Manage User Profile – must have Default NASIS Group set to ‘Soil Metadata Repo Admin Group’ to enter/edit NREPO data.

How to load sytem in to nasis –

queries - soil metadata repository – NREPO – Repo Sync – from system or NREPO – Repo Sync – from system were sysver like ? -- run against national database – wildecards

Local database – accept

Close local database tab

Table – nasissite – hightlight Soil Metadata Repository

Table Editor menu – select ‘load related for selected rows’, select system as target table. Other possible tables to load – domain group and unit of measure

Check out

Check in

Upload all changes to nasis

Clear local database

Copy data model – change sysvernm

NREPO Visio diagram

Domains

Unit of Measure

# **Introduction**

The purpose of this document is to help people understand how to perform the data model creation and maintenance activities. This system is necessary in part because the NASIS application is metadata driven. The NASIS application requires a lot more metadata than is needed for a database in general. Ultimately we ended up using this system for all of our data models. The NASIS project encompasses the following databases:

1. NASIS Server

2. NASIS Client

3. Staging Server

4. Soil Data Warehouse (SDW)

5. Soil Data Mart (SDM)

6. Soil Data Mart Data Management (SDMMGT)

7. Business Application Framework (BAF)

8. Business Application Framework Log (BAFLOG)

9. Business Application Framework SDA Cache (BAFSDACACHE)

10. State Soil Survey (SSURGO)

This document describes the following two major entities:

1. The NASIS Repository Data Model

2. The NASIS 6 Style Metadata Standard

Even though we use the NASIS Repository for all of the above data models, only the following data models require up-to-date NASIS 6 Style Metadata because it is used by various applications: NASIS – both server and client, staging server, soil data warehouse, soil data mart and soil data mart data management.

# **The NASIS Repository Data Model**

The NASIS repository (NREPO) is a set of tables that are used to enter the details of a specific data model. Below is a database diagram of the NASIS Repository tables.



## **Current Database/Data Model Conventions**

### **General Conventions**

In general, each data model should be third normal form, although we don’t go through any formal process to determine this. Exceptions need to be justified.

Table and column names should provide a good indication of what that table or column records. Names should be well thought out.

Within a data model, column names must be logically unique. In other words, there should not be two columns in different tables with the same name where any other attribute of that column varies (data type, field size, precision, minimum, maximum, domain, description, etc).

A column shouldn’t be too generically defined, .e.g. "SequenceNumber". A column name that requires one definition in one context and a slightly different definition in another context should be separated into two different columns.

Barring a legitimate reason to the contrary, all referential integrity should be enforced at that database engine level.

## **Nomenclature Conventions**

### **General Conventions**

In general, the only characters allowed in a name are letters, upper and lower case, digits, spaces and underscores. A name should begin with a letter. Spaces are only permitted in the name of something that doesn’t correspond to a database object, e.g. table and column labels. Underscores are only permitted in logical names, domain names, constraint names and index names.

To make names more readable, our current convention is to use a variant of camel casing. The first letter of a word or truncated word should be capitalized and the remainder should be in lower case. In general, acronyms and abbreviations should be in all upper case. When name length is an issue, certain words can be truncated, abbreviated or even represented by a single capital letter.

Exceptions to these conventions will be noted where appropriate.

For the NASIS database, we originally defined both a physical name and a logical name for each table and column. The NASIS database was originally implemented in Informix, and at that time Informix allowed a maximum of 18 characters in a physical name, and many of our physical names were pretty cryptic. In our newer SQL Server databases we no longer make a distinction between logical and physical names, but this convention will likely be retained for NASIS, even when it is implemented in SQL Server because in queries, or scripts based on the NASIS CVIR language, either logical or physical names can be used, and changing either of these names would break a lot of existing queries, calculations, validations and reports.

### **Databases/Data Models**

A database/data model name may contain a maximum of 30 characters. A database/data model name may not contain any upper case letters, spaces or underscores. A database name may not be a SQL Server reserved keyword.

# **System**

The system table is a parent record for all of the objects that are associated with a data model, attributes, tables, columns, etc. In general, a record in this table corresponds to a particular version of a particular database.

## **Column List:**

**System Name** – The descriptive name of a database.

**System Version** – The version number of a database. In the NASIS Repository, there is typically more than one instance of almost every data model. No two instances of a particular data model should ever share the same version number. When does a new version of data model need to be created without creating a new instance of that data model, and when does a new instance of a data model need to be created? A general guideline is that we don’t typically update the version of an existing data model if a database that corresponds to that version exists in production. In this case, in general, we create a new instance of that data model, and then update its corresponding version number accordingly. We want to preserve the documentation of a data model version for which a corresponding production database exists.

#### **WARNING:** Releasing a new NASIS data model with a different version forces all NASIS user to recreate their local NASIS databases. Updating the NASIS data model version should never be done without consent of our sponsors in the Soil Science Division. When the NASIS data model version needs to be updated, the sponsors need to have time to contact all NASIS users and let them know that they need to upload all pending changes to the central NASIS database then check-in everything.

**Description** - A narrative text description of a database or information system.

**Creation Date** – The date the system record was created, format MM/DD/YYYY.

**Notes** – Notes describing decisions, issues, or other history related to the record.

**Sqladvanceoptions** – Use this column to capture any SQL Server advance database options. Currently this is used to document any special partitioning that is used at the database level.

### **Creating a new instance of an existing data model.** To create a new instance of an existing data model, you need to use the ‘copy selected trees’ option in NASIS to copy the existing data, and ‘paste rows/trees (inserting new rows)’ option in NASIS to create a new instance of the data model. You also must rename the system version, since the system name and system version combination must be unique.

# **Tablecollection**

The tablecollection table is used to group a set of tables. In NASIS this is used to group tables by business objects.

**Table Collection Name** – The unique name for a table collection (business object). It is often the same as the name of the table collection's root table. – For non NASIS data models, it is acceptable to use ‘All Tables’.

**Table Collection Sequence (NASIS)** -The sequence in which table collections are displayed.

**Table Collection Insert Sequence (NASIS)** - The sequence in which the corresponding table collection should be inserted into the database, relative to other table collections. The insert sequence is used when doing a batch of uploads to the server database, to make sure that foreign keys can get resolved. If it's wrong uploads could fail.

**Restricted NASIS Site ID (NASIS)** – The ID of the NASIS site to which creation of new instances of the corresponding table collection is restricted. A NASIS site is an administrative entity by which table collection instance ownership is managed. A user's default site has to be set to the restricted site in order to create a new record. This makes sure that the new record is owned by the restricted site, which then insures that other users can't edit it.

**Visible When Restricted (NASIS)** - A Boolean value that indicates whether or not the table in in the corresponding table collection can be edited via the general NASIS grid editor. Some table collections have their own specialized edit interface.

Restricted NASIS Site ID and Visible when restricted are client side settings that determine what NASIS Site members can edit the data and if the table should be visible in NASIS for users who do not have permission to edit the data.

In NASIS 7.3, some example Restricted NASIS Site ID data:

|  |  |  |
| --- | --- | --- |
| **Table Collection Name** | **Restricted NASIS Site ID** | **Visible When Restricted** |
| plant | 2 = Flora | true |
| Local plant | 6 = Local Plant | true |
| Domain group | 3 = Soil Metadata Repository | false |
| Project type | 1 = NCSS Pangaea | false |
| System | 3 = Soil Metadata Repository | false |

**Load All (NASIS)** - A Boolean value that indicates whether or not all data for the root table in the corresponding table collection should be automatically loaded whenever that table is first viewed.

Currently only the NASIS Site table has load all set to true.

**Visible in Grid Editor? (NASIS)** A Boolean value that indicates whether or not the table in in the corresponding table collection can be edited via the general NASIS grid editor. Some table collections have their own specialized edit interface. Currently the only tables that have the visible in grid editor set to false are Pedon DB Metadata, Choice List Set, and Edit Setup. In the past the Form table collection had this set to false, and Forms were not visible in the NASIS grid editor.

**Table Collection replication Type (NASIS)** - Indicates whether or not instances of the corresponding NASIS object can be selected for inclusion in a NASIS user's replicate or if the corresponding object should be automatically replicated to all NASIS client databases.

There are two replication types, auto replication and selectable for replication. When the table collection replication type is set to auto replicated, you can also enter the Customization Query and Customization Column. When the table collection is set to selectable for replication, both Customization Query and Customization Column should be null.

**Customization Query (NASIS)** - A query that specifies which columns should be displayed, and the default sort order of records in a customizable table collection choice list. Such a query may include columns from related table collections. Column lookups for child tables use the customized query to display columns from the parent record.

**Customization Column (NASIS)** - The column id of the column whose values should be stored to indicate which records should be displayed for a customized table collection. If you enter a customization query, you must also enter a customization column. Enter the column id value found in the tablecolumn table. Examples of customization column data:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Table Collection Name | Customization Column | Table Physical Name | Attribute Physical Name | Column ID | Customization Query |
| Area Type | 3188 | area | areaiid | 3188 | SELECT area.obterm [Obsolete?], nasissite.nasissitename [Area Type NASIS Site Name], areatype.areatypename [Area Type Name], area.areasymbol [Area Symbol], area.areaname [Area Name], area.areaiid [Primary Key] FROM area\_View\_NONDEL\_0 as area INNER JOIN areatype\_View\_NONDEL\_0 as areatype ON area.areatypeiidref = areatype.areatypeiid INNER JOIN nasissite\_View\_NONDEL\_0 as nasissite ON areatype.atdbiidref = nasissite.nasissiteiid ORDER BY nasissite.nasissitename, areatype.areatypename, area.areasymbol |
| Ecological Site | 15663 | ecologicalsite | ecositeiid | 15663 | SELECT ecologicalsite.obterm [Obsolete?], ecologicalsite.ecositeid [Ecological Site ID], ecologicalsite.ecositenm [Ecological Site Name], ecologicalsite.ecositeiid [Primary Key] FROM ecologicalsite\_View\_NONDEL\_0 as ecologicalsite ORDER BY ecologicalsite.ecositeid |
| Geomorphic Feature Type | 12756 | geomorfeat | geomfiid | 12756 | SELECT geomorfeat.obterm [Obsolete?], geomorfeattype.geomftname [Feature Type Name], geomorfeat.geomfname [Feature Name (singular)], geomorfeat.geomfiid [Primary Key] FROM geomorfeat\_View\_NONDEL\_0 as geomorfeat INNER JOIN geomorfeattype\_View\_NONDEL\_0 as geomorfeattype ON geomorfeat.geomftiidref = geomorfeattype.geomftiid ORDER BY geomorfeattype.geomftname, geomorfeat.geomfname |
| Local Plant | 12736 | localplant | lplantiid | 12736 | SELECT localplant.obterm [Obsolete?], localplant.lplantname [Common Name], plant.plantsym [Plant Symbol], plant.plantsciname [Scientific Name], plant.plantnatvernm [National Vernacular Name], localplant.lplantiid [Primary Key] FROM localplant\_View\_NONDEL\_0 as localplant INNER JOIN plant\_View\_NONDEL\_0 as plant ON localplant.plantiidref = plant.plantiid ORDER BY localplant.lplantname, plant.plantsym |

When using auto replication, all tables in the table collection are automatically downloaded to the local NASIS database and the above two customization columns can be used by the NASIS client to determine what data is display in drop down lists. The Customization Query is often used to return the obsolete column and the sort order of the data..-

Currently only the following table collections are set to Auto Replicate in NASIS: Area Type, Calculation, Choice List Set, Ecological Site, Edit Setup, Evaluation, Form, Geomorph Feat Type, Local Plant, Milestone Type, NASIS Site, NASIS User, Other Veg Class Type, Pedon DB Metadata, Plant, Project Concern Type, Project Data Type, Project Type, Property, Query, Report, Rule, Soil Series, Tech Soil Serv Type, Tech SS Prog Benefit, Unit of Measure, USFS Eco Class Type, USFS Interp Category, USFS Interp Restriction, USGS Quad Type, WS Import Map. Some of these also have a Customization Query and Customization Column.

When using selectable for replication, the NASIS user is able to select what data should be downloaded to their local NASIS database.

**Table Collection ID (Required)** – An integer value that uniquely identifies a table collection in the NASIS 6 Style Metadata. The table collection ID must be unique within a specific data model. Using the table collection ID allows the same ID to be used in the NASIS 6 Style Metadata for all versions of a data model. For example, the ‘system’ table collection ID of 264 has been used in all releases of NASIS, NASIS 5.1 to the current NASIS data model. Must be unique within each database.

**NOTE:** There is a table collection history table as well, if you want to document changes in more detail.

**System\_Table**

The system\_table table is used to document all of the properties for each table in a database.

**Table Logical Name** - The logical name for each database table. Required, must be unique within each database.

**Table Physical Name** - The physical (SQL) name of each database table. Required, must be unique within each database. A table name may contain a maximum of 30 characters. A table name may not contain any spaces or underscores. A table name may not be a SQL Server reserved keyword.

A table name should be as connotative as possible. Don’t choose a shorter name over a longer name at the expense of clarity. A table name should accurately reflect what a table records. In general, table names should be singular rather than plural, e.g. "mapunit" rather than "mapunits".

**Table Label** - A descriptive business oriented name for a database table. A table label may contain a maximum of 80 characters. A table label may contain any valid printable character. Required, must be unique within each database.

**Table Help Text** A narrative description of each table. The help text should make the purpose of this table clear to all potential audiences, which may include both end-users and developers.

**Soil entity type (NASIS) -** A way to group the various tables in a database, this information is only used by some of the NREPO reports.

|  |
| --- |
| **Valid options:** |
| Aggregated Data |
| Data Dictionary |
| Dynamic Domain |
| Eco Site Inventory |
| Interp Generation Related |
| Pedon Data |
| Pedon Lab Data |
| Project and TSS |
| Replication Related |
| Series Data |
| Site Data |
| SSURGO Export Related |
| System Related |

**Import Export File Name -** The base part of the file name of a table's associated ASCII pipe delimited import/export file. The complete name of a table's associated import/export file is the base name followed by the characters ".txt". For example, if the base name is "area", the name of the associated import/export file is "area.txt". Usually the import export file name is the same as the table physical name. Required, must be unique with a database. This is used for SSURGO exports.

**Visible (NASIS)** – There is a checkbox that indicates whether or not the corresponding table should be visible in the NASIS grid editor. A table that is not visible in the NASIS grid editor should not be able to be selected as the target table for a NASIS query. Required, default = checked.

**Selectable(NASIS)** – There is a checkbox that indicates whether or not records from the corresponding table can potentially be created, read, updated or deleted in NASIS. A table can be selectable but not visible in the NASIS grid editor. Such a table is typically managed by a special editor in NASIS. A table that is not selectable cannot be referenced in a NASIS query. When a table is set to selectable = unchecked, both visible and editable should be unchecked as well. Required, default = checked.

**Editable(NASIS)** – There is a checkbox that indicates whether or not the contents of the corresponding table can be changed by a user through either the NASIS grid editor or one of the special NASIS editors. Required, default = checked.

For NASIS 7.3, the following tables were set to visible = false or selectable = false or editable = false:

|  |  |  |  |
| --- | --- | --- | --- |
| **Table physical name** | **Visible** | **Selectable** | **Editable** |
| contextsuffix | false | false | false |
| customchoicedomain | false | true | false |
| customchoicelistset | false | true | true |
| customchoicelookup | false | true | false |
| distcompmd | true | true | false |
| distinterpmd | true | true | false |
| distlegendmd | true | true | false |
| distmapunitmd | true | true | false |
| disttextmd | true | true | false |
| editsetup | false | true | true |
| editsetupelement | false | true | false |
| editsetuptable | false | true | false |
| formfavorites | false | true | false |
| keyrangeclient | false | false | false |
| keyrangeserver | false | false | false |
| nasismessage | false | false | false |
| nasismessageuser | false | false | false |
| nasissession | false | false | false |
| pedondbcolumnmetadata | false | true | false |
| pedondbtablemetadata | false | true | false |
| phvnirscanrawdata | false | true | false |
| queryfavorites | false | true | false |
| replicationparameter | false | false | false |
| reportcolumn | false | true | false |
| reportfavorites | false | true | false |
| ruleevalcomp | false | true | false |
| rulerulecomp | false | true | false |
| siteiristubeimage | false | true | false |
| soilseries | true | true | false |
| soilseriesmlrasusing | true | true | false |
| soilseriesstatesusing | true | true | false |
| soilseriestaxfamother | true | true | false |
| soilseriestaxmineralogy | true | true | false |
| soilseriesURL | false | false | false |

**No Insert (NASIS)** - There is a checkbox that indicates whether or not end users should be able to insert records into the corresponding table. As of NASIS 6.0, this was implemented as "no insert" only, i.e. it does not prevent deletes. Required, default = unchecked.

In NASIS 7.3, following tables where set to no insert = true:

|  |
| --- |
| **Table physical name** |
| distcompmd |
| distinterpmd |
| distlegendmd |
| distmapunitmd |
| disttextmd |
| ncsslayerlabdata |
| ncsspedonlabdata |
| usfseclevel |
| usfsecoclass |
| usfsectype |
| usfsicat |
| usfsinterp |
| usfsirclass |
| usfsirestrict |

**Root Table (NASIS)** – There is a checkbox that indicates if the corresponding table is the root table of an object hierarchy. Each NASIS table collection must have one and only one root table. Root tables in the NASIS database require additional supporting tables, see “XXXX” for more details. Required, default = unchecked.

In NASIS 7.3, the following tables were set to root table = true:

|  |  |
| --- | --- |
| **Table collection name** | **Table physical name** |
| Area Type | areatype |
| Calculation | calculation |
| Choice List Set | customchoicelistset |
| Data Mapunit | datamapunit |
| Distribution | distmd |
| Domain Group | domaingroup |
| Ecological Site | ecologicalsite |
| Edit Setup | editsetup |
| Evaluation | evaluation |
| Form | form |
| Geomorph Feat Type | geomorfeattype |
| Legend | legend |
| Local Plant | localplant |
| Mapunit | mapunit |
| Milestone Type | milestonetype |
| NASIS Site | nasissite |
| NASIS User | nasisuser |
| NCSS Lab Data | ncsspedonlabdata |
| Other Veg Class Type | othvegclasstype |
| Pedon | pedon |
| Pedon DB Metadata | pedondbtablemetadata |
| Plant | plant |
| Project | project |
| Project Concern Type | projectconcerntype |
| Project Data Type | projectdatatype |
| Project Type | projecttype |
| Property | property |
| Query | query |
| Report | report |
| Rule | rule |
| Site | site |
| Site Association | siteassoc |
| Soil Series | soilseries |
| System | system |
| Tech Soil Service | techsoilservice |
| Tech Soil Serv Type | techsoilservicetype |
| Tech SS Prog Benefit | techsoilservprogbenefit |
| Transect | transect |
| Unit of Measure | uom |
| USFS Eco Class Type | usfsectype |
| USFS Interp Category | usfsicat |
| USFS Interp Restriction | usfsirestrict |
| USGS Quad Type | usgsquadrangletype |
| Vegetation Plot | vegplot |
| WS Import Map | wsimportmap |

### Depending on if a table is a root table or not a root table, certain columns are typically created, see “Which Columns should typically be Included in a NASIS Data Object’s Root Table? (NASIS Specific)” or “Which Columns should typically be Included in a NASIS Table that is not a NASIS Data Object’s Root Table? (NASIS Specific)“ for more details.

**Temporary Table (NASIS)** – For a table, a Boolean value that indicates that the table is created as a temporary table. . The “NREPO -SQL Script – CreateTables” report ignores any temporary tables. Required, default = false.

In NASIS 7.3, there was only one table set to table temporary = true, the icomponent table.

**Aliased**– There is a checkbox that indicates that the table is created as an alias. . The “NREPO - SQL Script – CreateTables” report ignores any aliased tables. Currently, the only database that has any aliases defined is the Staging Server. The “NREPO - SQL Script - CreateStagingAliases - create synonyms” report is used to create the Staging Server synonyms using the tables that are set as aliased. The list of aliased tables in the Staging Server should match the list of Staging Alias Required (NASIS) tables in NASIS. Required, default = unchecked.

In Soil Staging 7.3, the following tables had Aliased = true:

|  |
| --- |
| **Table physical name** |
| evaluation |
| nasisgroup |
| nasisgroupmember |
| nasissession |
| nasissite |
| nasissiteadmin |
| nasisuser |
| property |
| rule |
| ruleevalcomp |
| ruleratingclass |
| rulerulecomp |

**Static Table (NASIS)** - There is a checkbox that the table's contents are static. . Required, default = unchecked.

In NASIS 7.3, the following tables are set as static tables:

|  |
| --- |
| **Table physical name** |
| contextsuffix |
| pedondbcolumnmetadata |
| pedondbtablemetadata |

**Client Only (NASIS)** - There is a checkbox that, when set, indicates that the corresponding table is present only in a NASIS client database. . Required, default = unchecked.

In NASIS 7.3, there were only two tables that are set as client only, keyrangeclient and replicationparameter.

**Server Only (NASIS)** - There is a checkbox that, when set, indicates that the corresponding table is present only in the central NASIS server database. . Required, default = unchecked.

In NASIS 7.3, the following tables are set as server only tables:

|  |
| --- |
| **Table physical name** |
| contextsuffix |
| keyrangeserver |
| nasismessage |
| nasismessageuser |
| soilseriesURL |

**Create AS View (Currently not used)** – There is a checkbox that indicates whether or not the corresponding "table" is really a view. NASIS code makes no distinction between tables and views, so this attribute is only relevant to the database creation script. The “NREPO - SQL Script - Create Tables” report ignores any views. Required, default = unchecked.

**Staging Alias Required (NASIS)** – There is a checkbox that indicates that the corresponding table in the staging database requires an alias. This is used to document NASIS tables that are used directly in the Staging Server database. This is done by creating synonyms in the Staging Server database using NASIS as the source of the data. Required, default = unchecked.

In NASIS 7.3, the following tables are set as staging alias tables:

|  |
| --- |
| **Table physical name** |
| evaluation |
| nasisgroup |
| nasisgroupmember |
| nasissession |
| nasissite |
| nasissiteadmin |
| nasisuser |
| property |
| rule |
| ruleevalcomp |
| ruleratingclass |
| rulerulecomp |

**Staging Counterpart Required (NASIS)** – There is a checkbox that indicates that a table of the same name and similar structure exists in the Staging Server database. This is used to document tables in NASIS that are also created in the Staging Server database. Required, default = unchecked.

**Pedon PC to NASIS** - There is a checkbox that indicates that the table is included in the import from Pedon PC to NASIS. Required, default = unchecked.

.

**Note:** Any changes to the list of Pedon PC to NASIS tables also requires changes to the Pedon PC application.

In NASIS 7.3, the following tables are set as Pedon PC to NASIS tables:

|  |
| --- |
| **Table physical name** |
| pediagfeatures |
| pedon |
| pefmp |
| pehydricfieldindicator |
| peinfiltrationch |
| peinfiltrationchdata |
| peinfiltrationfh |
| peinfiltrationfhdata |
| peinfiltrationsummary |
| pepenetrationresistance |
| perestrictions |
| pesoilstability |
| petaxhistfmmin |
| petaxhistmoistcl |
| petaxhistory |
| petext |
| petxhistfmother |
| phcemagent |
| phcolor |
| phconccolor |
| phconcs |
| phcracks |
| phdb |
| phdbcompliantcavity |
| phdbcore |
| phdbcorereading |
| phdbscoop |
| phdbscoopreading |
| phdesgnsuffix |
| phfeatcolor |
| phfeatures |
| phfmp |
| phfrags |
| phhuarts |
| phlabresults |
| phmottles |
| phorizon |
| phpores |
| phpvsf |
| phpvsfcolor |
| phrdxfeatures |
| phredoxfcolor |
| phroots |
| phsample |
| phstructure |
| phtext |
| phtexture |
| phtexturemod |
| plotplantinventory |
| site |
| siteaoverlap |
| siteassoc |
| siteassocsite |
| siteassocsoi |
| siteassoctext |
| sitebedrock |
| siteecositehistory |
| siteerosionacc |
| sitegeomordesc |
| sitemuoverlap |
| siteobs |
| siteobstext |
| sitepm |
| sitesoilmoist |
| sitesoiltemp |
| sitesurffrags |
| sitetext |
| transect |
| transectestcomposition |
| transecttext |
| vegplot |

**Import Pedon Diagnostic Message SQL Statement** - A SQL statement used in the construction of an import Pedon diagnostic message for the corresponding table in a Pedon database. The statement returns the User ID and Rec ID of the Site, Transect or Site Association associated with the corresponding table in a Pedon database.

Some examples of the Import Pedon Diagnostic Message SQL Statements from NASIS 7.3.

|  |  |
| --- | --- |
| **Table physical name** | **Import Pedon Diagnotic Message SQL Statement** |
| pedon | SELECT pedon.upedonid, pedon.peiid FROM pedon WHERE pedon.peiid=[Rec ID]; |
| pefmp | SELECT pedon.upedonid, pedon.peiid FROM pedon INNER JOIN pefmp ON pedon.peiid = pefmp.peiidref WHERE pefmp.pefmpiid=[Rec ID]; |
| site | SELECT site.usiteid, site.siteiid FROM site WHERE site.siteiid=[Rec ID]; |
| siteaoverlap | SELECT site.usiteid, site.siteiid FROM site INNER JOIN siteaoverlap ON site.siteiid = siteaoverlap.siteiidref WHERE siteaoverlap.sareaoviid=[Rec ID]; |
| siteassoc | SELECT siteassoc.usiteassocid, siteassoc.siteassociid FROM siteassoc WHERE siteassoc.siteassociid=[Rec ID]; |
| transect | SELECT transect.utransectid, transect.tsectiid FROM transect WHERE transect.tsectiid=[Rec ID]; |
| transectestcomposition | SELECT transect.utransectid, transect.tsectiid FROM transect INNER JOIN transectestcomposition ON transect.tsectiid = transectestcomposition.tsectiidref WHERE transectestcomposition.tsectestcompiid=[Rec ID]; |
| vegplot | SELECT site.usiteid, site.siteiid FROM (site INNER JOIN siteobs ON site.siteiid = siteobs.siteiidref) INNER JOIN vegplot ON siteobs.siteobsiid = vegplot.siteobsiidref WHERE vegplot.vegplotiid=[Rec ID]; |

**DAG Level** - A table's level in the in the set of directed acyclic graphs that are defined for the corresponding data model. A root table corresponds to level 0. The DAG level is calculated using the relationship master data, so all indexes and relationship data must be entered before you can calculate the DAG level. The DAG calculation can be found under the table ‘System Table’. Use the following process to generate the proper DAG level data:

1. Select Calculations
2. Select System Table
3. Run calculation – DAG Level – 1. Delete Current DAG Levels
4. Run calculation – DAG Level – 2. Assign New DAG - The first time, run this several times in a row, at least 5 times.
5. Select Reports under Soil Metadata Repository
6. Run both reports: ‘NREPO – DAG Level Missing’ and ‘NREPO – DAG Level Wrong’. Continue to run step 4 till these reports show no records.

You must calculate the DAG levels for all databases, the DAG level is what is used to determine the order for deleting the tables in the “NREPO -SQL Script – DropTables” report script.

**Table ID** - An integer value that uniquely identifies a table in the NASIS 6 Style Metadata. The table ID must be unique within a specific data model. Using the table ID allows the same ID to be used in the NASIS 6 Style Metadata for all versions of a data model. For example, the ‘area’ table ID of 256 has been used in all releases of NASIS, NASIS 5.1 to the current NASIS data model. Must be unique within each database.

The Table IDs can be manually entered, if you want to control what table IDs are used or there is a calculation under ‘System Table’ to populate null values. The calculation is called ‘Update Table ID where ID is null’. Running the calculation on the selected set, instead of highlighting specific rows, makes sure that all null Table IDs get populated.

**Table Description** – The descriptive name of a table.

**Sql Advance Options** - Use this column to capture any SQL Server advance table options. Currently this is used to document special locking details that is used for WSS in the baf and bafsdacache databases.

#### **Examples:**

ALTER TABLE [dbo].[AoiSoilMapUnitPolygon] SET ( LOCK\_ESCALATION = AUTO )

**NOTE:** There is a system table history table as well, if you want to document changes in more detail.

**Domain Group**

### **Domains**

For our project, a domain is defined as a finite set of acceptable character strings. Although we do have domains where all choices represent either a valid integer or floating point value, we haven’t ever implemented a truly numeric domain.

### **Domain Integrity**

How we implement domain integrity varies from application to application. Here are some of the approaches we have used.

1. All domains are stored in a common table that an application uses to constrain choices during data entry. This is how the NASIS application implements domain integrity. The application is enforcing domain integrity, but a highly generalized approach is used.

2. A domain is implemented as a related "lookup" table, and referential integrity enforces domain integrity. There is a lot of this in the Soil Data Mart and Soil Data Mart Management databases. The proliferation of lookup tables can be annoying.

### **Domain Documentation in the Data Model Repository**

A common structure is used to document all domains in our data model repository. This scheme was originally developed for NASIS. Not all attributes that are defined may be necessary in all cases, but most domains can be documented using this scheme.

For a domain as a whole, we record the following:

1. Domain ID. Domain ID is an integer value that unambiguously identifies a domain.

2. Domain name. A domain name should provide a good indication of what a domain represents. For a domain that is associated with only one logical attribute, the domain name is often similar or exactly the same as the corresponding attribute logical name.

3. Ordering. This attribute indicates if a domain is explicitly ordered or sorted ascending on Shorter String (see immediately below). These are the only two options currently supported in our data model repository.

4. Ordered? A checkbox for indicating if the members of a domain can be logically ordered. When a domain can be logically ordered, that order must be specified using "Member sequence" (see immediately below), and the lowest value must always correspond to sequence number 1. In NASIS, when this value is set, the less than and greater than operators can be used in a query that includes the corresponding attribute. For Soil Data Viewer, when this value is set, a user may be allowed to change the default tie-break rule at runtime, when aggregation is performed.

5. Display label? When Shorter String (see immediately below) is used for data entry, and displaying Shorter String alone in a choice list is not sufficient for the user to be able to make an informed selection, setting this value indicates that both Shorter String and Longer String should be displayed in any choice list for the corresponding domain. For some domains, the values in Shorter String are relatively short cryptic codes that require further explanation.

6. Customizable? - A checkbox for indicating whether or not a particular domain is customizable by the NASIS user.

For each member of a domain, we record the following:

1. Member sequence. At the current time, domain members must be either explicitly sequenced, or sort ascending on "shorter string". Members are sequenced, beginning at one.

2. Member ID. Member ID is an integer value that unambiguously identifies a member of a domain. These values should be assigned sequentially, beginning with one. In NASIS, this is the value that is actually stored in the database in order to record a domain selection.

3. Shorter String. Shorter string is an alphanumeric string containing 128 or fewer characters. For NASIS, this is the string that must be used for data entry. Since everything in NASIS is case sensitive, the string is usually in all lower case. Within a domain, this value must be unique.

4. Longer String. Longer string is an alphanumeric string containing 254 or fewer characters. This tends to be a longer, more connotative, mixed case string. In a report, a domain selection is typically represented by this value. Within a domain, this value must be unique.

5. Description. When what a domain choice corresponds to cannot be inferred from either Shorter String or Longer String, a corresponding narrative text description is required. You can always choose to provide a description, for any reason. It is OK to provide a description for some members but not all members.

If an application has no need to have both a shorter and longer version of an alphanumeric string, the same value can be recorded in both fields.

Should the occasion arise where the value used for data entry is not the shorter string, the data model repository will have to be updated to support this option.

**Unit of Measure**

# **Attribute**

The NASIS Repository makes a distinction between an “attribute” and a “column”. A column must be associated with one and only one attribute, but an attribute may be associated with one or more columns, in one or more tables. Attributes and tables are not directly related:

Attribute -> Column -> Table

Soil data includes both point data and what we refer to as “aggregated” data. Point data is data at a particular point on planet Earth. Pedon data is point data. Data about map units is considered to be “aggregated” data in that it is based in part on data from multiple points. Therefore, the data for many attributes is reported either as a range of values, or a range of values with a corresponding “expected value”. We refer to such an attribute as a “modal attribute”. Other terms include “high, low” or “high, low, representative value”, also known as “HL” and “HLRV”.

The same attribute is sometimes related to many columns. This allows us to record attributes at the attribute level that we don’t want to have to define for every column instance of that attribute.

In our data model repository, for a column associated with a HL or HLRV attribute, we record a single column, in order to cut down on the amount of duplicate data entry that would otherwise be required.

Any process that creates the scripts required to create a database instance of a data model programmatically expands columns associated with a modal attribute into multiple columns. The same expansion occurs whenever reports or metadata are created.

**Attribute Logical Name** – The attribute logical name may contain a maximum of 30 characters. It may not contain any spaces. Required, must be unique.

**Attribute Physical Name** – The attribute physical name may contain a maximum of 30 characters. It may not contain any spaces or underscores. It may not be a SQL Server reserved keyword. Required, must be unique.

A column name should be as connotative as possible. Don’t choose a shorter name over a longer name is the expense of clarity. A column name should accurately reflect what a column records

For a NASIS database only (NASIS the application as opposed to NASIS the project/staff), a column name may contain a maximum of 33 characters, but only when the corresponding column is modal (low/high or low/rv/high) and/or calculated. In NASIS, an extension is added to the name of a modal column, and for a calculated column, a companion column with its own extension, ‘\_s’ is created to indicate the source of the corresponding value. For example, in NASIS, attribute "sandtotal" is a low/rv/high column that may also be automatically calculated. Therefore this attribute is represented by a total of six columns:

| **Column Name** | **Purpose** |
| --- | --- |
| sandtotal\_l | Records the low end of the range of total sand. |
| sandtotal\_ls | Indicates if the low end of the range of total sand was calculated or manually entered. |
| sandtotal\_r | Records the representative value of total sand. |
| sandtotal\_rs | Indicates if the representative value of total sand was calculated or manually entered. |
| sandtotal\_h | Records the high end of the range of total sand. |
| sandtotal\_hs | Indicates if the high end of the range of total sand was calculated or manually entered. |

**Attribute Label** - This column is being dropped in the NASIS 7.4 data model.

## **Attribute Logical Data Type -** Specify the logical data type, current options are:

|  |
| --- |
| Binary |
| Boolean |
| Calculation |
| Choice |
| Date/Time |
| Edit Setup |
| SQL Geometry |
| ESRI Geometry Line – no longer being used |
| ESRI Geometry Point – no longer being used |
| ESRI Geometry Polygon – no longer being used |
| Evaluation |
| File Reference |
| Float |
| GUID |
| Integer |
| Money |
| Narrative Text |
| Property |
| Query |
| Report |
| Rule |
| String |
| Hyperlink |
| Unknown |
| XML |

#### **Process for adding/changing logical data types.**

Just adding a new logical data type to the domaindetail table or making changes to the domaindetail data is not sufficient to actually see these changes in NASIS because NASIS is metadata driven and the list logical data types are coming from the MetadataDomainDetail table, not the domaindetail table. There are two options for showing changes to the logical data types in NASIS: 1)Release a new version of NASIS which includes the domaindetail changes or 2)Make the changes to the logical data type in the MetadataDomainDetail table on the NASIS server database, wait a day, then reinitialize your local NASIS client database. There is a nightly BCP process at NITC that downloads all of the metadata tables from the NASIS server database, which is used when a NASIS client data is reinitialized. The DomainID value for logical data type is 737.

**Attribute Field Size** - For attributes that have attribute logical data type = string, you can specify the length of the string. You would not specify a field size for columns that have a physical data type of varchar(max).

**Attribute Precision** – For attributes that have attribute logical data type = float, you can set the decimal precision, and for attributes that have column physical data type = datetime2, you can set the date precision.

**Unit of Measure** – Where appropriate, the unit of measured is determined by selecting the appropriate unit of measure from a drop down menu.

**Domain** – Where appropriate, the Domain is determined by selecting the appropriate domain from a drop down menu. The logical data type must be set to “choice” in order to select a domain.

**Attribute Shorter Date Time Interval Qualifier (NASIS)** For attributes that have attribute logical data type = date/time, the attribute shorter date time interval qualifier is used to derive Date Time Precision in the NASIS 6 Style Metadata.

Current options:

|  |
| --- |
| Year |
| Month |
| Day |
| Hour |
| Minute |
| Second |
| Fraction |

**Attribute Longer Date Time Interval Qualifier (NASIS)–**The attribute longer date time interval qualifier is no longer used.

**Attribute Minimum** - The minimum value allowed. This value is required only where appropriate. A value must obviously be within the range of the corresponding data type. A floating point value should not be expressed where precision exceeds the corresponding precision value. It is permissible to provide a minimum without a corresponding maximum, and vice versa.

**Attribute Maximum** – The maximum value allowed. This value is required only where appropriate. A value must obviously be within the range of the corresponding data type. A floating point value should not be expressed where precision exceeds the corresponding precision value. It is permissible to provide a minimum without a corresponding maximum, and vice versa.

## **Attribute Case Sensitive –** This indicates that the data in the column must be searched in a case sensitive manner. Currently the only attributes that are case sensitive are the mapunit symbol and horizon name. When generating the SQL for the database, a collation of ‘Latin1\_General\_BIN’ is used to indicate a case sensitive column.

## **Attribute Official Definition -** A narrative description of each attribute. The official definition should make the purpose of this attribute clear to all potential audiences, which may include both end-users and developers. This data is stored in the NASIS 6 Style Metadata as the column description.

## **Attribute Extended Definition -** An expanded narrative description of each attribute. The extended definition should expand on the purpose and use this attribute. This data is stored in the NASIS 6 Style Metadata as the column help text

**Attribute Id** - An integer value that uniquely identifies an attribute. The attribute ID must be unique within a specific data model. Using the attribute ID allows the same ID to be used for all versions of a data model. For example, the ‘area name’ attribute ID of 4542 has been used in all releases of NASIS, NASIS 5.1 to the current NASIS data model. Must be unique within each database.

The Attribute IDs can be manually entered, if you want to control what attribute IDs are used or there is a calculation under ‘Attribute’ to populate null values. The calculation is called ‘Update Attribute ID where ID is null’. Running the calculation on the selected set, instead of highlighting specific rows, makes sure that all null Attribute IDs get populated.

**Tablecolumn**

**Column Default Sequence** The column default sequence is used to order the list of columns for each table. Required.

**Attributeiid** - The attribute that is associated with this tablecolumn record is determined by selecting the appropriate attribute from a drop down menu.

**Column Label** - A column label may contain a maximum of 80 characters. A column label may contain any valid printable character. Required.

**Physical Data Type** - Indicates the corresponding column's physical data type. The logical data type property is entered for the attribute. The logical data type and the physical data types should be compatible, for instance it would not make sense to select ‘datetime’ and the logical data type and then select ‘integer’ for the physical data type. Required.

#### **Process for adding/changing physical data types.**

Just adding a new physical data type to the domaindetail table or making changes to the domaindetail data is not sufficient to actually see these changes in NASIS because NASIS is metadata driven and the list physical data types are coming from the MetadataDomainDetail table, not the domaindetail table. There are two options for showing changes to the physical data types in NASIS: 1)Release a new version of NASIS which includes the domaindetail changes or 2)Make the changes to the physical data type in the MetadataDomainDetail table on the NASIS server database, wait a day, then reinitialize your local NASIS client database. There is a nightly BCP process at NITC that downloads all of the metadata tables from the NASIS server database, which is used when a NASIS client data is reinitialized. The DomainID value for physical data type is 5199.

**NOTE:** New physical data types cannot be added to the NASIS database without making application changes.

**Column Display Size(NASIS) -** The default display width of a column in spreadsheet view and reports, in characters .

.

**Default Type** - The type of data to be used as the default value for a table column. Below is the current list of default values that can be selected.

|  |  |
| --- | --- |
| Choice Sequence | Default Type |
| 1 | current date/time |
| 2 | literal |
| 3 | nasis group ID |
| 4 | nasis site ID |
| 5 | nasis user ID |
| 6 | identity |
| 7 | zero length string |
| 8 | propagate from hierarchy parent |
| 9 | non-quoted literal |
| 10 | propagated |

**Column Literal Default Value** – When literal is selected as the default type, enter the literal default value here.

|  |  |  |  |
| --- | --- | --- | --- |
| Table | Column | Default Type | Default Value |
| area | obterm | literal | 0 |
| attribute | attcasesensitive | literal | 0 |
| calculation | dataafuse | literal | 0 |
| chaashto | rvindicator | literal | 0 |
| chtexturegrp | stratextsflag | literal | 0 |
| columnlookup | branch | literal | 0 |
| pedon | pedrecorigin | literal | NASIS |
| soilseriesmlrasusing | mlraareatypeiidref | literal | 5 |
| soilseriesstatesusing | stateareatypeiidref | literal | 3 |

**Set Default on Row Change (NASIS)** –This is a checkbox for indicating which columns in a NASIS table should have their default values automatically updated whenever any change is made to an instance of the corresponding table. The most common columns with this setting are typically the ID of the NASIS user who last modified an instance of that table, and the date and time when that change occurred. Required, default = false.

Examples:

|  |  |
| --- | --- |
| Table Name | Column Name |
| area | recwlupdated |
| area | recuseriidref |
| areatext | recwlupdated |
| areatext | recuseriidref |
| areatype | recwlupdated |
| areatype | recuseriidref |

**Set Default on Object Change (NASIS) -** This is a checkbox for indicating which columns in a NASIS Data Object’s root table should have their default values automatically updated whenever any change is made to an instance of the corresponding NASIS Data Object. The most common columns with this setting are typically the ID of the NASIS user who last modified an instance of any table in that NASIS Data Object, and the date and time when that change occurred. Required, default = false.

Examples:

|  |  |
| --- | --- |
| Table Name | Column Name |
| areatype | objwlupdated |
| areatype | objuseriidref |
| calculation | objwlupdated |
| calculation | objuseriidref |
| datamapunit | objwlupdated |
| datamapunit | objuseriidref |

**Aggregation** - An integer number that uniquely identifies an attribute's ordinality (aggregation). Some attributes are recorded as a single representative value and some attributes are recorded as a high, low and representative value. An attribute's aggregation indicates how many physical columns a logical attribute resolves to. Required, default = ‘none’.

|  |
| --- |
| Aggregation Choices: |
| High, Low |
| High, Low, Representative Value |
| Representative Value |
| List |
| List, Representative Value |
| None |

**Column Not Null** - There is a checkbox for indicating which columns do not allow null values. Required, default = false.

**Column Calculable (NASIS)** – There is a checkbox for indicating which columns can be derived by a “calculation” script. If the “Protected” checkbox is not checked, it means the data can also be entered directly by a user. If the “Protected” checkbox is checked, it prevents the user from directly entering the data. Columns that are set to “Calculable” have another column added to the database with a suffix of “\_s”. This column is used to indicate the source of the data, was it “calculated” or “manually entered”.

For calculated columns, the logical data type must also be set to “calculable”. Required, default = unchecked.

**Column Visible (NASIS)** - There is a checkbox for indicating which columns should be visible in either the NASIS grid editor or one the special NASIS editors. There are two common reasons for not making a column visible: 1)to not show Foreign key iids, those that end in ‘iidref’ and 2)to hide an obsolete column from the application. Required, default value = checked.

**Column Protected** – There is a checkbox for indicating which columns should be prevented from being changed by end users. The following column columns are usually protected: 1) table record iids, those that end with “iid”, 2)foreign key iids, those that end with “iidref”, and 3) record and object when last updated columns. Required, default value = unchecked.

column\_unique There is a checkbox for indicating which columns mustbe unique. Required, default = unchecked.

.

column\_business\_oriented (NASIS) There is a checkbox for indicating which columns contains soil business related data. Required, default = false. Examples:

column\_sort\_sequence (NASIS) NULL examples:

column\_sort\_ascending (NASIS) NULL examples:

column\_sort\_type NULL examples and existing options

column\_alignment (NASIS) Describes how the display of data in a grid cell is aligned horizontally

.

in\_replication\_select\_list There is a checkbox for indicating which columns should be included in the corresponding table's replication selection list. Required, default = unchecked. This is used to determine which columns are displayed in the Local Database Setup module.

Give examples:

Why set this???

Give examples

column\_include\_in\_pedon\_pc There is a checkbox for indicating which columns are in the data model for Pedon PC. Required, default = unchecked.

fed\_geo\_data\_cmte There is a checkbox for indicating which columns are considered part of the Federal Geographic Data Committee soil metadata standard. Required, default = unchecked.

column\_server\_identity There is a checkbox for indicating which columns should be defined as an identity in the server database. Required, default = unchecked.

Example:

column\_spatialdata There is a checkbox for indicating which columns contain spatial type data. Required, default = unchecked. Logical and physical datatypes

column\_spatial\_reference\_id The spatial reference ID is a number that identifies the datum and projection for spatial data. Valid entries

Examples:

file\_content\_column\_iid\_ref The Column ID (MetadataTableColumn.ColumnID) of the column used to store the contents of a file. This is an attribute of the corresponding column. The logical data type of the corresponding column must be "File Reference". A File Reference type column will display a file name in a table editor grid, and will also include buttons labeled Open or Save. Open is used to import the contents of a file into the column identified by this attribute. Save is used to copy the imported file contents to a file whose name is stored in the File Reference column.

There's only one example, the VNIR file.

Examples of current data:

column\_field\_size\_override Use this column to override the attribute field size. This allows the same attribute to have different field lengths, in different tables. This is primarily for varchar and char columns.

column\_help\_text NULL

column\_id An integer value that uniquely identifies a column in a data model, not just within a table. Optional to enter, but must be populated. There is a calculation for this.

originating\_person NULL

contact\_person NULL

pending\_action NULL

pending\_status NULL

**Tablecolumn required columns, unique constraints and calculations**

**Columnlookup (NASIS ONLY)**

It is hard to describe all the possibilities. The basic idea is just to identify which columns will be displayed for a foreign key. But the columns can come from the parent table or tables higher in the hierarchy, so the relationship has to be specified for each table. There can be placeholder tables, meaning a table that doesn't have any display columns but is used to establish a relationship to a higher table. There can also be display columns from more than one parent path, which is what the branch attribute is used for.

Get a report to run to show all columnlooks – use this as a baseline: NREPO-QA-Mod-Dep- G1 Column Lookups - Show All

**Indexmaster**

constraint\_or\_index\_name The name that is used to physically implement an index in a database management system.

unique\_index A Boolean value that indicates if the corresponding index is a unique index.

foreign\_key\_index A Boolean value that indicates if the corresponding index is used for a foreignkey relationship.

primary\_index A Boolean value that indicates if the corresponding index is the primary index for the corresponding table.

clustered\_index A Boolean value that indicates if the corresponding index should be created as a clustered index.

index\_deferrable A Boolean value that indicates if the corresponding index can be deferred.

UC -

We defer some UCs so that a copy/paste can be performed. The pasted copy of a record might have a UC violation, so we store it in the local database and display an error message. The constraint is enforced in the server database, so it has to be resolved before uploading.

performance\_index A Boolean value that indicates if the corresponding index was created for performance reasons only.

spatial\_index A Boolean value that indicates if the corresponding index is a spatial index.

index\_column\_number For the associated index, how many columns are part of the index.

index\_column\_names NULL

index\_description For the associated index, the index description.

index\_id An integer value that uniquely identifies a table index in the NASIS-style metadata.

sql\_advance\_options To capture any SQL Server advance database/table options in the NREPO data.

originating\_person NULL

contact\_person NULL

pending\_action NULL

pending\_status NULL

Index master unique constraints

calculations

**Indexdetail**

**Relationshipmaster**

index\_iid\_ref An internal ID (integer) that is part (or all) of a key that uniquely identifies a record in another table. Also known as part (or all) of a "foreign key". In cases where the \_iid\_ref is used as part of a lookup (choice list) into another table, NASIS users can edit this value by entering a valid choice and thus "link" to a record in another table. In all other cases, this value is managed by NASIS and cannot be edited.

dependency\_index\_iid\_ref An internal ID (integer) that is part (or all) of a key that uniquely identifies a record in another table. Also known as part (or all) of a "foreign key". In cases where the \_iid\_ref is used as part of a lookup (choice list) into another table, NASIS users can edit this value by entering a valid choice and thus "link" to a record in another table. In all other cases, this value is managed by NASIS and cannot be edited.

system\_iid\_ref An integer number that uniquely identifies a database or information system repository in a particular database.

relationship\_name A name given to a relationship between two tables in a relational database. If there is more than one relationship between the same two tables, the name of each of those relationships must be unique., otherwise use ‘default’ as the relationship name.

The relationship name is used for the "join by name" expression in a NASIS query, so the names should be meaningful. The NASIS query generator converts "join by name" into actual join conditions for the named relationship. If there is only one possible join between two tables, the name "default" is usually used.

delete\_fail A Boolean value that indicates if the delete rule for the corresponding relationship is "fail". If the delete rule is not "fail", it is assumed to be "cascade".

cardinality Indicates whether the relationship between the left table and right table is one to one or one to many. For a one to one relationship, a record in the left table is related to zero or one record in the right table. For a one to many relationship, a record in the left table is related to zero or more records in the right table. Cardinality does not indicate whether or not the relationship is mandatory.

mandatory Indicates if in order for a record to exist in the right table of a relationship, a corresponding record must exist in the left table of that relationship, i.e. mandatory = "yes". In other words, when mandatory is "no", a record may exist in the right table of a relationship without having a corresponding record in the left table of that relationship.

As far as I can tell this just means that the foreign key can be null.

in\_hierarchy When InHierarchy is set, the two tables involved in the corresponding relationship are considered to be part of the same hierarchy. A hierarchy is a graph with a single root node where no branches ever converge. In a data model, a table may be a member of one and only one hierarchy. For the NASIS data model, MetadataTableCollection records all hierarchies that are defined. Other data models, like the one for the Staging Server, includes hierarchies but no table collections.

The drill-down operation in the table editor follows the in-hierarchy relationships. For example, the Legend Mapunit Area Overlap table is shown as a child of Legend Area Overlap and not Legend Mapunit, because the relationship between Legend Mapunit and Legend Mapunit Area Overlap is not in-hierarchy. It just provides a way of resolving multiple paths to the same table.

favorite\_child A Boolean value that indicates if the right table in a relationship corresponds to the left table's "favorite child". When visiting the "children" of a table, the "favorite child" is the default table that will be visited when no explicit child table is specified.

load\_find\_related A Boolean value that indicates whether or not Load Related and Find Related are enabled for the corresponding relationship.

This is normally set True for relationships between tables in different business objects (aka table collections). There might be some exceptions.

paste A Boolean value that indicates if records can be created in the "right" table of a relationship by copying and "pasting" records from the "left" table in that relationship. In other words, new records are pasted into the "right" table of a relationship, where the only columns populated either correspond to columns that also exist in the left table of that relationship, or columns that have a default value.

The explanation isn't quite right. Normally you can only paste data from a table into other rows of the same table. Pasting across a relationship means creating a record in the "right" table that contains a FK to the "left" record, instead of an actual copy. This is only possible if the remaining columns of the "right" table are nullable or have defaults.

constraint\_deferrable A Boolean value that indicates if the corresponding constraint can be deferred on the client database, i.e. it is not created.. Any other guidelines

Like deferrable UCs, we can also defer FKs in the local database because sometimes the parent record has not been downloaded.

foreign\_key\_constraint\_name The foreign key constraint name associated with the corresponding relationship.

relationship\_id An integer value that uniquely identifies a relationship between two tables in the NASIS-style metadata.

originating\_person NULL

contact\_person NULL

pending\_action NULL

pending\_status NULL

RM unique constraints

# **Common Columns**

Originating Person

Contact Person

Pending Action – add, change, delete

Pending status - approved, pending, rejected

# **Root Tables**

## **Which Columns should typically be Included in a NASIS Data Object’s Root Table? (NASIS Specific)**

An integer primary key column whose default type is “Identity” (autonumber). That column’s physical name is typically like “\*iid”.

A foreign key column that identifies the NASIS site that owns that record. That column’s physical name is typically like “\*dbiidref”.

A foreign key column that identifies the NASIS group that owns that record. That column’s physical name is typically “grpiidref”.

A datetime column that indicates when that NASIS Data Object instance was last updated. That column’s physical name is typically “objwlupdated”.

A foreign key column that indicates which NASIS user last updated that NASIS Data Object. That column’s physical name is typically “objuseriidref”.

A datetime column that indicates when the root table record of that NASIS Data Object instance was last updated. That column’s physical name is typically “recwlupdated”.

A foreign key column that indicates which NASIS user last updated the root table record of that NASIS Data Object instance. That column’s physical name is typically “recuseriidref”.

## **Which Columns should typically be Included in a NASIS Table that is not a NASIS Data Object’s Root Table? (NASIS Specific)**

An integer primary key column whose default type is “Identity” (autonumber). That column’s physical name is typically like “\*iid”.

An integer foreign key column identifying its “in hierarchy” parent table. That column’s physical name is typically like “\*iidref”, and its corresponding label is typically “Lineage”.

A datetime column that indicates when a record in that table was last updated. That column’s physical name is typically “recwlupdated”.

A foreign key column that indicates which NASIS user last updated a record in that table. That column’s physical name is typically “recuseriidref”.

Many non-root tables in NASIS include a column that allows an authorized user to explicitly sequence the child table records for the corresponding parent table record, because that column is also typically that table’s primary sort key. That column’s physical name is typically “seqnum”.

**Process to create release new data model**

Input data into the tables usually in this order: system, attribute, table collection, systemtable, tablecolumn, column lookup (nasis), index master, index detail, relationship master.

Many of the tables also have a history table – input data in these as you see fit.

**Create index** – guidelines

PK –iid

FK – iidref

Unique –

DI – performance

1 cluster index

**Create relationships** – guidelines

On parent table – PK index usually – must have the foreign indexes created first before you can create a relationship

**Create column lookups** (NASIS only) guidelines

Column leve l

Give examples, easy to hard – existing lookups

What metadata table this populates?

**Calculations**

Run necessary calculations: attribute, index master, relationship master, system table, table column, unit of measure, domain

**How to add a new Unit of Measure**

Must populate uomid - use either the calculation or manual enter a value. Must be unique.

**How to add a new physical data type**

**How to add a new logical data type:**

**How to add new choice lists (domains)**

**Reports**

All begin with ‘NREPO’ and are group by type.

**Create SQL Scripts Reports** – special ones by data model

NREPO - SQL Script – CreateTables, etc.

Enter Type C for Client Scripts, Type S for server scripts, Type O for Other

Prerequisites – what calculations, QA and other reports to run first

**Create Metadata Reports**

Prerequisites – what calculations, QA and other reports to run first

What systems to have in selected set

Map metadata tables – with source data – attribute, tablecolumn, generated, etc for each table.

Metadata visio diagram

**NREPO-Syle Metadata Reports**

– published reports on various websites – list these.

How to best format these – George instructions.

**CmpMod Reports**

**QA Reports**

Sytle Metadata Reports – pdf’s for websites

How to generate using George’s instructions

**SSURGO processes.**

Create SSURGO tables

**Pedon PC process.**

**Other tables:**

Locks – tablename\_l

Deletes – tablename\_d

Tablename\_x

When to create these and indexes on each

Settings – visible, selectable, editable, tabserveronly – any other setting???