**Tracking data provenance, use and attribution within the BIEN3 database**

Brad Boyle

2 Oct. 2013

**Contents**

1. Introduction 1

2. Terms and definitions 2

3. The dataset as the unit of citation 2

4. Applications of metadata on data provenance, use and attribution 3

4.1. Why are data use and attribution metadata important? 3

4.2. Examples uses 3

5. A data use and attribution schema extension for BIEN 3 5

5.1. Overview of the schema 5

5.2. Table definitions 6

5.3. Example of data use and attribution based on the proposed schema 9

# 1. Introduction

The BIEN database consists of millions of biological observations contributed by numerous data providers around the world. The success of BIEN depends to a large degree on how well it respects the wishes and concerns of those data providers regarding access to and acknowledgment of their data. Respecting those wishes and concerns will in turn hinges on the ability of BIEN to manage metadata describing data providers, data users, and their relationships to specific datasets.

Solving the metadata challenge will provide additional benefits for the long-term sustainability of BIEN. Storing and tracking information pertaining to usage and attribution will enable automation the delivery of this information to data providera and users alike. Information on data ownership and conditions of access can be used to enable data users to contact data owners directly, thus minimizing the need for database administrators to devote scarce time to forwarding requests or otherwise serving as messengers between data users and data providers.

This document provides an overview of key data elements needed for describing and managing information pertaining to datasets, data providers and data users. It also gives examples applications of this information and presents a proposed data provider and data usage extension to the vegBIEN schema.

# 2. Terms and definitions

Below are definitions of key terms related to data provenance, use and attribution.

*data provider*: a person or institution responsible for creating or distributing data.

*dataset*: a collection of records from the same place, with the same attribution requirements. each flat file from a data provider is a separate dataset, but gets split apart into multiple datasets if different rows are from different providers (e.g. an aggregator like GBIF would be split into separate datasets for each herbarium, even though the data is from the same flat file).

*proximate data provider*: the single data provider from whom BIEN obtained a dataset.

*data author:* a person or institution responsible for creating the dataset. A dataset can have one or more authors, or none if the author is unknown. Subset of data provider.

*data publisher*: the data provider who is the person or institution responsible for originally publishing the dataset. Publishing typically means digitizing and distributing the data via the internet. Normally a dataset has zero (if unpublished) or one data publishers. However, it is possible to have more than one data publisher if alternative versions of a dataset have been published. For example, different versions of the Alwyn Gentry Forest Transect Dataset were published by MBG and by SALVIAS. Subset of data provider.

*data indexer*: an institution which distributes data obtained from multiple data publishers. Also called an *aggregator*. A dataset can have zero to many data indexers. Subset of data provider.

*primary contact*: the single data provider (person or institution) to whom questions regarding use and attribution of a dataset should first be directed. Each dataset should have one primary contact.

*alternate contact*: additional data providers (persons or institutions) to whom questions regarding use and attribution of a dataset can be directed, after first attempting to contact the primary contact. A dataset can have zero, one or many alternate contacts.

# 3. The dataset as the unit of citation

As stated in ‘Recommended practices for citation of data published through the GBIF network’[[1]](#footnote-1) the unit for citation is the dataset. For this reason, all records in a dataset must originate from a common set of data providers.

In terms of database rules, this definition has some important implications. One implication is that records obtained from a data indexer or aggregator represent many datasets, not one. For example, a data dump of specimen records from GBIF consists of many datasets obtained from many herbaria. "SALVIAS Plots" is not one but several datasets from different data providers, each with their author data access conditions. A second implication is that specimens obtained directly from a herbarium and specimens obtained from an indexer always represent separate datasets, even when all records ultimately originated from the same herbarium. This difference is demonstrated by the different citation requirements for these two datasets. For example, the source citation for ARIZ records obtained from ARIZ is "ARIZ", whereas the source citation for ARIZ records obtained from GBIF is "ARIZ, GBIF". Finally, note that access conditions for a given dataset can vary by data provider within the same set of providers. For example, Peter Jorgensen and MBG are both data providers for the Madidi Forest Plots (Jorgensen the data author and MBG the data publisher). Whereas Jorgensen requires offer of co-authorship as a condition of use of his data, MBG requires only citation.

# 4. Applications of metadata on data provenance, use and attribution

## 4.1. Why are data use and attribution metadata important?

In the context of BIEN, metadata pertaining to data providers, datasets and data users have several important applications. These include:

1. Acknowledgment and citation of data providers
2. Tracking access to specific datasets by BIEN users
3. Tracking publications based on specific datasets
4. Communicating conditions-of-use
5. Regulating access to data
6. Communicating with data providers
7. Identifying who provided the data directly to BIEN

## 4.2. Examples uses

1. Acknowledgement and citation of data providers

* All providers of all datasets used for a publication should be cited directly in the acknowledgments section of that publication. Alternatively, very long lists of data providers can be acknowledge via a link to an acknowledgment page on the BIEN website. A stable url should be provided to ensure permanence of this acknowledgement [discuss: can BIEN realistically provide stable urls? Or should this be role of publisher?]
* The BIEN database should support the automatic extraction of lists of data providers and data provider roles (see definitions, below) for a given dataset
* GBIF recommends that "all data providers should be individually cited, and the role played by each contributor can be recorded in the citation."[[2]](#footnote-2) In BIEN terminology, this means listing all data providers and indicating whether they are authors, publishers or indexer. As stated above, this list should be included directly in the acknowledgments section of a publication, or, if the list is large, on a linked page on the BIEN website or the publisher's website.
* Example applications of the above terminology can be seen on the Participants section of the BIEN website[[3]](#footnote-3),[[4]](#footnote-4).

1. Tracking access to specific datasets by BIEN users

* The BIEN database should support tracking of data download by BIEN users
* Users must register to download data
* Download events (user x dataset x timestamp) are tracked
* Users agree to tracking as a condition of access
* Data providers may view download logs showing all download events for their data

1. Tracking publications based on specific datasets

* Data providers should also be able to view summaries of all publications based on their datasets
* As long as users register to download data (see #2 above) and also report publications involving downloaded data, tracking of publications for datasets should be automatic

1. Communicating conditions-of-use

* All datasets have a default condition of use
* Most likely this should be permission to track downloads + citation in publications + report publications to BIEN.
* Data providers can set additional conditions of use and for individual datasets
* The restrictions can apply to all users or be changed for individual users
  + Example: data provider A requires coauthorship for use of dataset X, but citation only for use of dataset Y?

1. Regulating access to data

* All datasets have a default access level
* These can be set by a data provider as public (full data available without request), by request only (only metadata searchable without request), or hidden (visible to data providers and BIEN DB administrators)
* Data providers can change access for levels individual users. Examples:
* The default condition of access for dataset X, as set by data provider C, is "by request only".
* However, data provider C allows users J, K and L to download the data directly without requesting permission as they are collaborators.
* User Q requests permission from data provider C to access dataset X. Data provider C authorizes user C to download dataset X.

1. Communicating with data providers

* The BIEN DB should contain all the information needed to contact key data providers regarding access to specific datasets
* This information should be sufficient to support direct contact between BIEN users and data providers, without the need for intervention by BIEN DB administrators
* Designating primary and alternate contacts for each dataset minimizes the need for intervention by BIEN DB administrators

1. Identifying who provided the data directly to BIEN

* Can be important for resolving discrepancies between the original dataset and the version in BIEN
* Particularly important for datasets with more than one data provider

# 5. A data use and attribution schema extension for BIEN 3

## 5.1. Overview of the schema

In this section I take a first stab at a relational schema which models the concepts just described. The schema, illustrated by the entity relationship diagram (erd) in Fig. 1, would be implemented as an extension to the core BIEN3 database (vegBIEN). For more detail, see the SQL schema definition in Appendix 1, or the original MySQL Workbench file (ask me).

Although the schema is fairly highly specified (I've gone so far as to declare participation constraints and a number of business rule) my goal is not to "cast in stone" a final schema, but rather to provide a productive starting point for discussion. Based on past experience, we'll make fast progress at converging on the final schema if we start out with a fully fleshed-out model rather just generic table definitions and a handful of requirements. Bottom line is that any part of this schema may be altered or discarded entirely, as long as what replaces it can still satisfy the use example given earlier.

The main tables of the schema are `dataProvider`, `user` and `dataset`. `dataProvider` and `user` are both strict subsets of the existing vegBIEN table `party`. That is, the relationships `dataProvider`:`party` and `user`:`party` are 1:1, with optional participation by table `party`. `party` represents any person or institution within the database, whereas `dataProviders` are those parties which have a relationship to one or more datasets in one or more roles as a data provider. This relationship is described by table `dataProvider\_dataset`. Links between `dataProvider\_dataset` and lookup tables `dataProviderRole` and `accessCondition` allow data providers to have one or more roles (author, publisher, etc.) and specify one or more access conditions ("citation", "authorship", etc.) with respect to a dataset.

A `user` is any party who is person registered in the BIEN database with a username and password, and who in theory is capable of accessing datasets directly. I realize that creation of a user interface is out of scope for the current BIEN development cycle. However, even in the absence of a user interface, we still need a mechanism for tracking data use. A `user` table, linked to the existing table `party`, provides a straightforward means of doing so, and provides forward-compatibility in the event we do develop a user interface and provide tools for users to download data directly.

Table `dataset` enables sets of observations to be groups such that they are related to a shared set of data providers, access permission, and conditions of use. On the user end, access to datasets are tracked by table `downloadLog`. User-specific permission for accessing individual datasets are assigned via table `user\_dataset\_permission`, which also connects to a lookup table of standard permissions (see `permission` under Table definitions, below).

I have represented the relationship between datasets and biological observations in the BIEN database using a hypothetic table `observation`. In fact, I'm still uncertain how we should manage that link. In SALVIAS, data access and download logging is to users assigned at the level of individual plots. However, plots are grouped into "projects", and it is the project which is the unit of management for data providers and database administrators. That is, a given data provider can only see and manage access for plots belonging to a project for which that provider has administrative rights. The data provider can set permissions for all plots within a project, or for any given plot individually. Although this approach works well, I'm not certain how or if it can be applied to BIEN, which contains specimens (represented by individual records) and traits, in addition to plot. However, I do not think we should go down the road of defining datasets and assigning permissions and conditions at the level of individual observations, which in BIEN can be a single measurement of an individual tree. Although the ERD implies exactly that level of granularity, I think we need to discuss a more manageable way of grouping observations into datasets.

## 5.2. Table definitions

* *`dataProvider`*: person or institution related to one or more datasets by one or more roles as defined in table `dataProviderRole`. Subset of table `*party*`.
* `user`: person registered with BIEN database for purposes of accessing data or as administrator. Subset of table `*party`*.
* *`dataset`*: a collection of records from the same place, with the same attribution requirements. each flat file from a data provider is a separate dataset, but gets split apart into multiple datasets if different rows are from different providers (e.g. an aggregator like GBIF would be split into separate datasets for each herbarium, even though the data is from the same flat file).
* *`dataProviderRole`*: lookup of pre-defined roles of dataProviders with respect to datasets. Values (see **Definitions**, above):
* *author*: zero to one per dataset (where zero='unknown')
* *publisher*: zero to one per dataset
* *indexer*: zero to one per dataset
* *primary* contact: one primary contact per dataset (business rule)
* *secondary* contact: zero to many secondary contacts per dataset
* *`dataProvider\_dataset`*: links a dataProvider to a specific dataset. One dataProvider may be linked to multiple datasets. Each dataProvider must have at least one role, as defined by table `dataProvider\_dataset\_role` (business rule).
* `*accessCondition*`: lookup of access conditions. Access conditions can be pre-defined (e.g., "citation required") or custom.
* *`dataProvider\_dataset\_accessCondition`*: links one or more conditions of access, as specified by a dataProvider, to a specific dataset.
* `*downloadLog*`: records download events of specific dataset by specific users at specific times.
* `*permission*`: lookup of pre-defined data access permissions. Permissions describe access to datasets assigned to individual users, as follows:
* 0 - hidden: metadata and data are visible only to data providers and BIEN db administrators
* 1 - metadata only: metadata is visible and searchable without permission, but full data is downloadable only by request to data provider.
* 2 - full access: full data may be downloaded without prior permission
* 3 - can grant access: full data may be downloaded without prior permission, and user may change permissions to for other users for this dataset.
* `*user\_dataset\_permission*`: grants a permission level to a specific user for a specific dataset. Users without a specific permission assigned to them in this table are assumed to have either (a) full access (level 3) if they are a data provider for that dataset, or (b) the default permission for this dataset, as defined in table `dataset`, if they are not a data provider for that dataset.

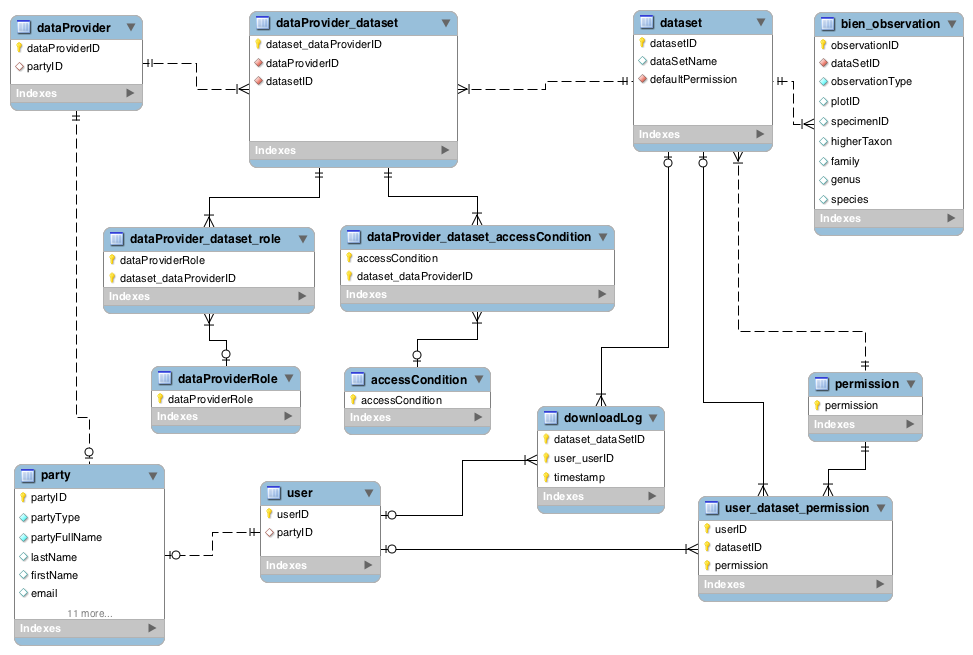


Fig. 1. Proposed schema for managing metadata pertaining to datasets, data providers and users within the BIEN database. Table `party` is an existing table from the current vegBIEN schema. Table `observation` is a hypothetical placeholder for a biological observation within the vegBIEN schema, included to demonstrate how one or more specimen, plot or trait observations are grouped together as datasets.

## 5.3. Example of data use and attribution based on the proposed schema

To aid our discussion of the merits and shortcomings of the proposed schema extension, I've prepared a couple of hypothetical views of familiar datasets. These views show what queries of actual data, stored in the proposed schema, might look like. To understand them, it helps to be familiar with the terms in section **2. Terms and definitions.**

Table 1. Examples of data sets and default access (permission) levels

|  |  |
| --- | --- |
| datasetName | defaultPermission |
| Madidi Transects | by request |
| Kew specimens | public |
| Boyle transects | by request |
| Gentry transects | public |
| Enquist lab transects | by request |
| MGB specimens | public |

Table 2. Example view of data providers, their roles and requested conditions of access for plot and specimen datasets. Values within a column separated by semi-colons are separate records within a child table (see example schema, Fig. 1).

| **dataset** | **dataProvider** | **dataProviderRole** | **accessCondition** |
| --- | --- | --- | --- |
| Kew specimens | Kew | Publisher | Citation |
| Kew specimens | GBIF | Indexer; Primary provider | Citation |
| MGB specimens | MBG | Publisher | Citation |
| MGB specimens | Peter Jorgensen | Primary contact | None |
| ARIZ specimens | ARIZ | Publisher | Citation |
| ARIZ specimens | Michelle McMahon | Primary contact | None |
| Boyle transects | Brad Boyle | Author; Primary contact | Citation; Coauthorship |
| Boyle transects | SALVIAS | Publisher; Primary provider | Citation |
| Gentry transects | Alwyn Gentry | Author | Citation |
| Gentry transects | MBG | Publisher | Citation |
| Gentry transects | SALVIAS | Publisher; Primary provider; Primary contact | Citation |
| Enquist lab transects | Brian Enquist | Author; Primary contact | Citation; Coauthorship |
| Enquist lab transects | Brad Boyle | Author; Secondary contact | Citation |
| Enquist lab transects | SALVIAS | Publisher; Primary provider | Citation |
| Madidi Transects | Peter Jorgensen | Author; Primary provider; Primary contact | Citation; Coauthorship |
| Madidi Transects | MBG | Publisher | Citation |

Appendix 1. SQL to create the schema shown in Fig. 1.

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL';

-- -----------------------------------------------------

-- Table `permission`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `permission` ;

CREATE TABLE IF NOT EXISTS `permission` (

`permission` VARCHAR(250) NOT NULL COMMENT '\"Hidden\",\"Metadata only\",\n\"Full data\",\"Can assign access\"' ,

PRIMARY KEY (`permission`) ,

UNIQUE INDEX `accessLevel\_UNIQUE` (`permission` ASC) )

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `dataset`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataset` ;

CREATE TABLE IF NOT EXISTS `dataset` (

`datasetID` INT(11) UNSIGNED NOT NULL AUTO\_INCREMENT ,

`dataSetName` VARCHAR(100) NULL DEFAULT NULL ,

`defaultPermission` INT NOT NULL ,

PRIMARY KEY (`datasetID`) ,

UNIQUE INDEX `dataSourceName` (`dataSetName` ASC) ,

INDEX `fk\_dataset\_dataPermission1` (`defaultPermission` ASC) ,

CONSTRAINT `fk\_dataset\_dataPermission1`

FOREIGN KEY (`defaultPermission` )

REFERENCES `permission` (`permission` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = MyISAM

DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------

-- Table `party`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `party` ;

CREATE TABLE IF NOT EXISTS `party` (

`partyID` INT(11) UNSIGNED NOT NULL AUTO\_INCREMENT ,

`partyType` VARCHAR(20) NOT NULL DEFAULT 'person' COMMENT '\'person\',\'institution\'' ,

`partyFullName` VARCHAR(100) NOT NULL ,

`lastName` VARCHAR(50) NULL DEFAULT NULL ,

`firstName` VARCHAR(50) NULL DEFAULT NULL ,

`email` VARCHAR(150) NULL DEFAULT NULL ,

`primaryInstitution` VARCHAR(255) NULL DEFAULT NULL ,

`institutionName` VARCHAR(150) NULL DEFAULT NULL ,

`department` VARCHAR(150) NULL DEFAULT NULL ,

`address1` VARCHAR(150) NULL DEFAULT NULL ,

`address2` VARCHAR(150) NULL DEFAULT NULL ,

`city` VARCHAR(50) NULL DEFAULT NULL ,

`stateProvince` VARCHAR(50) NULL DEFAULT NULL ,

`postalCode` VARCHAR(25) NULL DEFAULT NULL ,

`country` VARCHAR(50) NULL DEFAULT NULL ,

`stateProvinceCode` VARCHAR(20) NULL DEFAULT NULL ,

`user\_userID` INT NOT NULL ,

PRIMARY KEY (`partyID`) ,

INDEX `partyType` (`partyType` ASC) ,

INDEX `lastName` (`lastName` ASC) ,

INDEX `firstName` (`firstName` ASC) ,

INDEX `fullName` (`partyFullName` ASC) ,

INDEX `institutionName` (`institutionName` ASC) ,

INDEX `country` (`country` ASC) )

ENGINE = MyISAM

DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------

-- Table `taxon`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `taxon` ;

CREATE TABLE IF NOT EXISTS `taxon` (

`taxonID` INT(10) UNSIGNED NOT NULL AUTO\_INCREMENT ,

`parentTaxonID` INT(10) UNSIGNED NULL DEFAULT NULL ,

`taxon` VARCHAR(255) NOT NULL ,

`rank` VARCHAR(255) NULL DEFAULT NULL ,

`nameClass` VARCHAR(50) NULL DEFAULT NULL ,

`taxonAuthor` VARCHAR(255) NULL DEFAULT NULL ,

`family` VARCHAR(150) NULL DEFAULT NULL ,

`genus` VARCHAR(100) NULL DEFAULT NULL ,

`species` VARCHAR(100) NULL DEFAULT NULL ,

`leftIndex` INT(11) NULL DEFAULT NULL ,

`rightIndex` INT(11) NULL DEFAULT NULL ,

`isAccepted` INT(1) NULL DEFAULT NULL ,

`isHybrid` INT(10) UNSIGNED NOT NULL DEFAULT '0' ,

PRIMARY KEY (`taxonID`) ,

UNIQUE INDEX `leftIndex` (`leftIndex` ASC) ,

UNIQUE INDEX `rightIndex` (`rightIndex` ASC) ,

INDEX `taxon` (`taxon` ASC) ,

INDEX `taxonAuthor` (`taxonAuthor` ASC) ,

INDEX `rank` (`rank` ASC) ,

INDEX `nameClass` (`nameClass` ASC) ,

INDEX `family` (`family` ASC) ,

INDEX `genus` (`genus` ASC) ,

INDEX `species` (`species` ASC) ,

INDEX `isHybrid` (`isHybrid` ASC) ,

INDEX `parentTaxonID` (`parentTaxonID` ASC) )

ENGINE = MyISAM

DEFAULT CHARACTER SET = utf8;

-- -----------------------------------------------------

-- Table `dataProvider`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataProvider` ;

CREATE TABLE IF NOT EXISTS `dataProvider` (

`dataProviderID` VARCHAR(45) NOT NULL ,

`partyID` INT(11) UNSIGNED NULL ,

PRIMARY KEY (`dataProviderID`) ,

INDEX `fk\_dataProvider\_party1` (`partyID` ASC) ,

CONSTRAINT `fk\_dataProvider\_party1`

FOREIGN KEY (`partyID` )

REFERENCES `party` (`partyID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `dataProviderRole`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataProviderRole` ;

CREATE TABLE IF NOT EXISTS `dataProviderRole` (

`dataProviderRole` VARCHAR(50) NOT NULL COMMENT '\"Author\",\"Publishe\",\"indexer\",\"Primary contact\",\"Secondary contact\"' ,

PRIMARY KEY (`dataProviderRole`) ,

UNIQUE INDEX `dataProviderRole\_UNIQUE` (`dataProviderRole` ASC) )

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `dataProvider\_dataset`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataProvider\_dataset` ;

CREATE TABLE IF NOT EXISTS `dataProvider\_dataset` (

`dataset\_dataProviderID` VARCHAR(45) NOT NULL ,

`dataProviderID` VARCHAR(45) NOT NULL ,

`datasetID` INT(11) UNSIGNED NOT NULL ,

INDEX `fk\_datasource\_dataProvider1` (`dataProviderID` ASC) ,

PRIMARY KEY (`dataset\_dataProviderID`) ,

INDEX `fk\_datasource\_dataset1` (`datasetID` ASC) ,

CONSTRAINT `fk\_datasource\_dataProvider1`

FOREIGN KEY (`dataProviderID` )

REFERENCES `dataProvider` (`dataProviderID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_datasource\_dataset1`

FOREIGN KEY (`datasetID` )

REFERENCES `dataset` (`datasetID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `user`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `user` ;

CREATE TABLE IF NOT EXISTS `user` (

`userID` VARCHAR(45) NOT NULL ,

`partyID` INT(11) UNSIGNED NULL ,

INDEX `fk\_user\_party1` (`partyID` ASC) ,

PRIMARY KEY (`userID`) ,

UNIQUE INDEX `userID\_UNIQUE` (`userID` ASC) ,

UNIQUE INDEX `partyID\_UNIQUE` (`partyID` ASC) ,

CONSTRAINT `fk\_user\_party1`

FOREIGN KEY (`partyID` )

REFERENCES `party` (`partyID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `user\_dataset\_permission`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `user\_dataset\_permission` ;

CREATE TABLE IF NOT EXISTS `user\_dataset\_permission` (

`userID` VARCHAR(45) NULL ,

`datasetID` INT(11) UNSIGNED NULL ,

`permission` INT NOT NULL ,

PRIMARY KEY (`userID`, `datasetID`, `permission`) ,

INDEX `fk\_user\_data\_access\_dataset1` (`datasetID` ASC) ,

INDEX `fk\_user\_data\_permissions\_dataPermission1` (`permission` ASC) ,

CONSTRAINT `fk\_user\_data\_access\_user1`

FOREIGN KEY (`userID` )

REFERENCES `user` (`userID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_user\_data\_access\_dataset1`

FOREIGN KEY (`datasetID` )

REFERENCES `dataset` (`datasetID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_user\_data\_permissions\_dataPermission1`

FOREIGN KEY (`permission` )

REFERENCES `permission` (`permission` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `downloadLog`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `downloadLog` ;

CREATE TABLE IF NOT EXISTS `downloadLog` (

`dataset\_dataSetID` INT(11) UNSIGNED NULL ,

`user\_userID` VARCHAR(45) NULL ,

`timestamp` TIMESTAMP NOT NULL ,

PRIMARY KEY (`dataset\_dataSetID`, `user\_userID`, `timestamp`) ,

INDEX `fk\_downloadLog\_user1` (`user\_userID` ASC) ,

CONSTRAINT `fk\_downloadLog\_dataset1`

FOREIGN KEY (`dataset\_dataSetID` )

REFERENCES `dataset` (`datasetID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_downloadLog\_user1`

FOREIGN KEY (`user\_userID` )

REFERENCES `user` (`userID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `bien\_observation`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `bien\_observation` ;

CREATE TABLE IF NOT EXISTS `bien\_observation` (

`observationID` INT NOT NULL ,

`dataSetID` INT(11) UNSIGNED NOT NULL ,

`observationType` VARCHAR(45) NOT NULL COMMENT '\"plot\",\"specimen\",\"trait\",\"other\"' ,

`plotID` VARCHAR(45) NULL ,

`specimenID` VARCHAR(45) NULL ,

`higherTaxon` VARCHAR(45) NULL ,

`family` VARCHAR(45) NULL ,

`genus` VARCHAR(45) NULL ,

`species` VARCHAR(250) NULL ,

PRIMARY KEY (`observationID`) ,

INDEX `fk\_bien\_observation\_dataset1` (`dataSetID` ASC) ,

UNIQUE INDEX `observationID\_UNIQUE` (`observationID` ASC) ,

CONSTRAINT `fk\_bien\_observation\_dataset1`

FOREIGN KEY (`dataSetID` )

REFERENCES `dataset` (`datasetID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `dataProvider\_dataset\_role`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataProvider\_dataset\_role` ;

CREATE TABLE IF NOT EXISTS `dataProvider\_dataset\_role` (

`dataProviderRole` VARCHAR(50) NULL ,

`dataset\_dataProviderID` VARCHAR(45) NOT NULL ,

PRIMARY KEY (`dataProviderRole`, `dataset\_dataProviderID`) ,

INDEX `fk\_dataProvider\_dataset\_role\_dataProvider\_dataset1` (`dataset\_dataProviderID` ASC) ,

CONSTRAINT `fk\_dataProvider\_dataset\_role\_dataProviderRole1`

FOREIGN KEY (`dataProviderRole` )

REFERENCES `dataProviderRole` (`dataProviderRole` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_dataProvider\_dataset\_role\_dataProvider\_dataset1`

FOREIGN KEY (`dataset\_dataProviderID` )

REFERENCES `dataProvider\_dataset` (`dataProviderID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `accessCondition`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `accessCondition` ;

CREATE TABLE IF NOT EXISTS `accessCondition` (

`accessCondition` VARCHAR(100) NOT NULL ,

PRIMARY KEY (`accessCondition`) ,

UNIQUE INDEX `accessCondition\_UNIQUE` (`accessCondition` ASC) )

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `dataProvider\_dataset\_accessCondition`

-- -----------------------------------------------------

DROP TABLE IF EXISTS `dataProvider\_dataset\_accessCondition` ;

CREATE TABLE IF NOT EXISTS `dataProvider\_dataset\_accessCondition` (

`accessCondition` VARCHAR(100) NULL ,

`dataset\_dataProviderID` VARCHAR(45) NOT NULL ,

PRIMARY KEY (`accessCondition`, `dataset\_dataProviderID`) ,

INDEX `fk\_dataProvider\_dataset\_accessConditions\_dataProvider\_dataset1` (`dataset\_dataProviderID` ASC) ,

CONSTRAINT `fk\_dataProvider\_dataset\_accessConditions\_accessConditions1`

FOREIGN KEY (`accessCondition` )

REFERENCES `accessCondition` (`accessCondition` )

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_dataProvider\_dataset\_accessConditions\_dataProvider\_dataset1`

FOREIGN KEY (`dataset\_dataProviderID` )

REFERENCES `dataProvider\_dataset` (`dataset\_dataProviderID` )

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

1. <http://www.gbif.org/orc/?doc_id=4659&l=en> [↑](#footnote-ref-1)
2. <http://www.gbif.org/orc/?doc_id=4659&l=en> [↑](#footnote-ref-2)
3. <http://bien.nceas.ucsb.edu/bien/people/data-contributors/> [↑](#footnote-ref-3)
4. <http://bien.nceas.ucsb.edu/bien/people/data-indexers/> [↑](#footnote-ref-4)