SISIAL – Neotoma Crosswalk

Simon Goring

Table of contents

1 SISAL Overview **1**

2 Adding Speleothem Specific Tables **3**
2.1 Controlled Vocabularies 3
2.1.1 Speleothem Types 3
2.1.2 Speleothem Drip Types 3
2.1.3 Vegetation Cover 4
2.1.4 Land Use 6
2.1.5 Land Cover 7
2.1.6 Entity Statuses 8
2.1.7 Depth Reference 8
2.1.8 Speleothem Entity Geology 9

3 Speleothem Entity Table **9**
3.0.1 Working with TRUE/FALSE data 10

1 SISAL Overview

SISAL is a global database of speleothem data [@ref] that uses a standard template for data upload. SISAL is stored as a MySQL database that can be queried and updated by users. Relative to Neotoma, the database shares many common features, for example, it stores data across `sites`, it measures depths along collection elements, and records chronologies for records. However, there are several differences based on conceptual and analytic differences between the primary data within each database.

The largest work to align SISAL with Neotoma requires the addition of a new `entity` table linking `sites` and `collectionunits`. In SISAL this `entity` table includes a large number of secondary fields, including elements of vegetation and land use cover, with fixed vocabularies.

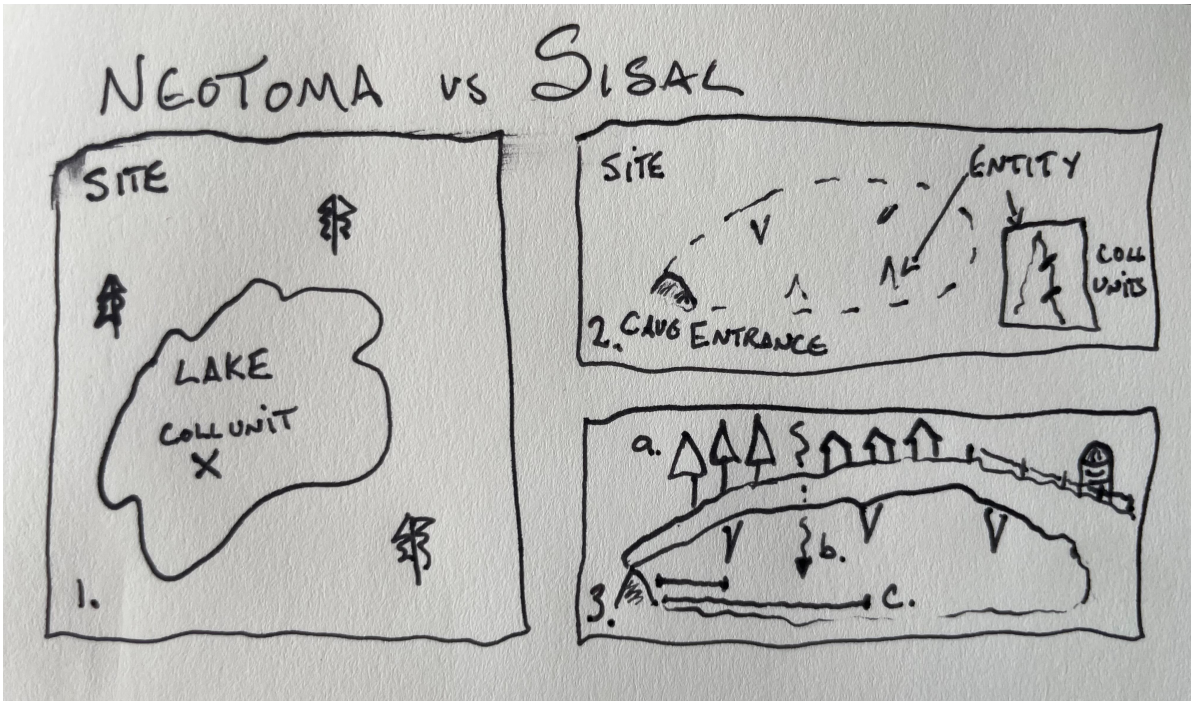

Figure 1: Differences between `site` and `collection unit` concepts in Neotoma and SISAL. In panel (1) a collection unit is a single core within a lake; the lake is treated as a homogeneous object, and any differences between cores within a lake would be noted in the `collectionunit.notes`. In panel (2) we see that an individual speleothem may have multiple cores, and each speleothem is an entity in its own right, with unique properties. In panel (3) we see that an individual entity may differ from entities within a single site (in this case a cave) based on surface vegetation, drip-water properties or distance from the cave.

2 Adding Speleothem Specific Tables

The speleothem entity table (named `speleothems`) is linked to the siteid. It helps link all speleothems together at a single site within a cave complex (for example). The entity is the specific speleothem, and each entity may have one or more collection units. So in the case of a speleothem, there is the organizing "Cave", the "Stalagtite", and then on that "Stalagtite" there is one or more collectionunits. Because the entity table relies on several fixed vocabularies as foreign key references, we define those first:

2.1 Controlled Vocabularies

2.1.1 Speleothem Types

- stalagmite
- stalactite
- flowstone
- composite
- mixed (add note to notes sheet)
- other (add note to notes sheet)
- unknown

```
CREATE TABLE IF NOT EXISTS ndb.speleothemtypes(
    speleothemtypeid serial PRIMARY KEY,
    speleothemtype varchar(50),
    speleothemtypenotes text,
    speleothemtypepublication int REFERENCES ndb.publications(publicationid)
)
```

2.1.2 Speleothem Drip Types

2.1.2.1 In SISAL

- seepage flow
- seasonal drip
- fast flow
- mixed (add note to notes sheet)
- other (add note to notes sheet)
- unknown

2.1.2.2 Baker et al. (1997)

Baker, A., Barnes, W. L. & Smart, P. L. Stalagmite drip discharge and organic matter fluxes in Lower Cave, Bristol. Hydrological Processes 11, 1541–1555 (1997). See reference here: https://www.nature.com/scitable/knowledge/library/drip-water-hydrology-and-speleothems-26394838/

- Seepage Flow
- Seasonal Drip
- Percolation Stream
- Shaft Flow
- Vadose Flow
- Subcutaneous Flow

```
CREATE TABLE IF NOT EXISTS ndb.speleothemdriptypes(
    speleothemdriptypeid serial PRIMARY KEY,
    speleothemdriptype varchar(50),
    speleothemdriptypenotes text,
    speleothemdriptypepublication int REFERENCES ndb.publications(publicationid)
);

CREATE TABLE IF NOT EXISTS ndb.entitydripheight(
    entityid int REFERENCES ndb.speleothems(entityid) ON DELETE CASCADE,
    speleothemdriptypeid int REFERENCES ndb.speleothemdriptypes(spelothemdriptypeid),
    entitydripheight real,
    entitydripheightunit int REFERENCES ndb.variableunits(variableunitid)
);
```

2.1.3 Vegetation Cover

Vegetation cover should encompass two main concepts, the cover type and the cover class. It would allow a many-to-one classification as well, providing the opportunity to describe classes as proportions of total cover within some defined bound. The SISAL database usues the following classification terms:

- evergreen
- deciduous
- shrubland
- grassland
- sparse
- barren
- mixed (add note to notes sheet)
- other (add note to notes sheet)

- unknown

We propose to modify this with a set of terms derived from the FAO Land Cover classification system:

- Barren: At least of area 60% is non-vegetated barren (sand, rock, soil) or permanent snow/ice with less than 10% vegetation.
- Permanent Snow and Ice: At least of area 60% is covered by snow and ice for at least 10 months of the year.
- Water Bodies: At least 60% of area is covered by permanent water bodies.
- Evergreen Needleleaf Forests: Dominated by evergreen conifer trees (>2m). Tree cover >60%.
- Evergreen Broadleaf Forests: Dominated by evergreen broadleaf and palmate trees (>2m). Tree cover >60%.
- Deciduous Needleleaf Forests: Dominated by deciduous needleleaf (larch) trees (>2m). Tree cover >60%.
- Deciduous Broadleaf Forests: Dominated by deciduous broadleaf trees (>2m). Tree cover >60%.
- Mixed Broadleaf/Needleleaf Forests: Co-dominated (40-60%) by broadleaf deciduous and evergreen needleleaf tree (>2m) types. Tree cover >60%.
- Mixed Broadleaf Evergreen/Deciduous Forests: Co-dominated (40-60%) by broadleaf evergreen and deciduous tree (>2m) types. Tree cover >60%.
- Open Forests: Tree cover 30-60% (canopy >2m).
- Sparse Forests: Tree cover 10-30% (canopy >2m).
- Dense Herbaceous: Dominated by herbaceous annuals (<2m) at least 60% cover.
- Sparse Herbaceous: Dominated by herbaceous annuals (<2m) 10-60% cover.
- Dense Shrublands: Dominated by woody perennials (1-2m) >60% cover.
- Shrubland/Grassland Mosaics: Dominated by woody perennials (1-2m) 10-60% cover with dense herbaceous annual understory.
- Sparse Shrublands: Dominated by woody perennials (1-2m) 10-60% cover with minimal herbaceous understory.
- Unclassified: Has not received a map label because of missing inputs.

```
CREATE TABLE IF NOT EXISTS ndb.vegetationcovertypes(
    vegetationcovertypeid serial PRIMARY KEY,
    vegetationcovertype varchar (128) UNIQUE,
    vegetationcovernotes text,
    vegetationpublicationid int REFERENCES ndb.publications(publicationid)
);

CREATE TABLE IF NOT EXISTS ndb.sitevegetationcover(
    siteid integer REFERENCES ndb.sites(siteid) ON DELETE CASCADE,
    vegetationcovertypeid REFERENCES ndb.vegetationcovertypes(vegetationcovertypeid) ON DELE
```

```
    vegetationcoverpercent integer,
    vegetationcovernotes text
);

CREATE TABLE IF NOT EXISTS ndb.entityvegetationcover(
    entityid integer REFERENCES ndb.speleothems(entityid) ON DELETE CASCADE,
    vegetationcovertypeid REFERENCES ndb.vegetationcovertypes(vegetationcovertypeid) ON DELE
    vegetationcoverpercent integer,
    vegetationcovernotes text
);
```

2.1.4 Land Use

Similarly we want to create a land use table:

- water body
- wetland
- forest
- farmland
- pasture
- concrete and built up
- mixed (add note to notes sheet)
- other (add note to notes sheet)
- unknown

The FAO Uses a system with the following classes:

- Forest
- Forest with natural or natural assisted regeneration
- Broadleaved forest
- Coniferous forest
- Bamboo or palm forest
- Mixed forest
- Forest plantations
- Broadleaved forest plantation
- Coniferous forest plantations
- Mixed forest plantations
- Other wooded lands
- Shrubs
- Fallow
- Wooded grassland
- Other land

- Natural and semi natural land
- Barren land
- Grassland
- Marshland
- Cultivated and managed land
- Annual crop
- Perennial crop
- Pastures
- Built up area (urban or rural)
- Inland water

```
CREATE TABLE IF NOT EXISTS ndb.landusetypes(
    landusecovertypeid serial PRIMARY KEY,
    landusecovertype varchar (128) UNIQUE,
    landusecovernotes text,
    landuseclasspublicationid int REFERENCES ndb.publications(publicationid)
);

CREATE TABLE IF NOT EXISTS ndb.sitelandusecover(
    siteid integer REFERENCES ndb.sites(siteid) ON DELETE CASCADE,
    landusecovertypeid REFERENCES ndb.vegetationcovertypes(vegetationcovertypeid) ON DELETE C
    landusecoverpercent integer,
    landusecovernotes text
);

CREATE TABLE IF NOT EXISTS ndb.entitylandusecover(
    entityid integer REFERENCES ndb.speleothems(entityid) ON DELETE CASCADE,
    landusecovertypeid REFERENCES ndb.vegetationcovertypes(vegetationcovertypeid) ON DELETE C
    landusecoverpercent integer,
    landusecovernotes text
);
```

2.1.5 Land Cover

I'm going to use this for speleothems only:

```
CREATE TABLE IF NOT EXISTS ndb.entitycovertypes(
    entitycoverid serial PRIMARY KEY,
    entitycovertype varchar(50),
    entitycovernotes text
)
```

```
CREATE TABLE IF NOT EXISTS ndb.entitycovers(
    entityid int REFERENCES ndb.speleothems(entityid) ON DELETE CASCADE,
    entitycoverid int REFERENCES ndb.entitycovertypes(entitycoverid),
    entitycoverthickness real,
    entitycoverunits int REFERENCES ndb.variableunits(variableunitsid)
)
```

2.1.6 Entity Statuses

Note that the status levels provided in the spreadsheet do not match the ones in the database.

```
CREATE TABLE IF NOT EXISTS ndb.speleothementitystatuses(
    entitystatusid serial PRIMARY KEY,
    entitystatus text
);

INSERT INTO ndb.speleothementitystatuses (entitystatus)
VALUES ('current'),
       ('current partially modified'),
       ('superseded'),
       ('completely supersedes'),
       ('completely superseded by'),
       ('partially supersedes'),
       ('partially superseded by'),
       ('not applicable');
```

2.1.7 Depth Reference

How is depth measured? I'm moving this to a higher level table (without "entity" in front of it) because I think this is important going forward regardless.

```
CREATE TABLE IF NOT EXISTS ndb.depthreferencesystem(
    depthreferencesystemid serial PRIMARY KEY,
    depthreference text,
    depthreferencenotes text
);

INSERT INTO ndb.depthreferencesystem(depthreference, depthreferencenotes)
VALUES
    ('from top', 'Measured Depth at section top (depth at bottom is depth + thickness).'),
```

```
    ('from base', 'Measured Depth at section top (depth at top is depth - thickness).'),
    ('midpoint', 'Measured depth at midpoint of the analysis unit (depth at top is depth - 0
    ('not applicable', 'Point measurement.'),
    ('unknown', 'Not reported or not submitted at time of data upload.'),
    ('assumed top', 'Not reported but submitted with assumed standard of top-depth reporting
    ('assumed bottom', 'Not reported but submitted with assumed standard of bottom-depth rep
    ('assumed midpoint', 'Not reported but submitted with assumed standard of midpoint-depth
```

2.1.8 Speleothem Entity Geology

For individual speleothems within mapped system, the underlying (or overlying) geology may differ. We want to create a table of acceptable geology parameters:

```
CREATE TABLE IF NOT EXISTS ndb.speleothementitygeology(
    speleothemgeologyid serial PRIMARY KEY,
    speleothemgeology text,
    speleothemgeologynotes text
);

INSERT INTO ndb.speleothementitygeology (speleothemgeology)
VALUES
    ('limestone'),
    ('unknown'),
    ('dolomite'),
    ('mixed (see notes)'),
    ('dolomite limestone'),
    ('marble'),
    ('marly limestone'),
    ('other (see notes)'),
    ('calcarenite');
```

3 Speleothem Entity Table

```
CREATE TABLE IF NOT EXISTS ndb.speleothems(
    siteid integer REFERENCES ndb.sites(siteid) ON DELETE CASCADE,
    entityid serial PRIMARY KEY,
    entityname text,
    entitystatusid integer REFERENCES ndb.speleothementitystatuses(entitystatusid),
```

```
    monitoring boolean,
    relativeageid integer REFERENCES ndb.relativeages(relativeageid),
    entrancedistance real,
    entrancedistanceunits int REFERENCES ndb.variableunits(variableunitsid)
    speleothemtypeid int REFERENCES ndb.speleothemtypes(speleothemtypeid)
);
```

3.0.1 Working with TRUE/FALSE data

There are a number of columns in the SISAL database that refer to elements such as fluid inclusions, organics or other elements that may be recorded but are not explicitly a part of the SISAL database. For these we use the table `externalspeleothemdata`. This allows us to define these elements and refer to an external resource using a PID. These would include (for example) references to noble_gas_temperatures, clumped_isotopes or fluid_inclusions. The structure of the `ndb.externaldatabases` table allows us to refer to external resources such as Dryad, PANGEA or CrossRef, if data is included within a publication. The table also supports a many-to-many relationship, so a single entity can be linked to multiple databases for various reasons.

```
CREATE TABLE IF NOT EXISTS externalspeleothemdata(
    entityid integer REFERENCES ndb.speleothems(entityid) ON DELETE CASCADE,
    externalid text,
    extdatabaseid integer REFERENCES ndb.externaldatabases(extdatabaseid),
    externaldescription text
);
```