lieu of the other clones.
- If there are no "S" carbon prep_code layers, the remaining prep_codes are averaged. The carbon weight = 1./(the number of clones with valid soc).

INFO: Comparing NSCN and ISCN carbon computations

The previous sections, of course, lead to the question of how to validate the new ISCN code given the differences in the computation. It also suggests that we need a user document explaining the differences.

The ISCN layer soc computations differ from the NSCN computations in that the ISCN computation:

- Uses oc if available when c_tot = 0.
- Ignores bd for NRCS profiles where the profile was filled by 1.45.
- Computes layer soc for each NRCS prep_code rather than a single prep_code chosen by "luck of the SQL draw"
- The precision is different.

The first two changes impact only a very small fraction of the data. The change in precision makes only a small difference in the computed soc value.

The ISCN profile soc computations differ from the NSCN computations in that the ISCN computation:

- Computes the layer weight only for layers with valid soc. The NSCN computation computed layer weight without regard to valid soc. In other words, the NSCN layer weight for both of the top(1) layers = ½ regardless of whether the layer has soc.
- Interprets layers without valid soc as gaps between layers. Consider what happens when layer mid(2) has no valid soc and a layer depth of 3cm. The NSCN computation will classify the profile as no_soc. The ISCN computation will classify the profile as "Complete(5cm)" if the total profile depth is >=1m and all other layers have carbon.
- Considers the NRCS prep_code. The "luck of the SQL draw" can change the computed soc value. This impacts approximately 5% of the samples.

- Uses the new NRCS profile names. The NRCS/NASIS 2014 database (mostly) imposes a different naming convention for profiles and sites. Relating the new to old names across the NRCS 2011 Access database, the NASIS 2011 Excel database submission, the NRCS 2014 Access database and the NASIS 2014 Access database required a number of heuristics.
- Stages NRCS profiles differently. The NSCN harvesting code did a much more aggressive folding of layers into a single profile than the ISCN harvesting logic. Some of these differences are due to the differences between the NRCS 2011 database and the NRCS/NASIS 2014 database; the 2014 databases are often filled with values that are not easily identified as placeholders (and therefore ignored). The newer staging code also takes a more conservative approach to folding. This impacts approximately 2% of the profiles.
- Uses a different precision. The differences may be small at layer granularity but aggregate over a profile.

The differences in layer weighting, soc gap interpretation, and precision complicate simple comparisons between the ISCN and NSCN computations for investigator submitted data. The additional differences in the NRCS prep_code handling, profile naming, and profile folding complicate simple comparisons between the ISCN and NSCN computations for NRCS data.

In other words, the challenge in comparing the ISCN and NSCN computations happen not in the "sunny day" case of an ideal – no overlaps, no layer gaps, no data gaps, single NRCS prep_code – profile, but rather all other profiles.

As such, a purely programmatic comparison was abandoned. Instead, a combination of "scenario" - the difference can be attributed to – eliminations and visual inspections were used. Additionally, all NSCN computations with valid soc (NSCN or AK gap fill) were migrated as analysis datasets. The intention is that more careful comparisons can be made at a future date.

INFO: Controlled Vocabularies

The staging code now validates all variables with a controlled vocabulary against that vocabulary. Submitted text strings must match the shortname text string. The shortname text string attempts to be long enough to avoid obscurity and short enough to avoid too many possibilities for typos. The staging code does no remapping or alias detection. In other words, "Alaska" is valid but "AK" is not.

The good news here is that using the archive values for search/sort/filter is simpler. The bad news is that choosing just one CV value for many of these variables is problematic. This problem is addressed by including a free text note variable in the group. That variable can be used for explanatory text and/or a more detailed description. For example, the NASIS 2014 database contains multiple values for landform and landscape associated with a given site. The original strings are concatenated to form the associated note, each string is remapped to follow the CV and the first encountered string is chosen as the value. For example, consider a site with landforms of "cuesta or hill" and "hillside" where the former is the first encountered string. The landform value would be "cuesta" and the associated note would be "cuesta or hill, hillside".

Older NSCN variable values that did not obey the relevant controlled vocabulary were remapped as part of the data migration. This simplifies subsequent data product production. The variable value remapping was either via simple alias (e.g. Alaska for AK) or via a mapping specified by the science curator. In the latter case, the original text string is migrated in the associated variable note as in the above example.

Going forward, the controlled vocabularies can be expanded (e.g. adding non-US states from NRCS) as necessary.

The table below summarizes the current non-disturbance CVs and describes changes from NSCN to ISCN. Note that this table may be out of date; when in doubt The Truth is in the ancdecode and ancCV tables in the ArchiveSoilCarbon database.

variable	CV	Comments
2d_position	2d_position	See landform.
burn_ev	Burn	This variable originated with the Alaska database. The CV was denigrated and the variable converted to free text. The existing values were a real mix; remapping was not practical.
soc_type	ccon_type	No change.
country	country	This CV was added.
observation_date_acc	date_qual	This CV was denitrated. The existing free text submissions (e.g. "1 month" or "<1 day") tended to be more informative than the CV strings.
Datum	datum	"old Hawaiian" and "guam" are in NRCS, but not in the CV. Changing to the NRCS 2014 database makes this moot as all lat/longs are in WGS84. The migration code for the NRCS 2011 database ignores this, but all other submissions will check the CV. If there is a simple standard conversion, I suggest we make that conversion as part of the migration.
aspect_cl	direction	"level" was added to the CV and "none" was remapped to "level". For

		migration, strings were also converted via simple alias (e.g. “n” became “north”).
drainagecl	drainage	Current submissions are a mix of “xxx drainage” and “xxx”. Suggest we pick one, clean the existing and check the CV forward. The current code uses “xxx”. There are also cases where the text string cannot be mapped; these need to be converted or not migrated. Examples include “”w-wm” and “e-es-w”.
flood_freq	flood_freq	No change.
fraction_scheme	fract_scheme	No submissions, hence no change.
gas_type	gas_type	No submissions, hence no change.
publish_phase3	inc_exc	No change.
landform	landform	The 2d_position, landform, and landscape CVs were rebuilt from the USDA-NRCS documentation. The short description text string and not the two/three letter code was used. Existing values were remapped The NRCS reference defines a set of (code, definition) tupleOne downside is that the landscape CV uses almost many of the same terms – an example is “hill” vs “hills”. Another issue here is that the NASIS submission often included one or more terms (eg “cuesta or plateau” or “cuesta, hill”). Either we need to have a rule (eg pick the first term) or we need to add a “secondary” variable. I suggest the rule.
landuse	landsat	This CV was defined by the LBNL overlay dataset. No change.
landscape	landscape	See landform.
other_type	other_type	No submissions, hence no change.
pond_freq	pond_freq	No change.
profile_zero_ref	profile_zero_ref	This variable was introduced due to BADM discussions with Margaret Torn. The NRCS profiles are clearly a mix of “zero at top of organic and/or surface” and “zero at top of inorganic”. Introducing this variable with a defined default makes the carbon computation code both more robust and simpler.
Runoff	runoff	No change.

site_perm	site_perm	No change.
stand_maturity	stand_age	The CV does not include all combinations (eg “mature, even age” is in the CV, but “mature, uneven age” is not). There are a few submissions from NSCN that use the missing combinations. I suggest we consider the missing combinations and decide if they should be added or are nonsensical.
State	state_province	Altered from two letter codes to longer strings (e.g. AK becomes Alaska). Canadian provinces and territories added. Additional strings added as necessary for NRCS migration.
add_taxon_flag	yes_blank	No change.
labeled_addition	yes_blank	No change.
polar_flag	yes_blank	This variable was used as part of the Alaska gap-fill algorithm. It is now pushed as part of the soc_flag (quality flag).
transect_flag	yes_blank	No change.

An important additional open issue is the handling of unknown or not applicable values.

- For any required variable that has a CV, we should include some sort of “unknown” concept in the vocabulary.
- For all other variables, we have a choice:
  - We can prohibit values that are not in the CV in the staging code. In this case, the template must be changed and resubmitted.
  - We can remove values that match “unknown”, “NA” or other similar strings in the staging code. In this case, the template would not require changes and the value would be flagged as an informational log message. The archive database would contain no value; in other words, there would be no difference between a template containing “unknown” and a template containing a blank.

INFO: Data migration from NSCN

See the above section on controlled vocabularies for changes to any variable with a controlled vocabulary. Again, in all cases, whenever an existing string is changed other than a direct map (eg “AK” to “Alaska”) the existing string will be in the associated note variable.

Migrated disturbance data will use only the AmeriFlux/Fluxnet “dist_lite”. dist_lite eliminates most of the disturbance-specific parameters in favor of a disturbance-specific CV indicating the nature of the disturbance (eg flood, hurricane or drought for dm_ext_weather) with associated date or date range and comment. While this does lose some information, it also leverages what we've learned from BADM. The information could be a) migrated via comment or b) migrated at some point later. We've also seen only ~200 disturbances other than the very free form Alaska submissions (which are primarily either "none (mature vegetation)" or "geomorphic processes").

Other data spring cleaning includes:

- Variable min/max values were ignored.
- Four variable values that could not be converted to numbers with simple text replacements such as elimination of "<" or "~" were converted to numbers by Luke.
- Variables that violated the CV were remapped to CV. That includes converting shortname strings to longer strings (eg "AK" to "Alaska") as well as Luke's science informed remappings. In these cases, the original text string was preserved in an associated note variable (eg landform_note for landform and landscape).

The LBL land cover and MPI GPP, MAT, MAP et al. special datasets were not migrated. These datasets are generated by lat/long lookup. Given the changes in the NRCS 2014 dataset and the number of newly submitted sites, the plan forward is to regenerate these datasets.

The ISCN ecoregion dataset was also not migrated. This dataset holds the SciScope Omernik ecoregion lookup. The data were not migrated, but rather regenerated from the site lat/longs. That regeneration can be done at any time after staging new templates.

See the section below for information on migrating the NSCN and Alaska carbon computations.

INFO: Data migration from NRCS 2014

Thanks to Luke, we obtained a version of the NRCS Access database (analysis_pc_2.1_47915pedons_contains_lab_data_table_and_taxonom.mdb) and the associated NASIS database (NCSS_Soil_Characterization_Database_Sept_2014.mdb) as of September 2014.

This section discusses challenges encountered staging the new data and relating that data to the older NRCS 2011 database. Note that the 2014 database was obtained after the initial migration of the 2011 database was completed; that migration was discarded in favor of the newer data.

The good news with respect to the NRCS 2014 database is:

1. The database contains:
    a. 389364 total layers of which 334329 layers have carbon and/or bulk density and 38087 additional layers have soil texture but no carbon and/or bulk density
    b. 153405 new layers with data not present in the NRCS 2011 database.
    c. 62349 distinct profile keys. Of those 1411 are invalid (no corresponding row in the profile table. There are also 566 profiles with duplicate names. My best guess is that about 60,000 profiles can be staged in comparison with 37380 profiles in the NRCS 2011 database.
    d. 61841 distinct site keys. Of those, 4 are invalid (no corresponding row in the site table). There are 637 replicated names. Many of the replicated names are familiar and associated with similar issues in the NRCS 2011 database. I won't have a guess for how many sites will stage until the site folding logic is tweaked (see below).
2. There are 7146 layers in the NRCS 2011 database that are not present in the NRCS 2014 database. Either these layers have no carbon or were "dangling" – without profile or site keys. In other words, this will not be an issue when comparing the old and new derived carbon values.
3. In general, the foreign key relationships between the layer, profile, and site tables are more robust. There are fewer dangling references and fewer duplicate profile and site names.
4. All lat/longs are now in WGS84 decimal degrees. No conversion required.
5. The soil_series and soil_taxon strings are now parsed into distinct columns and the longer string regularized. There are three values: sample, correlated, and "SSL".
6. The layer_top/layer_bot information for layers also in the NRCS 2011 database is unchanged with the exception of 57 layers. Again, unlikely to shift the carbon computation significantly.
7. The lat/long information for layers also in the NRCS 2011 database seems comparable. Only 2% of the changes cannot be attributed to datum conversion, rounding error, or other causes.
8. While there are a number of schema changes, most are relatively minor. For example, the "Carbon and Extractions" is now "Carbon_and_Extractions" and the user profile name column user_pedon_id is now upedonid.

The good news wrt the NASIS 2014 database is:

1. We have direct access to the database rather than an exported data product. This makes the data harvest both simpler and more robust.
2. After a conference call with the NRCS folks and after some experimentation, we have closed on using the profile name (pedlabsampnum column) to relate this

database to the NRCS database. After joining the NRCS NCSS_pedon_taxonomy table to the NASIS pedon table, we then use the NRCS site_key to join to the NCSS_site_location table and the NASIS siteiid/siteiidref to link to the NASIS site table.

3. The database contains:
    a. Number of profiles with NRCS match
    b. Number of sites (with NRCS match)

The not so good news is:

- The bulk of the new data has only carbon and no bulk density data.
- Half of the profiles and three quarters of the sites associated with layers in the NRCS 2011 database have been renamed.
    - In most cases, the old and new names are something that a human might recognize as "renamed".
        - Examples of profile old name: new name include: 70MI051001:S1970MI051001, S01TN-155-002:01TN155002, 53OK117035:53OK117035(Lab Data), and 83IL183013m:83IL183013a. The danger here is that there are many sites with similar names.
        - About 5% of the renamed profiles and 7% of the renamed sites have little or no such simple relationship. Examples include: 61PR121001:D99PR079005, S01NM-035-016:Jerag Tax - NTC11, 99CA109005:790204.
    - On the conference call, we heard that the goal here was to introduce a common naming convention of <S>YYYYSSCCCnnn where the first S is optional, YYYY is the year, SS is the state, CCC is the county code and nnn is a sequence number ensuring unique names. Unfortunately, the actual data base contents does not achieve that goal.
    - There are two profile name columns: pedlabsampnum and upedonid. The former string is used to join to the NASIS information. Neither the pedlabsampnum nor the upedonid strings are unique.
    - There is one site name column: usiteid. The strings are not unique. Experimentation with the linkage to the NASIS database suggests that only the pedlabsampnum should be used to join the databases. Site information is joined by first joining the profile and then using the site keys in the profile tables to join the sites.
- The observation_date has been moved from the profile table to the site table and converted to a text string. The strings cannot be simply cast as a date as there are a few strings such as "6/1/9863" and there are dates in the future.
    - The observation date changes in about 25% of the profiles with no immediately obvious pattern (the new dates are both before and after).
    - My guess is also that the date "7/15/2014" is used as some sort of indicator as it appears in 14% of the rows; the next most common date

appears in 2.5% of the rows. On the conference call we learned that this was unknown to the NRCS folks.
    - The NASIS database also tracks the observation_date as a site variable, uses a different format text string with an "Actual Observation Date" date identifier. The NRCS database also contains some cases of multiple dates per site and bad date strings.
- The NRCS database now maintains a 1:1 relationship between the site_key in the NCSS_pedon_taxonomy table and rows in the NCSS_site_location_table. In other words, profiles that previously shared a site now have unique sites. Unfortunately, the site metadata (especially lat/long) is not always replicated when a new site row is created. Some sites also lost metadata. Since the names may have been changed, there is no simple way of repairing the lost metadata.

The above has some fairly icky implications for resolving the NRCS 2011 data with the NRCS 2014 data. A case in point is the NRCS 2011 site "1 Exeter".

- In the NRCS 2011 database, there are three distinct rows in the site_table with the name "1 Exeter". There is no lat or long or other site metadata. Each site row corresponds to a distinct profile and each profile has a unique date. The older NSCN staging code folded the site rows into one site as there is no distinguishing information. The profile names are "S0ME-019-027", "S0ME-019-10", and "S0ME-019-15",
- In the NRCS 2014 database, the same layers again contribute to three distinct profiles. The profile names here are "S0ME-019-027", "ME-02-07-019-10", and "ME-02-07-019-15". Each profile again has a unique site_key but the names are now "ME-02-07-019-10", "ME-02-05-019-15", and "1 Exeter". There are three observation dates, but none of them match the observation dates in the NRCS 2011 database. LOCATION ?

A note of explanation. The above comparisons for layer_top, layer_bot, lat, long, and observation_date were made by walking the NRCS_2014 spatial hierarchy to retrieve values and then independently walking the NRCS_2011 spatial hierarchy to retrieve the equivalent values. In that process, an interesting observation was that the site_key, pedon_key, and layer_key relationships appear to have been (mostly) unchanged across the two databases where a new "site" was not added.

The plan forward is as follows:

1. Continue to harvest only profiles containing at least one layer with carbon, bulk density, or soil texture data.
    - All layers in each profile are harvested. This is necessary for the layer sequencing logic. For example, consider a profile with the zero reference at the bottom of the organic layer, two layers in the organic horizon, and carbon only in the lower organic layer. Keeping the upper organic layer is important to set the actual depth of the profile.

2. Use the pedlabsampnum as the profile name and the linkage between the NRCS and NASIS databases.
   - Visual inspection of the name collisions for profiles suggests that the profiles are actually the same. In other words, with one exception that appears to be an overlap layer, the profiles have the same layer information and data. These will be folded.
3. Use the NRCS database as the preferred source for lat/long.
   - The NRCS database contains digital lat/longs and no datum; the internal column documentation indicates that the datum is WGS84.
   - The NASIS database contains lat/long in degrees, minutes, seconds with datum. Missing (null) seconds are treated as 0; missing degrees or minutes cause the value to be ignored.
   - Some spot checks suggest that the conversion may or may not have been made and/or an incorrect conversion was made in some cases.
   - The staging code folds the NASIS elevation (if any) into the lat/long. When an elevation is available with no lat/long, the code repairs the check for required lat/long.
4. Use the NASIS database as the preferred source observation date.
   - When the NASIS database contains multiple observation dates, the earliest date is used.
   - In theory, since the observation date is now tracked on the site and the site_name follows a naming convention, we should be able to extract the year from the site_name to fill missing observation dates. In practice, no site missing an observation date followed the naming convention.
5. Use the NRCS correlated (corr) values as the preferred source for soil_series and soil_taxon.
   a. Missing or obviously bad strings (eg '.') are filled with the sample (samp) values (if any).
   b. The correlated and sample values are pushed to the profile_properties_note.
6. Remap landform and landscape strings to follow the CV per Luke's mapping.
7. Ignore "NL" (National Lab) and FN (Foreign) state values.
8. Resolve the colliding profile names by visual inspection. The 13 of the 14 impacted profile names were determined to have identical layers and orthogonal NRCS prep_codes. The remaining collision appeared to be an extra layer associated with a sampled profile.
9. Continue to fabricate profiles for layers with invalid pedon_keys but valid site_keys. (Why this happens remains a mystery given the other work on the database.) Profile names will prefix the disambiguated site name with "ISCN:" e.g. "ISCN:83AK122006:10778".
10. Continue to fold any site with the same name and location information (lat/long or, if no lat/long, state). There is no reason to distinguish these given the linkage

to NASIS via the profile pedlabsampnum. Note that this will cause less folding than the 2011 database due to the site name changes.

11. Continue to disambiguate any remaining sites with the site key. An example is "83AK122006:10778" where 10778 is the site_key.
12. Disambiguate any sample, profile, or site name in the NSCN database by prefixing the spatial locator name with "ISCN:". This is specifically intended to enable migrating investigator submitted datasets and the older NSCN carbon computations. The collision here in the original Alaska Ping submissions. Knowing what we know now, these submissions would have been handled either as a DUPE or as an analysis datasets.
13. Handle the issue of comparison of profile carbon computed with the 2011 and 2014 database via an analysis dataset. See the section on "Migrating NSCN carbon computations".
14. Skye <xxx> has an algorithm implemented in R which does the bulk density gap fill. Luke can work with her to submit an analysis dataset for the NRCS and/or other data. Given the NRCS 2011:NRCS 2014 experience, I suggest that that submission include at least the same spatial locator metadata.

INFO: NRCS profile and site folding

The primary challenge in migrating data from the NRCS 2011 Access database is resolving the spatial locators. Access does not enforce key relationships. There are foreign keys in the ncss_layer and ncss_pedon (profile) tables that have no corresponding primary key in the ncss_site table. Worse, there are multiple rows with the same profile name in the ncss_pedon table and multiple rows in the ncss_site table that have the same site name (user_site_id). In some cases, the multiples are associated with the same metadata (eg lat/long). In other cases, the multiples have different metadata including variables such as state from the nrcs_site_area table. There are also a number of rows that have no metadata.

The older NSCN migration started at the site level and worked down to the sample level.

- Rows in the ncss_site table that shared the same name and same lat/long were folded into one "survivor" site.
- Rows with different metadata were disambiguated by appending the site_key to the name (eg "83AK122006:10778" where 10778 is the site_key).

Once the sites were disambiguated, the profiles were handled in a similar fashion using the observation_date (if any) as the differentiating metadata. Finally, samples were linked to profiles. Samples with no data were eliminated and that elimination rippled up through profiles with no surviving samples and sites with no surviving profiles.

The new ISCN migration starts at the sample level.

- The first step is to retire samples that have no carbon (c_tot or oc), bulk density (bd_samp or bd_whole), or soil texture data. Note that this retirement is more aggressive than the NSCN algorithm as these sites may have other measurements (eg ph). Note that the soil texture was included as it is used in the Alaska gap-fill carbon computation.

- The next step is to resolve the (site_key, pedon_key) pair in the ncss_layer table with the (site_key, pedon_key) pair in the ncss_pedon_taxonomy table. In general, these are identical; he pedon_key acts as a foreign key in the ncss_pedon table and the site_key acts as a foreign key in the ncss_site_location table. When that is not the case
    - if the ncss_layer keys pair is valid, they take precedence over the ncss_pedon_taxonomy table key pair.
    - if the ncss_layer key pair is not valid, the ncss_pedon_taxonomy key pair is used.
    - If neither pair is valid, manual fixup is attempted.
        - if the layer "fits" in a profile with valid keys, the layer inherits that key pair. "fit" here is a subjective combination of "the sample name is like the others" and "the layer_top and layer_bot fill a gap in the profile".
        - if the layer is the only layer in the profile, the layer is retired.

After resolving the samples, the profiles and sites are then resolved.

- Profiles that have no associated samples are retired.
- As noted above, the colliding profile names were manually resolved after visual inspection.

After resolving the profiles, the sites are then resolved.

- Sites with no associated samples and/or profiles are retired.
- Sites with unique name, lat, and long are accepted. This is the overwhelmingly common case (~95%).
- Sites with common names with matching lat and long are folded. There is no reason to distinguish the rows.
- Sites with common names where exactly one has a lat and long are folded. There is no reason to distinguish the rows.
- Sites with common names, but distinct lat and long are disambiguated. The site with the lowest site_key retains the name; all others are renamed by prepending "ISCN:' and appending ":" + site_key.
    - While this seems identical to the NSCN algorithm, it does not always yield the same result because of the earlier and more aggressive retirement of samples and changes to the site names in the NRCS 2014 database.

As an example of what this means in practice, consider the three samples as shown in the table below. The sample, profile, and site names are from the NRCS database.

Note that there are actually two profile_keys and two site_keys; the sample 04N00305 has one such pair and the samples 04N00703 and 04N00704 have a different pair.

sample	profile	Site	NSCN	ISCN
04N00305	S03DE-003-304	S03DE-003-304	All three samples are retained. The site name for all three is S03DE-003-304 as it best matched the profile name.	The 04N00305 sample is retired due to no data. The remaining samples are staged with site name DNE-4.
04N00703	S03DE-003-304	DNE-4		
04N00704	S03DE-003-304	DNE-4		

Again, the good news is that roughly 95% of the NRCS profiles have unique keys and are not impacted by the above. Impacted sites also tend not to have valid lat/long and/or are shorter than 1m.

A secondary concern when migrating the NRCS data is how to handle the NRCS prep_code. In true "late binding" aka "delay that decision as long as you can" mode, the data are migrated with the prep_code. A given variable group has exactly one prep_code. The good news here is that ~98% of the layers that have soc have the prep_code "S" for both carbon and bulk density.

INFO: Migrating NSCN carbon computations

The NSCN database implemented two carbon computations:

- NSCN: a simple non-gap-fill computation
- Alaska: a gap-fill computation per Kris Johnson for Alaska sites.

Both include directly computed values for layers and aggregate layer computations for profiles.

- The layer spatial quality flag indicated the validity of the layer_top and layer_bot. The layer carbon quality flag indicated the availability of the carbon and bulk data values (NSCN) or fill type (AK).
- The profile spatial quality flag indicated gaps/overlaps and whether the profile extended to 1m. The profile carbon quality flag indicated whether the value was valid to at least 1m and/or had been gap filled.

The NSCN computations were exported in the layer BWT, the profile BWT (1m and total), and the carbon to 1m reports. The Alaska profile computations (1m and total) were exported via a report.

The non-gap-fill layer computation can diverge from the NSCN non-gap-fill computation.

- The layer computation may diverge due to actual data changes, variable precision, NRCS prep_codes, and variable rounding.
- The profile computation may diverge due to actual layer data changes, variable precision, variable rounding, changes in the profile folding, and changes in the profile naming.

What this means for the migration is the following:

1. The migration is accomplished by constructing four analysis datasets. Each includes only profiles with valid soc computations.
    - NSCN non-gapfill data. This dataset includes all layer computations with valid layer_top and layer_bot and their associated profile aggregates. It also includes the aliasing from the new NRCS 2014 profile and site names to the older NRCS 2011 profile and site names.
    - NSCN 1m data. This dataset includes only valid 1m profile aggregates. It also includes the relevant NRCS aliasing.
    - Alaska data. This dataset includes all layer computations with valid carbon computations resulting from the NSCN implemented gapfill algorithm. The same query used to generate the report is used. In other words, the migration code moves bug-for-bug values with no checking or other computation. It also includes the relevant NRCS aliasing.
    - Alaska 1m data. This dataset includes only valid Alaska 1m profile aggregates. Like the Alaska data, it is bug-for-bug compatible and includes the relevant NRCS aliasing.
2. The relevant spatial locator metadata were included. In particular, the layer_top, layer_bot, hzn_desgn, observation_date, lat, long, datum, country and state were included. This ensures that these will populate any derived BWT data product.
3. The actual carbon and bulk density used to generate the soc were included. Note that these values are selected from c_tot/oc and bd_samp/bd_tot/bd_whole or generated by the Alaska gap-fill algorithm.
4. Spatial locator and carbon computation quality flags were added to the grp_soc. These variables are currently ComputeOnly, but we may want to include them in future templates as free text variables enabling other investigators to indicate quality.
    a. For the NSCN datasets, the sample soc_carbon_flag is the data availability mask indicating the presence of c_tot, oc, bd_samp, bd_tot, and bd_whole and one of "missing Data","no Data" or "noFill" indicating the success of the computation and "NRCS" if the source data are from the NRCS 2011 access database.
    b. For the Alaska datasets, the sample soc_carbon_flag includes the data availability maks and the gap-fill equation (if any)
    c. The sample soc_spatial_flag indicates whether the layer is "Contiguous", "Discontiguous" or "MissingInfo"

    d. Note that all samples in a profile with a valid profile soc computation are migrated regardless of whether the sample itself has a valid sample soc computation.

5. For each profile and site, a ComputeOnly locator_alias variable was added whenever the NRCS 2014 name and/or folding differed from the NRCS 2011 name and/or folding. From a database perspective, this variable is an embedded spline that is updated by transaction.
6. For each profile and sample, a ComputeOnly locator_parent_alias variable was added whenever the NRCS 2014 parent name and/or folding differed from the NRCS 2011 parent name and/or folding. From a database perspective, this variable is also an embedded spline that is updated by transaction.

INFO: Migration from the original Alaska soil carbon database

As part of the QA/QC of the NSCN data migration, three older datasets in the original Alaska Soil Database were discovered. These were migrated forward into the new database.

As part of that migration, two additional issues were discovered:

- The soiltaxon variable was migrated as soil_series rather than soil_taxon.
- The landform variables in the Alaska database were not migrated with full fidelity to the NSCN database. In particular, there is additional information that could now be migrated as a landform_note.

As a result, this migration generated 3 "new" submissions and a number of "correction" submission templates containing the "delete" of the soil_series and updates to the soil_taxon and grp_landform.

INFO: About analysis datasets and investigator reported carbon

The introduction of the dataset concept should[1] simplify investigator computations, ISCN data product computations as well as changes in algorithms and/or data corrections that feed into those computations.

All data in an analysis dataset are treated as a separate dataset and the database does no merging across datasets other than overlay datasets. All dataset shares the spatial locator fraction_name, sample_name, profile_name, and site_names. Each dataset may have different layer_top, layer_bot, observation_date ,soil_taxon, lat, long or other spatial locator data values.

[1] Or so the moose says on his itty bitty card.

An example of an investigator reported carbon dataset is the Alaska gap-fill algorithm migrated from the NSCN database. The algorithm implements a number of heuristics to fill missing sample carbon or bulk density measurements to enable a profile carbon computation.

For each sample from an Alaska site in the NSCN database, the layer_top, layer_bot, c_tot, oc, bd_samp, bd_tot, bd_whole and silt_tot_psa associated with a named sample were imported.

- That ensured that the base sample data used for the computation are preserved regardless of any subsequent corrections to the source datasets used in the computation.
- That also ensured that the values could be populated in a data product (BWT).
- This also enabled the database to include these samples in the non-gap-fill computation for comparison purposes.

The soc, soc_type, and soc_method for each sample were also imported.

For each profile, the observation_date, soc, soc_type, soc_method, soc_lcount, and soc_depth were imported. For each site, the lat, long, datum, country and state were imported. These spatial locator metadata are imported to ensure that they are preserved and can populate data products.

Using that dataset as an exemplar, the best practices for an analysis dataset are:

- Include at least the lat, long, country and state for each site. The datum is assumed to be WGS84.
- Include the observation_date, soil_series, and soil_taxon for each profile.
- Include the layer_top, layer_bot and hzn_desgn for each layer.
- Include the carbon and bulk density data used to compute soc. This both insures that the values are known as well as enables a non-gap-filled computation to be performed by the database for comparison.
- Include a profile aggregate soc, soc_depth, and soc_lcount .The database does not aggregate investigator submitted layer soc values.

The first investigator-submitted analysis dataset is the Permafrost dataset.
This dataset consists of X data templates and a metadata sheet. The metadata sheet becomes the parent dataset for the other templates. The additional templates are staged as "analysis" templates associated with that parent dataset.

Open issues here are:

- The submission of soc_type and soc_method are currently optional. The metadata c_method is also optional. Should one or more of these be required?
- The free text soc_spatial_flag and soc_carbon_flag quality flags are currently internal only. Should they be exposed to other submitters ?

INFO: About overlay datasets

The NSCN database contained 3 overlay datasets:

- Landuse data from LBNL
- GPP and meterological data from MPI
- Omnerik ecoregion data from LBNL

Each of these was generated by reverse lookup of lat/long. Instead of migrating that data from NSCN, the data are going to be regenerated using the new lat/longs. The LBNL ecoregion data was used as the example dataset to understand the submissions process.

Overlay dataset templates contain two sheets – a metadata sheet and a lat/long sheet. The lat/long sheet is pre-populated to include four columns:

- overlay_name. This unique identifier is used by the staging logic to recognize the lat/long combination. The identifier is necessary to avoid the challenges associated with precision changes in numeric values across export/import.
- lat . Decimal latitude. This can be assumed to be in WGS84, although some older data may use another datum.
- long. Decimal longitude. This can be assumed to be in WGS84, although some older data may use another datum.
- datum. The datum string follows the datum CV.

The pre-populated columns should not be changed.

At submission, the lat/long sheet should include additional columns for each of the submitted overlay variables. The variable names and the mappings to variable groups must have been previously defined. Rows may be removed from the sheet if no lookup is possible; blank values in overlay variables may also be used. Strings such as "NA" or "Unknown" should not be used.

At staging, each lat/long/datum trio in the sheet is expanded to create a site_name, lat, long, and datum quartet. Each overlay data group is also expanded to include its "XXX_locator_dataset" variable. This variable tracks the parent dataset name of the lat/long/datum used to do the reverse lookup of the overlay ariable group. Like the overlay_name, the XXX_locator_dataset variable is used to avoid all of the precision issues associated with real number import/export of lat/long.

INFO: About plots

A plot is a spatial collection object between sites and profiles. The concept of plot appears in other ecological databases such as the FIA tree database.

The NSCN database implemented this concept as "cluster". A cluster was an optional spatial locator between site and profile. The only variable associated with a cluster was cluster_note – a free text description of the cluster. The internal implementation chose to define a default cluster to preserve a single spatial hierarchy.

Treating clusters or plots as an optional spatial locator interacts with and complicates the dataset handling. The fundamental issue is that clusters are collection objects defined by humans and not really spatial locators. Cluster groupings of profiles at sites may well differ across analysis datasets. Cluster are also inherently splines as a given site may be associated with multiple profiles.

The plan forward is twofold.

- First, we should call a plot a "plot" and not a "cluster". Hearing "plot" made me finally get why Luke wanted the concept.
- Second, we should treat plots as pseudo spatial locators – outside the enforced hierarchy with special spatial validation.

All plot information will be self-contained on the (TBD) plot sheet of the submission template. For each plot, the sheet will contain:

- the plot_name
- the plot_description
- the profile_name of the two or more child profiles

Note that the sheet layout may appear somewhat odd when more than one plot is included. My best idea to date is that the sheet has multiple rows with "profile_name" such that a given plot is entered into a single column.

The (TBD) plot sheet can be included in any data submission. Internally, the plot name and plot description will be treated as a site level grp_plot group. This group will include one plot_name, one plot_description, and any number of plot_name rows. (Note the similarity here with locator_alias.) This requires some special code at staging, but the code is relatively straightforward.

How plots are exposed to users remains an open question. One implication of the above is that any aggregation of variables across a plot in a BWT will require special code or be punted to the (somewhat skilled) data users.

INFO: BWTs and data products

A BWT or "big wide table" is the primary data product form produced from the ISCN database. A BWT contains submitted and computed data values at the relevant spatial locators from the relevant datasets at a specific point in time. All BWTs use the same internal mechanism for generation - an index table that defines the source variables, custom query generation, and cursor traversal.

The older NSCN BWTs are not dataset specific and do not include dataset information. The only provenance information is in the left most upper most cell which contains a version identifier.

The new ISCN BWTs include two additional columns that contain dataset information. These columns contain:

- The parent dataset name of the sample data submissions. This is the dataset that sourced the layer, carbon, and bulk density used in the carbon computations. It is also the dataset that sources any lat, long, country, state or observation_date.
- The parent dataset name of the carbon computations from that sample data submission. Here, the parent dataset identifies the nature of the algorithm used to compute carbon.

As an example of the above, consider an NRCS profile in Alaska. The same layer could generate profile carbon computed by the ISCN non-gap-filled algorithm, the NSCN non-gap-filled algorithm, the NSCN Alaska gap-fill algorithm, and the Permafrost analysis set.

As noted previously, the current plan for BWT generation does no variable filling. In other words, when producing a data product, we do not fill variables such as lat/long, soil_taxon, or c_tot when the parent dataset does not include those variables. While data filling seems desirable, how we can express what's actually going on to the user is a mystery to me. Expressing "I used data from the following datasets to create this dataset" as an English (or even Tagalog) language statement is relatively simple; expressing that same concept programmatically gets very complex and very subject to simple errors such as typos or cut/paste errors.

What this means for BWT generation can best be explained by using the above NRCS profile in Alaska as an example.

- If the NRCS 2014 database contained different values for lat/long than the 2011 database, the ISCN values will differ from the NSCN and Alaska carbon computation values.
- If the Permafrost analysis submissions did not include lat/long, that computation will not have lat/long. While it seems likely that the Permafrost analysis submissions depended on the NRCS 2011 values, that dependency isn't assured.
- If the Permafrost analysis submissions do include lat/long, those values may or may not match (to whatever precision) the values from the NRCS 2014 or NRCS 2011 dataset.
- If the NRCS 2011 database did not include soil_taxon, the NSCN and Alaska computations using that database will not have soil_taxon.

- If the NRCS 2011 Permafrost dataset included values for soil_taxon than the NRCS 2014 dataset, those values will may either match or differ.

To make the point even more strongly, consider what happens when an updated NRCS 2015 database is ingested as a correction to the NRCS 2014 dataset. Any new lat, long, and soil_taxon values will replace the older values. What value should be chosen to fill?

All of that being said, a user of the exported BWT data product will have all of the submitted values for lat, long, soil_taxon, etc for a given named spatial locator. That user can then make science informed choices across the different values. In other words, a way of thinking about the lat, long, and soil_taxon values is that they are metadata that inform the use of the actual profile carbon data.

An additional plan forward is to generate a "BWT creation log" table. That table would contain the version identifier, creation date, and contributing datasets associated with a given BWT generation from a BWT index table. To the extent to which this table is exposed to end-users is at your discretion.

- The intention is to enable regeneration of a specific data product at some point in the future without actually having to invest in the thinking and code to actually do so.
- This would also be a necessary step in the generation of alternate data products such as permafrost variants with filled values as it identifies the source of potentially retired (older, corrected) data.

Note that many of the rows in this table will be "internal-only" for code development versions that are subsequently abandoned or "LATEST" versions that are subsequently superceded.

Note to Deb and Luke: What does the download tool actually do with that left most upper most cell? If it's not exporting that cell, why have it? If it is, please think about what string you want that cell going forward.

Overlay variables such as ecoregion or mat_cru are filled when the relevant variable locator_dataset matches the parent dataset name. What's really going on here is that the parent dataset contained the lat/long value used to lookup the overlay variable.

INFO: Internal tables

This section provides a basic enumeration of the internal auto tables and their usage. Auto tables are used only internally and are programmatically generated. Exported tables are either Display tables used by web parts or BWT tables used for data products. Auto, display, and BWT table generation uses a number of temp tables for simple debug; these temp tables are always dropped and regenerated within the relevant stored procedure. Also, as noted elsewhere, the approach with the auto tables is to generate the worst-case combinations thereby enabling simpler aggregation and/or filtering.

The auto tables include:

- ancvarauto. This table contains all variable values.
    - The astring column contains the data value as submitted; this column is for reference only and should not be used for subsequent table generation.
    - The fstring column contains the formatted string for all variables. Formatting text variables applies the CV capitalization, converts CV short names to longer descriptions, etc. Formatting real or integer variables applies any precision (eg lat is 4 digits). This column should be used for any text output.
    - The datavalue column contains a real value for any real or integer variable. This column can be used for internal code computations (such as lat/long bounding boxes) to avoid any additional typecasts which may impact precision.
- sampleauto. This table contains the layer sequencing information. That includes the layer_seq, layer_indicator, and profile_indicator.
- samplecarbonauto . This table contains the layer carbon computation.
- profileauto. This table contains non-carbon profile information.
- profilecarbonauto. This table contains carbon profile information aggregated from samplecarbonauto.
- siteauto. This table contains non-carbon site information.

A bit of explanation with respect to the combinations of the above tables is handy.

- The ancvarauto table tracks the parent and submission dataset for the spatial locator AND the data value in each row. In other words, a row containing the lat for a site has up to four different datasets:
    - The submission dataset of the spatial locator. This identifies the dataset that first contained the spatial locator name.
    - The parent dataset of the spatial locator. If the spatial locator was added via a correction dataset or analysis dataset, this identifies that submission dataset.
    - The submission dataset of the data value. If the data have been corrected, the submission dataset identifies the correction dataset.

  The parent dataset of the data value. Like the parent dataset of the spatial locator, this identifies the original source of the data value. In general, all four datasets will be the same; the spatial locator and the associated data were submitted on a single template and have not been corrected.
- All other tables track only the parent datasets associated with the spatial locator and the relevant variable groups contributing to the table.

- The sampleauto table tracks the parent dataset of the grp_XXX (layer_top, layer_bot, and hzn_desgn) and the parent dataset of the sample name.
- The carbonauto table tracks the parent dataset of the XXXX. that feed into the soc_predict computation.
- The profileauto tracks the parent dataset of XXX that determine the profile depth and profile_indicator as well as XXX.

INFO: About staging data submissions

All data enters the ArchiveSoilCarbon database through the ISCNStage database. Data submissions are either standard Excel templates or special format Excel workbooks used for overlay dataset submissions. All data submissions must contain a metadata sheet. The format of each special format workbook is determined by the nature of the data; a common format for site information will have a site_name column and some number of data columns.

Data submissions follow the workflow:

- The excel file is uploaded to the ISCN web site
- A curator examines the uploaded file and determines whether the dataset is a new, overlay, analysis, or correction dataset. If the dataset is a correction dataset, the curator determines the parent dataset.
- Data are staged from the uploaded template to the staging database
- A number of data validity checks are performed
- A curator views the reports generated by those validity checks
- The curator works with the dataset curator to address any issues
- Validated data is pushed to the archive database
- The various internal data product machine-generated tables are regenerated in the archive database
- External data products (eg BWTs) are regenerated in the archive database and pushed to the ISCN web site

All submitted data pass through the same validity checks, curator review, push to archive, and regeneration of the data product tables.

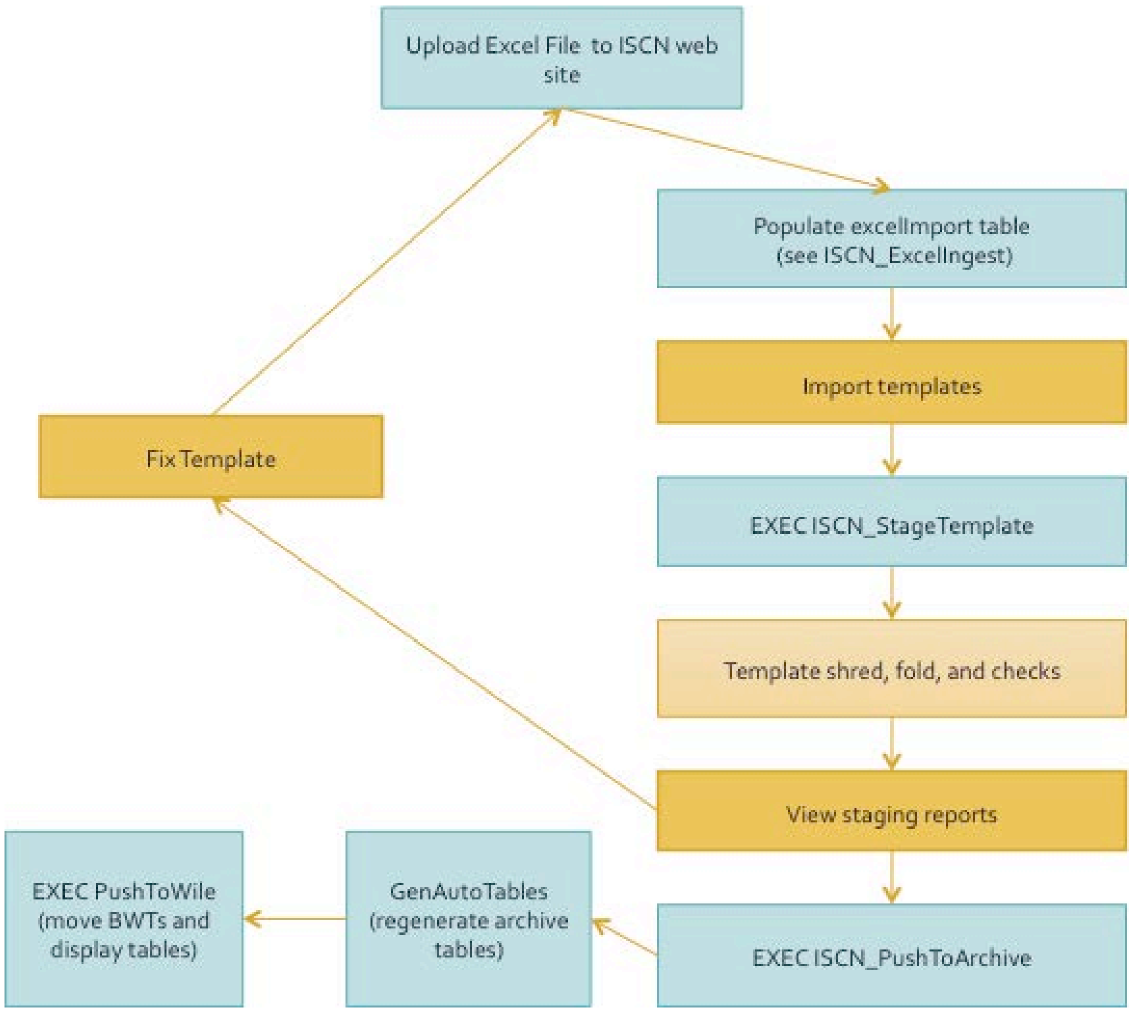

Upload Excel File to ISCN web site
Populate excelImport table (see ISCN_ExcelIngest)
Import templates
Fix Template
EXEC ISCN_StageTemplate
Template shred, fold, and checks
View staging reports
EXEC PushToWile (move BWTs and display tables)
GenAutoTables (regenerate archive tables)
EXEC ISCN_PushToArchive

Figure 1 Excel template processing

Best practices will be to stage any new submission as either a "special" dataset or a "new" dataset even if the dataset is an analysis dataset or curator template. This will cause the spatial locators to be checked within the submission as well as validated against the archive contents.

- Staging the submission as a special dataset will cause any differences in the spatial hierarchy as well as any new spatial locators to be flagged.
- Staging the submission as a new dataset will cause any existing spatial indicators to be flagged.

The generated reports will let the curator chose if and how to address any flagged locators. For example, consider an analysis dataset submission containing both new and old spatial locators. Staging the submission as a new dataset will flag all locators that already exist in the database. The curator can then decide whether that locator is the same (eg the same site at the same or close enough lat/long) or different (eg a sample with very different layer_top and layer_bot or with a very different profile parent). If the locator is "the same" the name collision should be expected where as if the locator is "different" the name must be disambiguated. Once the spatial locator names have been harmonized, the template can be restaged as "special", "new", or "analysis" as appropriate.

ISCN curators may also submit specialized "curator templates". Curator templates are distinguished by the iscn_curator_userid and iscn_curator_comment variables. Curator templates enable data curation tasks such as changing a site lat/long/datum in response to email from the tower team. Curator templates should include only those variables that are being updated and impact only one parent dataset.

The processing code makes very few assumptions about the contents of a submission other than the required provenance metadata is present. As noted above, all submissions must have an associated valid:

- dataset_name. This field is free text and must be unique (ie not exist in the archive database or other staging template). When Excel files are re-used, the dataset_name must be changed.
- dataset_description. This field is treated as free text.
- curator_name. This field is treated as free text.
- curator_email. This field must pass basic checks on email format.
- curator_organization. This field is treated as free text.
- modification_date. This field must pass basic checks on date format.

Submissions without the minimum metadata will not stage to the archive database. When submissions are staged as a group, any submission that fails will block all other submissions from staging to the archive database.

CHECK THIS: All submissions are tagged with a curatorUserID and curatorComment. The curatorUserId is determined by:

- The older Alaska data are tagged with "<KRIS JOHNSON>"
- The NRCS data migrated from the NRCS access database are tagged with "vaningen"
- Sciscope and other lookups done by me or my minions are tagged with "vaningen"
- All other submissions are tagged with "Nave".

The curatorComment contains the iscn_curator_comment text (if any) and any additional text generated by the processing code.

INFO: About databases

The document includes some high level descriptions of what's going on in the database behind the scenes as well as explicit instructions for specific tasks.

- The descriptions are specifically intended to help you interpret the various log entries and/or make judgments as to when to update what internal data product (aka "automated table" or "machine generated schema").
- The descriptions are specifically not intended to enable you to "fix up" something that "goes bad" by running some number of SQL queries. The good news wrt the current "variable group" implementation is that variable groups are A Very Good Thing and always staging data through a staging database means that the data in the archive database is likely to be more science ready. The bad news is that while the implementation is moderately robust when things go wrong, trying to manually correct that without taking the long way around and just restaging something is very likely to lead to something even more wrong. The additional bad news is that there are in doubt, run a backup of the ArchiveSoilCarbon database and be prepared to do a restore.

All tasks listed here are performed when logged into eddy.lbl.gov. Unless otherwise noted, the task is performed with a stored procedure (SP) in the ISCNStage staging database. The archive database is ArchiveSoilCarbon. Unlike the AmeriFlux BADM, the ISCN ancdecode and ancCV tables are maintained in-situ in the archive database. Like the AmeriFlux BADM, the ISCN ancdecode table is not edited manually but rather generated and validated programmatically.

The ISCNStage database is not backed up. All tables and all stored procedures (SPs) can be regenerated from svn (the SPs), the Archive database (local copies of tables such as the ancdecode table) or importing an Excel sheet. The staging database name MUST be ISCNStage for any of the SPs to operate correctly. Any migration decode tables (eg NRCS_Index or NASIS_Index) tables must be managed as Excel spreadsheets and versioned via svn.

The ArchiveSoilCarbon database is backed up. It holds "The Truth" for all datasets, spatial locators and data. The SPs are versioned with svn. You should run a private backup of this database before doing anything risky and/or anything about which you are uncertain. You cannot back out changes manually. Rolling back via a complete backup is the only safe practice. It can also save you a lot of time recovering as the backup takes very little time.

TASK: Staging an ISCN standard template

Templates can be processed individually or in groups. The excelImport table in the ISCNStage database drives the processing. Each row in that table guides the processing of a single template worksheet; in other words, a standard template with layer, profile, site and metadata sheets will have 4 rows.

TBD pending wile table. To view templates uploaded via the wile web site, use the UploadedTemplates report. This report views the contents of <TBD table from similar

to Ameriflux > on wile. The templates are ordered with the newest (most recently submitted) first.

TBD. To view templates that have been processed and pushed to the archive, use the ProcessedTemplates report. This report tags each template with the dataset and suppresses the NRCS, NASIS, and older Alaska database submissions.

Simplistically, the Excel template staging implements the following workflow:

- Set up the ISCNStage database
- Set up the excelImport table
- Import the template worksheets into the database
- Shred, fold, and check the imported templates
- View the reports
- Push the validated spatial locators and data to the archive database

The AGU2013A SP contains annotated sample queries necessary to implement this workflow.

To set up the ISCNStage database:

```
EXEC  [dbo].[ISCN_StageDBSetup]
```

This SP deletes all but a very few tables from the ISCNStage database and recreates the base tables necessary for processing templates such as the ancdecode table.

To set up the excelImport table for a new dataset template containing metadata, site, profile, and layer sheets, the following can be used:

```
EXEC [dbo].[ExcelImportGen]
 @excelPath = 'NSCNTemplate_PermafrostRCN_Harden_rev KLM.xls',
 @excelTag = 'Harden'
```

This generates one row for each sheet (4 rows) in the excelImport table. The row for the metadata sheet assumes row orientation; the rows for the site, profile, and layer sheets assume column orientation.

The same SP can be used to generate a single row corresponding to a single template sheet.

```
EXEC [dbo].[ExcelImportGen]
@excelPath = 'NSCNTemplate_PermafrostRCN_Harden_rev KLM.xls',
@excelTag = 'Harden'
@excelSheet = 'site',
@excelOrientation = 'column'
```

The above should be used to add excelImport table rows when processing a new dataset template containing disturbance, gas, fraction, or treatment sheets. Note that including the @excelOrientation is required to indicate whether the imported sheet has column or row orientation.

When submitting a correction or analysis dataset, the additional @submitType and @parentDatasetName parameters should be included.

```
EXEC [dbo].[ExcelImportGen]
@excelPath = ‘NSCNTemplate_PermafrostRCN_Harden_rev KLM.xls ',
@excelTag = 'Harden',
@submitType = 'analysis',

EXEC [dbo].[ExcelImportGen]
@excelPath = ‘NSCNTemplate_PermafrostRCN_Harden_rev KLM_update.xls',
@excelTag = 'Harden,
@submitType  = 'correction ',
@parentDatasetName = 'Permafrost Harden'
```

When in doubt, see the comments in the SP for an explanation of the parameters.

Note that the @excelPath becomes the submitgroup provenance information for each group. As such, ensuring that the file name is unique or can be tracked to a unique name in some other robust fashion is strongly recommended. The AmeriFlux web site appends a unique identifier to all uploaded files; I recommend that the ISCN web site do the same.

The SP generates the full command line for the Excel2DBingester tool used to import each sheet. The command line generated by the above invocation for the metadata and site sheets are:

```
Excel2DBingester.exe "NSCNTemplate_PermafrostRCN_Harden_rev KLM.xls"
-T ISCNStage Metadata_Harden metadata
Excel2DBingester.exe "NSCNTemplate_PermafrostRCN_Harden_rev KLM.xls"
ISCNStage Site_Harden metadata
```

The syntax is:

```
Excel2DBingester.exe "<excel file name>"
<orientation>  <database> <import database table> <excel worksheet>
```

Where <orientation> = "-T" indicates a row-oriented sheet and <orientation> is blank indicates a column-oriented spreadsheet.

The command lines can be retrieved by:

```
select importtext from excelImport
```

You can also simply generate the command line directly. A few notes about Excel2DBIngester execution:

- Excel2DBingester may throw a popup informing you that there is a lot of material on the clipboard for each imported file. You MUST click the "yes" (I want to save the information).
- The most common causes of Excel2DBIngester crashes are typos in the file name or sheet name (eg "sitefoo" rather than "site")
- Templates that have a lot of entries in lower sheet rows can also cause a crash. An example is a Site sheet with 20+ REFERENCE_PAPER entries. Transpose the sheet to column orientation and try again (specifying the @excelOrientation = 'column' and/or omitting the "-T" argument in the above command line.)
- Excel2DBIngester also hangs when warning conditions are encountered. You must terminate execution with Control C and then fix the template.
- Blank characters in the first column will cause import to hang. You will see an output message of the form:

  ```
  Warning: Header name is blank in worksheet: site at: $P$1
  Warning: Header name is blank in worksheet: site at: $P$1
  ```

  Delete a large number of columns/rows following the offender and re-import the template.
- Duplicate Variable strings in the first column will cause import to file. You will see an output message of the form:

  ```
  Warning: Duplicate header name(s) found in worksheet Site for name:
  landscape  at : $P$1
  Warning: Duplicate header name(s) found in worksheet Site for name:
  landscape E at : $Q$1
  ```

  In the above case, the template submitter inserted two rows with landscape in the first cell (3 such rows in all) rather than using successive columns.
- Executions of Excel2DBIngester can leave "zombie" Excel processes. These won't be a problem when importing a few templates, but either you need to login/logout or use taskmgr to kill these off when importing 10+ templates particularly if you encounter a crash. In other words, if you encounter a crash when importing what otherwise looks like a correct template, make sure you don't have zombie processes locking relevant files.

To shred, fold and check the imported templates:

```
EXEC [dbo].[ISCN_StageTemplate]
```

This processing:

- shreds each non-blank Excel template data cell into the importvar table
- appropriately groups the contents of the importvar table into the importgrp table
- extracts all spatial locators into the importsite table
- constructs the importdataset table containing provenance information on each dataset
- executes a number of variable, group, spatial locator and metadata checks

Consider a dataset "My house" with two sites "front yard" and "back yard". Each site has a lat, long, datum, and elevation.

- The importdataset table will contain one row for "My house".
- The importsite table will contain three rows – one each for the "front yard" and "back yard" sites and one for the "My house" dataset.
- The importgrp table will contain six rows.
    - One row containing grp_dataset_description which locates the required dataset_description row in the importvar table associated with "My huose"
    - One row containing grp_curator which locates the required curator_name, curator_email, and curator_dataset rows in the importvar table associated with "My house"
    - Two rows containing grp_latlong which locate the rows in the importvar table associated with the lat, long, and datum for "front yard" and "back yard"
    - Two rows containing grp_elevation which locate the rows in the importvar table associated with the elevation for "front yard" and "back yard"
- The importvar table will contain
    - One row each for dataset_description, curator_name, curator_email, and curator_organization associated with "My house"
    - Eight rows containing the lat, long, datum and elevation for "front yard" and "back yard"

After staging a template, you proceed to the checking a staged submission tasks and then the pushing to the archive task.

When staging multiple templates as a group, the query below can be used to check that all of the Excel worksheets identified in the excelImport table have, in fact, been successfully imported. If any rows are returned, there was some sort of error importing the file with Excel2EBIngester attempt. You should resolve this issue and then rerun the shredding and data checks by re-executing ISCN_StageTemplate.

```
select importid, 'failure to import ' + importfile
from excelimport where importtable
not in (select name from sysobjects
where OBJECTPROPERTY(id, N'IsUserTable') = 1)
```

INFO: Checking staged spatial locators

The template staging code runs a number of checks on the spatial locators and tags any locator that fails those checks as HOLD. The checks are common to all staged datasets. The ISCN curator is responsible for correcting the problem or identifying the HOLD as coding bug.

Spatial locators collide whenever the locator name recurs at the same spatial locator level. Collisions may happen between a template and the archive or between multiple templates staged as a group. Collisions may be expected or unexpected. Expected collisions will push to the archive without curator action. The curator must resolve all unexpected collisions. Consider a template with a "front yard" site_name:

- If the "front yard" string appears in more than one row on the site template sheet, the collision is unexpected and the rows must be disambiguated (including any dependent profiles and layers).
- If the "front yard" string also appears as a profile_name, there is no collision.
- If the "front yard" string exists as a site_name in the archive database and the submission is a new dataset, the collision is unexpected and a new name must be chosen.
- IF the "front yard" string exists as a site_name in the archive database and the submission is a correction or analysis dataset, the collision is considered expected.

Each importsite row is tagged with a "resolved" variable. The "resolved" variable indicates what will happen if and when the contents are pushed to the archive database. Resolved values associated with spatial locators (sites, profiles, samples, and fractions) include:

- ACCEPT: the spatial locator name does not collide with the archive or other staging template and the dataset is not a special dataset.
- REJECT: this spatial locator is used only for NRCS migration. The locator has been folded with another locator due to missing or duplicate metadata.
- EXISTS: the spatial locator name collides with the archive and dataset is an analysis, correction, or special dataset.

- HOLD: the collision is unexpected and must be resolved. The spatial locator name collides with the archive or archive or other staging template and the dataset is a new dataset OR the dataset is a special dataset and the site name does not exist in the archive.

Note that best practices are that special datasets are not staged only with other special datasets and not new or analysis datasets.

The recommended way to resolve a collision is to postfix a short dataset identifier to the new colliding name. (Eg "8K236920" becomes "8K236920:PENG". If the locator really is the same as the colliding locator then the staging dataset is not actually a new dataset and there are larger issues to be resolved.

The parent:child relationships within the spatial hierarchy are also checked.

- New spatial locators with no children generate informational log messages but may be staged at the curator's discretion. This may or may not be an error and only the curator can decide.
- New spatial locators with no parents in a new dataset generate errors and are tagged as HOLD. A human must resolve this. For example, each profile_name on the layer sheet must have a corresponding row on the profile sheet.
- New spatial locators with parents that exist in the archive are allowed in analysis or correction datasets. For example, a new profile_name on the profile sheet associated with an existing site_name does not require that the site sheet have a row corresponding to that site_name.
- Existing spatial locators must not have conflicting parents. The spatial locator is tagged as HOLD. For example, if the archive contains a profile "front yard 1" associated with the site "front yard", and a template contains a profile "front yard 1" associated with a site "new front yard", the "front yard 1" profile will be held. This is most likely due to confusion (eg inadvertent reuse of the profile name "front yard 1") or typos (eg "fornt yard").
- A special dataset must only contain existing spatial locators and must not introduce parent conflicts.

CvI note: review implementation code per above. In particular, the existing parent needs care when we get to the (updated) Permafrost set.

INFO: Checking staged data

The template staging code runs a number of checks on the staged data and tags any group that fails those checks as HOLD. The checks are common to all staged datasets. The ISCN curator is responsible for correcting the problem or identifying the HOLD as coding bug.

Each importdataset row is tagged with a "resolved" variable. The "resolved" variable indicates what will happen if and when the contents are pushed to the archive database. Resolved values associated with datasets are:

- ACCEPT: the dataset is a new, special, or analysis dataset, has minimal required metadata and the dataset name does not collide with any other dataset in the archive or current staging dataset. If the dataset is a correction dataset, the parent dataset also exists in the archive or the current staging datasets. All is well.
- HOLD: the dataset name exists in the archive, lacks required minimal metadata, or the dataset is a correction dataset and the parent dataset does not exist in the archive or current staging dataset.

A variable group is valid when all of the required variables are present and each variable passes the quality checks. A few examples of invalid groups:

- lat is specified, but long is blank.
- lat and long are specified, but lat = "190".
- datum is specified as "NAD 83"
- The same contact_name appears twice on the metadata sheet.

Variable group collisions are central to processing correction datasets. The new variable group will replace the colliding archive group. To collide, variable groups must be from the same parent dataset, associated with the same spatial locator, have the same variable group type and trigger the variable-group specific collision rules.

- A simple example is a correction of lat and long at a site; the new submitted correction value collides with the archive value.
- An example of a non-collision is submission of new fire disturbance at a given site on a different date.

Each importgrp row is also tagged with a "resolved" variable. Resolved values associated with data variable groups include:

- ACCEPT: the variable group is valid and does not collide with any other variable group in the current staging datasets.
- DUPE: the variable group is valid and collides with another variable group in the archive within the parent dataset. In this case, the variables and variable values in the submitted group match those in the colliding archive group. The new group is staged for tracking purposes. The original colliding group is retained as the source for all data products.
- REPLACE: the variable group is valid and collides with another variable group in the archive within the parent dataset. In this case, the variables and variable values in the submitted group do no match those in the colliding archive group.

The original colliding group is retired and the newly staged group becomes the source for all data products.
- HOLD: the variable group is invalid or causes an unexpected collision within the current staging datasets.
- DELETE: this variable group should be deleted due to curator action. Note that there is currently no tool for this.

Note that DUPEs and REPLACEs can only happen in correction datasets where the submission matches an existing parent dataset. Submission of the same variable group in two different analysis datasets does not cause a collision; both groups will be ACCEPTed if otherwise valid.

You should review any variables and/or groups that indicate collisions to make sure that the subsequent push to the archive will do what you intend.

TASK: Using the staging reports

To check a stage, following reports can be used. The order below is a suggested order in which you view the reports.

- CheckSummaryStats lists the total number of variable groups (“data”) and spatial locators (“locator”) at each spatial level for each resolved value for each template.
    - This report is intended to let you rapidly identify template sheets with issues that must be addressed prior to the push to archive. Spatial locator issues such as colliding names, duplicate names or missing parents are indicated by a “locator” row with HOLD resolved. Data issues such as CV violations, data range issues or missing required parameters in a variable group are indicated by a “data” row with HOLD resolved .
    - This report will also let you rapidly identify missing or blank template sheets. A blank resolved spatial locator dataset cell indicates that the dataset name is missing or blank on the metadata sheet or the metadata sheet is missing. Any other blank or resolved spatial cell indicates that a new spatial locator was detected on a dependent sheet and the source sheet is missing or blank. For example, the layer sheet contains a profile_name column and there are strings in that column that do not match the archive database names but the submission did not include a profile sheet.
- CheckImportSummary groups log messages and includes a counter of the for each unique log entry. When staging very large templates, this report allows you to see the most common issues. For example:
    - a counter of 16 with message of “ERROR: data not in CV state_province” indicates that there are 16 rows on the identified site sheet containing

non-blank strings for STATE or PROVINCE that do not match the state_province CV. Either the strings must be corrected or the CV must be expanded to include the new strings. All ERRORs must be fixed as any will cause the entire variable group to be on HOLD.

  - a counter of 3900 with message of "INFO: lat/long precision" indicates that there are 3900 rows on the identified site sheet with lat/long with precision less than 4 digits. Correcting INFO messages is at the discretion of the curator; INFO does not cause the variable group to be on HOLD. NOTE This assumes we convert the current HOLD to be INFO only.

- CheckImportLocatorSummary allows you to review only the HOLD spatial locators. For example a row with a count of 2, name of "83-11M23,100" and curatorcomment string containing "name collision within dataset" at the sample level indicates that "83-11M23,100" appears twice as a sample_name in the identified template. All reported issues must be addressed and the templates restaged.
- CheckCV enumerates all errors associated with CV variables. Each row contains the rowtext, relevant CV and offending datavalue. Either the datavalue string must be corrected or the CV expanded.
- CheckTemplate allows you to review a single template sheet. All variables and all variable groups are listed with the associated group "resolved" and comments in the event of an issue with the group or variable. Again, note that a group will be held if any of the variables within the group fail validity checks.
- CheckImportLog is the more detailed and verbose counterpart to CheckImportSummary. Each message is displayed. Note that this can become long (and cause the report rendering to crash) when staging large templates or large templates groups. In that case, view this report after correcting most/all of issues in the CheckImportSummary report as it also contains any additional issues such as code bugs not associated with a given template. The report includes a rowid column. For smaller Excel templates, the SQL rowid will be nearby the Excel row containing the issue. Unfortunately, there is no guaranteed of ordering between Excel and SQL.
  - Site locator issues will have blank rowtext strings. The datavalue column contains the site locator string.
  - Data issues will have a valid rowtext and datavalue pair.

  Again, any ERROR issues must be corrected prior to staging to the archive. Addressing INFO issues are at the discretion of the curator.
- CheckCV enumerates all errors associated with CV variables. Each row contains the rowtext, relevant CV and offending datavalue. Either the datavalue string must be corrected or the CV expanded.
- ChecknonCV enumerates all errors not associated with CV variables. Errors include:
  - bad numeric values for REAL or INTEGER variables. Examples are "100+" or "30-60")

- submitting an optional parameter variable without submitting the primary variable. An example is submitting soc_depth without submitting an soc value or submitting acknowledgement_usage without acknowledgement.

Additional handy reports off
http://eddy.lbl.gov/Reports/Pages/Folder.aspx?ItemPath=%2fISCN&ViewMode=List
may also be handy. Reports may be ugly, but adding something "close" tends to be simple, so please feel free to ask.

Note that these reports are sourced from the ISCNStage database. Cleaning the database will remove all data from those reports. In other words, if you are about to edit a template to correct data errors, EITHER save a copy of the relevant reports to static files (Excel or PDF) OR make all of your edits before attempting to restage your edited template. Current best practice is to upload the edited template at all times.

When all staged templates pass all checks, you can push to the archive.

TASK: Pushing to the archive

The push to the archive database should only be attempted if all of the staged spatial locators and all staged data has passed all checks. In particular the push will fail if there are any groups with resolved = "HOLD". Prior to pushing to the archive, you should run a private database backup. To push to the archive:

```
EXEC [dbo].[ISCN_PushToArchive]
```

This report stages the datasets, sites, variable groups and variable values to the archive database.

After executing this SP, you should again view the CheckImportLog report. You should see three INFO rows giving you maximum identity columns in the archive database tables prior to and after the archive push as well as predicted counters of staging rows. If you see any other rows, particularly a row with "ERROR: Archive <var/grp> push count mismatch", there was a problem with the push. Please contact me (and include the location of your backup).

If your stage includes a correction dataset with variable group replacements, you should do a at least one spot check of a replaced group. Make sure that any existing colliding group has been retired appropriately.

INFO: Staging spatial locator check details

Spatial locator hierarchy checks include:

- Does the locator appear more than once at the same level in the hierarchy in the same dataset? This is an error – the same name is used more than once – and must be corrected.
- Does the locator name already exist in the archive database? This may or may not be an error.
    - If the dataset is a new dataset, this is an error. The locator is a new locator and the name must be disambiguated and the template must be resubmitted.
    - If the dataset is an overlay dataset, no new locators other than the dataset locator are permitted. All other locator names must exist in the database.
    - If the dataset is an analysis dataset, this may or may not be an error. The correction may intend to add a new locator to the new analysis dataset. This case is flagged as an informational message to the curator as only a human can determine intent. If the intention is to add the locator, the submission can be processed. Otherwise, the name should be disambiguated and the template resubmitted.
    - If the dataset is a correction dataset and the locator name belongs to the parent dataset, this is not an error. The submission has new/updated data for the locator.
    - If the dataset is a correction dataset and the locator name does not belong to the parent dataset and the parent dataset is not an analysis dataset, this is an error. The locator is a new locator and the name must be disambiguated and the template must be resubmitted.
    - If the dataset is a correction dataset and the parent dataset is an analysis dataset, this may or may not be an error. The correction may intend to add a new locator to the analysis dataset. This case is flagged as an informational message to the curator as only a human can determine intent. If the intention is to add the locator, the submission can be processed. Otherwise, the name should be disambiguated and the template resubmitted.
    - IMP NOTE: WATCH OUT HERE for the case of a correction to analysis when you do that check.
- Does the locator appear more than once at the same level in the hierarchy within the same processing collection? This can occur when multiple datasets are staged together. This may or may not be an error.
    - The collision may be due to different datasets choosing the same new name for multiple different locators. In this case, the curator must choose a "winner". The winner dataset can be staged immediately. The loser datasets must be edited to disambiguate the name and then staged. Note that once the names are disambiguated, the datasets can again be processed together.
    - The collision may be due to different datasets reusing the same name for the same locator. One of the datasets is a new or correction dataset, all

    others are analysis datasets. Here too, the curator must chose a "winner". The winner dataset can be staged immediately. The loser (analysis) datasets can then be staged. Note that a minimum of two processing stages is necessary.

- Does the locator have a parent (higher level in the hierarchy)? All spatial locators must have a parent.
    - The most common case is that the parent is submitted with its children as a new dataset. All locators must have parents. The parent of a site is the dataset.
    - If the locator is submitted as part of an analysis or correction dataset, the parent may exist in the archive database but not the submitted template. An example of this is a new profile at an existing site. This case is flagged as an informational message to the curator as only a human can determine intent. If the intention is to add the locator as a child to the parent, the submission can be processed. Otherwise, the locator name should be disambiguated and the template resubmitted.
    - If the dataset is an overlay dataset, no new locator names other than the dataset name are permitted.
    - Note that the cluster level is optional. Sites may have both cluster or profile children
- Does the locator have a child (lower level in the hierarchy)?
    - The most common case is that the parent is submitted with its children as a new dataset. Any parent locator without a child is flagged as an informational message to the curator as only a human can determine intent. If the intention that the locator does not have a child, the submission can be processed. Otherwise, the child locator should be included and the template resubmitted. An example here might be submission of profile carbon without associated sample data.
    - If the locator is submitted as part of an analysis or correction dataset, the child may exist in the archive database but not the submitted template. This case is flagged as an informational message to the curator as only a human can determine intent. If the intention is to add the child to the existing parent, the submission can be processed. Otherwise, the parent should be added and the template resubmitted.
    - If the dataset is an overlay dataset, no new locator names other than the dataset name are permitted.

INFO: Staging data quality check details

The new staging code shares a number data validity checks with the AmeriFlux/Fluxnet BADM data. These include both individual variable checks, group checks, and checks with respect to the current archive contents. The code also includes a number of spatial locator hierarchy checks unique to ISCN. The variable collisions logic is also unique to

ISCN. The CheckAllDriver SP in the ISCNStage database drives this process. New checks are added as distinct SPs. Each SP should be coded to be independent of all others. Use of temporary tables is encouraged.

Individual variable checks include:

- If the variable has a controlled vocabulary (CV), the submitted data follows the vocabulary
- If the variable is numeric (real or integer), the submitted data is a number. Example of not a number include "N/A", "~3" and "<4.5"
- Lat/long submissions must meet at least 4 digits of precision.
- URL variables must have minimally correct form.
- Email variables must have minimally correct form.
- Dates must be convertible to the form "YYYY-MM-DD HH:MM" and must not be prior to 1800 or in the future. Dates must also have appropriate precision (eg "YYYY" or "YYYY-MM-DD" but not "YYYY-MM"). Note that this check includes any necessary conversion from internal Excel format.
- If there is an associated DATE_UNC (reported in integer days), the date must not have HHMM granularity.
- References must not have embedded DOIs as there is a separate variable.
- For any numeric values with defined min/max, the submitted data must fall within that range. An example is lat which must be in the range -180<= lat <= 180.
- SIGMA error parameters must pass minimal checks. Nave owns definition (if any) here.

Any variable check that fails will cause the variable group to be tagged as HOLD. All HOLDs must be corrected prior to staging to the archive database.

Variable group checks include but may not be limited to:

- Is there at least one primary variable in the group? In other words, the group has only dangling parameters.
- Are all required parameters present?
- Is there one and only one entry of a comment or other optional variable such as _note with multiple other entries? This check does not generate an error, but does generate an informational warning. The issue here is the following. Consider a template with multiple entries for a single variable. The first data column contains an _note variable that seems to be relevant to all of the data in the successive columns. Yet the template submitter entered only one data cell. This check flags this for curator action.
- Are the minimum suggested metadata variables present ? This check is only informational and checks that:
    - Sites have lat, long, and datum

- Profiles have observation_date
- Layers have hzn_desgn, layer_top, layer_bot
- Gas samples have
- Other samples have
- ??

Failure to pass any of those checks will cause the value to be tagged appropriately and the associated group will be tagged as resolved = HOLD due to bad data value or parameter.

Collision checks drive the variable retirement logic.

Note again, a new group is accepted in its entirety with no merge between the old and the new.

Variable group checks include:

Get something somewhere about checking site names for subset collisions. Advisory only.

Submissions with respect to the archive checks include

INFO: Archive database checks

The ancdecode table is currently generated by code and should not be manually edited. At some point in time, that will change. Whenever the table changes, executing CheckAncDecode should be run to check the table contents. This SP performs a number of validation checks and logs any issues in the generatelog table. In the event of any issues, the ancdecode table should be fixed ASAP as subsequent processing will result in garbage.

The XXX SP validates the spatial locator hierarchy; it is run after each archive database stage as part of the machine schema auto table regeneration. This SP checks for missing parents, missing children and potential duplicate spatial locators. Missing parents should not be permitted by the staging code; any observed cases must be manually fixed (if only by restoring an earlier archive database backup and selectively staging datasets). The SP maintains a list of known missing children; the SP should be updated after staging a new missing dataset with missing children. Potential duplicate spatial locators include:

- sites with very similar lat/long and similar names
- profiles with the same observation date at sites with similar names

Any potential duplicates are logged as informational entries to the generatelog. The intention here is that a human curator can review the log and take action as needed. An example of such action is contacting the submitters of two datasets with potentially similar sites and resolving whether the sites are, in fact, the same. Once such a determination has been made, TBD tools for correcting the database can be used.

Note: TBD tools here means "I'm not coding them until we have a test case".

TASK: Archive Database auto-generated table regeneration

After each push to archive or new site addition, all XXXauto, XXXDisplay, and BWT_XXX tables in the ArchiveSoilCarbon database should be regenerated. The auto tables hold key combinations of spatial locators and their characterization data and are used to generate the BWT and Display tables. The Display tables are used by the website. The BWT tables are exported data products.

Regeneration of these tables happens by invoking:

```
EXEC [dbo].[GenAutoTables]
```

Note again that this must be executed from the archive, not the staging, database.

After regeneration, the Display and BWT must be moved to wile. This is a partially manual process because we've had a fair amount of breakage on both sides (eg. failure to regenerate a display table, web part hangs due to changes in data). To move regenerated tables from eddy to wile, execute:

```
EXEC   [dbo].[MoveTablesToWile]
```

That SP only generates the queries necessary to update the wile tables. *It does not actually update the wile tables.* You must cut/paste the generated queries into a query window and then execute those queries. Execute each query individually. If you encounter an error, stop and do not execute any remaining query. Always update all tables (even though you may not think there are any relevant changes).

INFO: ISCN reports

The reports that are intended for curator use are described in the above sections. The other reports are intended primarily for code debug. Reports are in svn under Ameriflux\SSRS\InternationalSoilCarbon.

Note that these are quick and dirty reports. There are known cases where multiple data errors can create multiple confusing log entries, group tags, and variable tags. When in doubt, look at the relevant part of the template and correct the errors you see.

TASK: Add terms to the ancCV table

Changes to the ancdecode and/or ancCV tables CANNOT be made by edit of the existing tables in the ArchiveSoilCarbon database.

New terms are added to an existing vocabulary by adding rows to the ancCVNew table in the ArchiveSoilCarbon database. Note that this table includes the same sequence number as added to the Ameriflux/Fluxnet ancCV table. At present, all sequence numbers are NULL so the terms can only be sorted by shortname (the default) or description. When adding a new term to a CV, make sure you include the description.

When adding multiple terms, a new vocabulary, or resequencing multiple terms, you should rename the ancCVNew table to a temporary name (eg CVsave_CvIMay10) and then export the existing table into Excel. Make any edits in Excel, and reimport the new vocabulary as the ancCVNew table.

Rows should NOT be removed from the ancCVNew table. There is no existing way to retire "old" terms in a CV. At a minimum, such a tool would have to check for existing data that used the older terms.

After adding rows to the ancCVNew table, execute the relevant queries in the SP:

```
EXEC [dbo].[UpdateAncdecode]
```

This regenerates both the ancCV and ancdecode tables and runs a set of quality checks. In particular, this checks for duplicate CV terms, unused CV vocabularies, or other potential typos when editing the CVsave table.

Make sure you view the generation log and address any issue that is not tagged as "INFO: ".

```
select * from generatelog where reason <> 'INFO: Missing explanation'
```

TASK: Update the ancdecode table

Changes to the ancdecode and/or ancCV tables CANNOT be made by edit of the existing tables in the ArchiveSoilCarbon database.

Text changes to the ancdecode description or explanation fields can be made by changing the contents of those columns in the ancdecodeSave table in the ArchiveSoilCarbon database. To resequence variables, you can do the same export to excel, edit, import from excel process as when resequencing multiple ancCV terms.

After making any text changes, execute:

```
EXEC [dbo].[DBUpdateDecode]
```

Note that regeneration of the ancdecode table causes a number of (existing) variables to be deleted and then regenerated per policy. One of the things this means is that when you resequence rows, you should not use monotonically increasing sequence numbers. Try to space variables by at least 10. Also avoid using sequence numbers less than 10 to reserve space for the header.

All other changes to the ancdecode table require code changes to the DBUpdateDecode SP and the associated CheckAncDecode SP. Such changes specifically include adding a new CV, adding a new variable to an existing group, or adding a new variable group. It also includes altering min/max values, retirement rules, and whether a variable is required or optional. Code changes here often (but not always) require companion code changes to the ISCN staging validation code (ie the Checkxxxx routines in the ISCNStage database).

Make sure you view the generation log and address any issue that is not tagged as "INFO: "

```
select * from generatelog where reason <> 'INFO: Missing explanation'
```

I continue to add additional validity checks to the CheckAncdecode CV as we process data and as I think of additional checks. The long-term intent is that the check code can be run on the ancdecode table in the archive database at any time so that we can go back to simple edits of the table.

After successful regeneration, you must reset the staging database.

TASK: Standard ISCN template generation

ISCN templates are generated by manual cut, paste, optional transpose, and column size adjustment of queries of the ancdecode table in the ArchiveNationalSoilCarbon database. Only blank templates can be generated at this time. That means that a correction template should start with the original submitted worksheets.

The query below can be used to generate the standard metadata sheet.

```
select rowtext, variabletype, display, units from ancdecode
where sheet = 'M' and reportingtype = 'template'
order by sheetsequence
```

The metadata sheet is row-oriented; no transpose is necessary.

To generate other sheets, substitute different letters in the "sheet = 'M' " clause as summarized below.

template	Sheet query clause	Transpose?
metadata	sheet = 'M'	no
site	sheet like '%S%'	yes
profile	sheet like '%P%'	yes
layer	sheet like '%L%'	yes
gas	sheet like '%G%'	yes
other	sheet like '%O%'	yes
disturbance	sheet like '%D%'	no
treatment	sheet like '%T%'	no

To generate a curator template, the CURATOR_USERID and CURATOR_COMMENT rows must be included on the metadata sheet. The query below can be used:

```
select rowtext, display, units from ancdecode
where (sheet = 'M' and reportingtype = 'template')
or rowtext in ('curator_userid', 'curator_comment')
order by sheetsequence
```

INFO: About customized templates

New groups are created in the database whenever a validated primary variable is submitted accompanied by its validated required parameters (if any). Replacing a variable group is an all-or-nothing operation: the old group is retired in its entirety and the new group is accepted in its entirety with no merge between the old and the new.

Customized Excel template generation operates independently of the staging validation logic. When a template is generated, all of primary variables and their required parameters must be included. When a group has a lot of primary variables, omitting some of them from the template can easily lead to confusion. The isrequired column in the ancdecode table identifies all primary and required variables in each variable group.

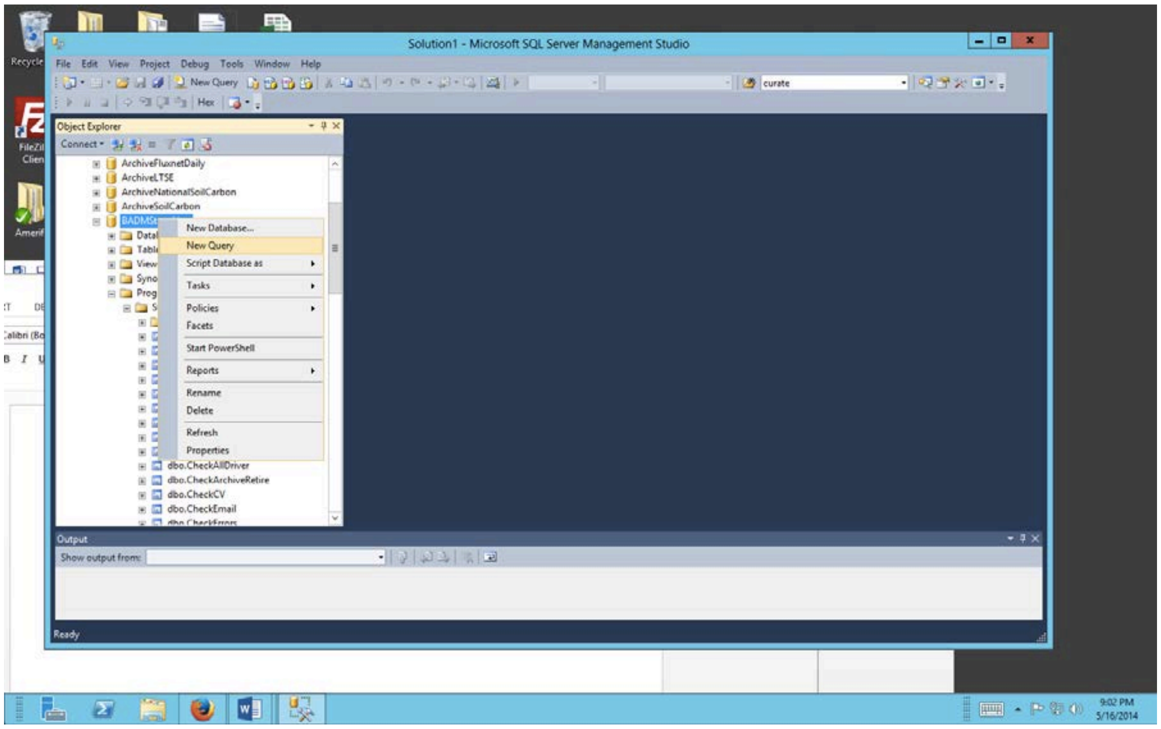
There are also a number of special cases that are not fully captured by the isrequired column. If the keepValue contains "Special", the group has one or more special case requirements. When generating a template, all variables in the group should be included in all generated templates.

TASK: Run a SQL query

To run a SQL query, you need to open a query window in SSMS (SQL Server Management Studio). If you don't have a shortcut to SSMS, you can create one via the Windows Application Search facility. The entry to this is above the Start menu that appears when you wave your mouse in the upper right corner.

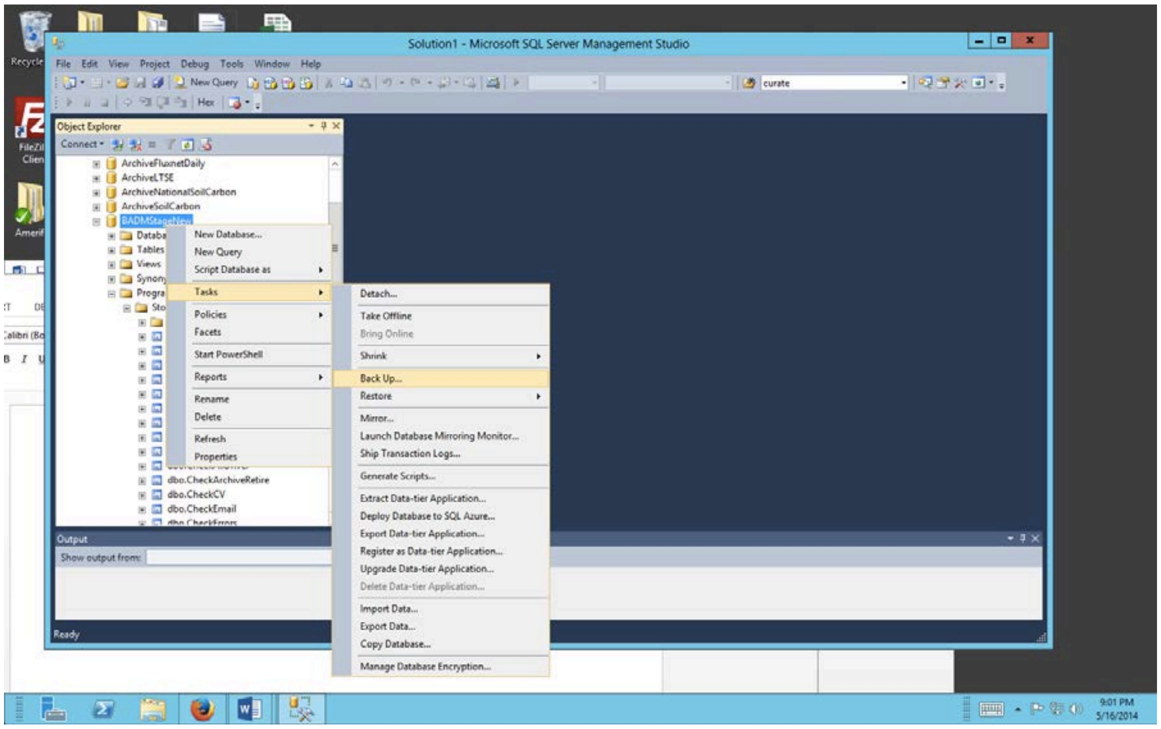
After opening SSMS, you first need to connect to the database server. You should see a small popup. All databases you need for ISCN management tasks are on eddy. Once you are connected to the server, you can right click on the Database list element to expand the list of databases. Right click on the database you want to use and select "Run Query".

SQL queries are case insensitive. You can execute *all* of the queries in your query window or you can highlight a given subset of those queries. The lower pain in the display is the results pane. You'll see the returned rows (although 40K+ rows can take a while) or any errors.

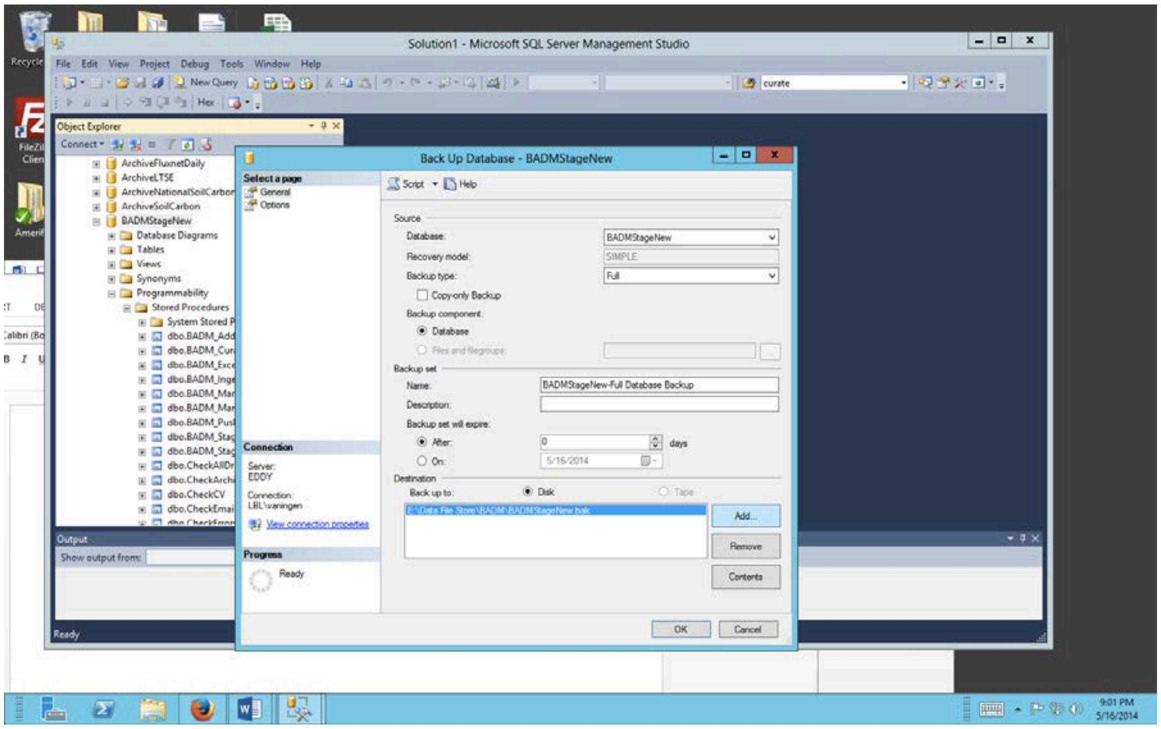


TASK: Run a SQL backup

Right click on the database name (eg ArchiveSoilCarbon). Follow the Tasks menu and select Backup.



That will bring up the Back Up Database popup. You need to change the target file type (see "Back up to" by first removing the existing target then selecting a new one. This prevents your backup from interfering with the regular scheduled (differential) backups.



Back Up Database - BADMStageNew

Source
Database: BADMStageNew
Recovery model: SIMPLE
Backup type: Full
[ ] Copy-only Backup
Backup component:
[● Database] ○ Files and filegroups:

Backup set
Name: BADMStageNew-Full Database Backup
Description:
Backup set will expire:
[● After: 0 days] ○ On: 5/16/2014

Destination
Back up to: [● Disk] ○ Tape
E:\Data File Store\BADM\BADMStageNew.bak

Connection
Server: EDDY
Connection: LBL\varingen

Progress
Ready

Backups take a few minutes.

Locators (e.g. the site 01 CL) and locator data (e.g. the lat, long, elevation, and datum in the grp_location for site 01 CL) go through different checking. ACCEPT means "ready to go"; HOLD means "nope, gotta be fixed".

Locators go on HOLD when the same name is used more than once, w

I had a bit of time, so I just reviewed the doc and the code.

No wonder you're confused.

The doc says that the lat/long precision check is INFO only and does not HOLD the group. The code HOLDs the group and logs an ERROR.

Now, another observation about the Stangberger and other templates.

I staged this and all other templates as "new". Per the doc and per the code, that means that all spatial locator names must be unique at a given level. In other words, when a site_name of "345" is encountered, that same site_name cannot exist more than once on the site sheet and cannot exist in the database.

I believe that’s the always right first thing to do as it checks for any potential collisions.

An option we now have is to restage these templates as “analysis” datasets.

There are a number of names in the inflight templates that have been adjusted (eg XXX_NSCN).

hen there is a sample that doesn't have a valid parent (e.g. a blank in profile_name) or some other problem associated with the locator hierarchy. When a locator does not have a child (e.g. when there is a site_name on the site sheet that does not appear on the profile sheet), that generates an informational message only but does not put the locator on HOLD. You'll see these problems in the CheckLocatorSummary report.

Data goes on HOLD when there is a problem with the one or more of the variables in the variable group. There are also a few cases where one variable group failing will impact another variable group; an example of that is failing to have all of the necessary dataset metadata such as missing the curator_email. You'll see these problems in the CheckTemplateErrors report associated with each template.

There's nothing wrong with the 01 CL locator. There is a problem with the locator "345"in the Stangenberger set. You will see that flagged (but with no reason) on the CheckLocatorSummary report. The name collides with one of the (new) names in the NRCS 2014 dataset. (And yes, I should be tagging that better).

Here's what happens with 01 CL and grp_location.

There are 4 variables in the group - lat (6.5), long (-77), elevation (130), and datum (NAD27).

- Lat, long, and elevation are all REAL variables. As such, they are first checked to make sure that they are actually numbers (e.g. not ">130" or "N/A") and within the current min/max range (if any).

- datum has a CV. As such, it is checked against that CV. The string "NAD27" passes; a string of "NAD 27" would not pass.
- Lat and long also have special checking. That checking currently enforces a minimum precision of 4 (not 5) digits. We can certainly relax that to an informational message or get rid of the check altogether.

When any of the variable within a group fail a check, the entire group is tagged as HOLD. Because groups either make it or fail together.

You'll see the same sort of "group" failure happening with sample 8002034.2 due to the bd_other of 0.78 which is less than the min value of 0.85. Fix that and the group will be fine. The current min/max values are attached.