MFE Database Table Descriptions

Kaija Gahm

2021-04-02

Contents

Introduction

The following is a list of overview “blurbs” for the tables in the MFE database. This is designed for a new person approaching the database and trying to understand where the tables come from and how they generally fit together.

Table Descriptions

BACTERIAL_PRODUCTION_BENTHIC

Lab incubations to measure benthic bacterial C productivity.

Variables:

```
##  [1] "projectID"
##  [2] "sampleID"
##  [3] "lakeID"
##  [4] "siteName"
##  [5] "dateSample"
##  [6] "dateTimeSample"
##  [7] "depthClass"
##  [8] "depthTop"
##  [9] "depthBottom"
## [10] "benthicBacterialProductionVolume_ugC_L_h"
## [11] "benthicBacterialProductionArea_mgC_m2_h"
## [12] "incubationDuration_h"
## [13] "metadataID"
## [14] "comments"
## [15] "flag"
## [16] "updateID"
```

BACTERIAL_PRODUCTION_PELAGIC

Lab incubations to measure pelagic bacterial productivity.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"                "sampleID"
##  [3] "lakeID"                   "siteName"
##  [5] "dateSample"               "dateTimeSample"
##  [7] "depthClass"               "depthTop"
##  [9] "depthBottom"              "BacterialProduction_ugC_L_h"
## [11] "incubationDuration_h"     "metadataID"
## [13] "comments"                 "updateID"
```

BENTHIC_INVERTS

Corresponding samples are in BENTHIC_INVERT_SAMPLES, joined by `sampleID`.

This table holds data about benthic invertebrates. Each row is a bug. Data includes size (length, mass, etc.), taxonomic columns, comments made during ID, etc.

The data in this table come from a variety of sources. Inverts are collected as part of the routine Long Lake sampling, as well as for a cross-lake survey in 2012, for the Crazy Chiro survey in 2015-16, etc. See the metadataID portion of the `sampleID` for data origin.

The `metadataID` column in this table should refer to the *lab processing methods* used for the bug in question, *not* to the sampling methods. Sampling methods should be covered by the metadataID portion of the sampleID, which corresponds to the `metadataID` column in BENTHIC_INVERT_SAMPLES.

Primary key: `invertID`

A couple database philosophy notes:

- This table currently contains taxonomic columns, which are potentially redundant with the columns in OTU. We are considering removing those columns, but haven't done so yet.
- This table contains the columns that compose the sampleID's, such as depthBottom, lakeID, and dateTimeSample. We have been discussing removing those as well, and having this only connect to BENTHIC_INVERT_SAMPLES by the sampleID.

Other important connections:

- `taxa` connects to `otu` in OTU
- `dryMass` is computed using the equations in DRY_MASS_EQUATIONS

Variables:

```
##  [1] "projectID"      "sampleID"       "invertID"       "lakeID"
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"
##  [9] "depthTop"       "depthBottom"    "supergroup"     "orderSample"
## [13] "family"         "subfamily"      "tribe"          "genus"
## [17] "otu"            "pupa"           "scopeID"        "geneticProcess"
## [21] "geneticID"      "confidence"     "trayNumber"     "invertNum"
## [25] "bodyLength"     "headWidth"      "dryMass"        "picker"
## [29] "comments"       "metadataID"     "updateID"
```

BENTHIC_INVERT_SAMPLES

Corresponding data are in BENTHIC_INVERTS, joined by `sampleID`.

Benthic invert samples from SAMPLES were separated out into their own table in September 2020, in response to some missing and unprocessed samples from the Crazy Chiro survey. Fixing those samples required us to add a new column, `processed`, to indicate whether a sample that had no associated bugs was actually NBC (no bugs caught) or whether it was taken but never processed. Instead of adding `processed` to the general SAMPLES table, we created a new table.

Other than `processed`, the fields in here are pretty standard for a sample table. `metadataID` should describe the *sampling*, not processing, protocol for the bugs in this sample.

Primary key: `sampleID`

Variables:

```
##  [1] "lakeID"         "siteID"         "sampleID"       "dateSample"
##  [5] "dateTimeSample" "depthClass"     "depthTop"       "depthBottom"
##  [9] "crew"           "weather"        "processed"      "comments"
## [13] "metadataID"     "updateID"
```

CHLOROPHYLL

This table contains measurements of chlorophyll concentration in sampled lake water.

There are the usual columns: `projectID` connects to PROJECTS, `sampleID` connects to SAMPLES, `lakeID` connects to LAKES, and `siteName` connects to SITES.

`runID` is given so that sample results can be matched to the standards for a given run on the chlorophyll analysis machine. `chl` is the final measured chlorophyll concentration, in units of ug/L. Two replicates were run for each (or at least, for most) samples, so `replicate` indicates which replicate is being measured. `flag` is 1 for possibly bad data; 0 otherwise. `comments` indicate particular circumstances of the run or any other oddities.

Primary key: `sampleID` + `replicate`

Variables:

```
dbListFields(con, "chlorophyll")
```

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"
##  [9] "depthBottom"    "runID"          "chl"            "replicate"
## [13] "metadataID"     "comments"       "flag"           "updateID"
```

COLOR

Spectrophotometer measurements of lake water.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"
##  [9] "depthBottom"    "abs440"         "g440"           "metadataID"
## [13] "comments"       "flag"           "updateID"
```

CREEL_BOATS

This table contains further details about the boats summarized in aggregate in CREEL_BOAT_SAMPLES. Whereas in that table each row was an instantaneous boat count, each row in this table is a single boat.

`boatcountID` links to CREEL_BOAT_SAMPLES, while `boatID` is the primary key for this table.

`boatNum` combines with `boatCountID` to create `boatID`. `vesselType`, `passengerCount`, `fishing`, `residentRover`, and `movingStationary` give further info about the boat.

Primary key: `boatID`

Variables:

```
dbListFields(con, "creel_boats")
```

```
##  [1] "projectID"        "boatCountID"      "boatID"           "boatNum"
##  [5] "vesselType"       "passengerCount"   "fishing"          "residentRover"
##  [9] "movingStationary" "notes"            "enteredBy"        "metadataID"
## [13] "updateID"
```

CREEL_BOAT_SAMPLES

One row per boat sample. `sampleID`'s here are a subset of those listed in CREEL_SAMPLES, but this table contains further information specific to boat samples only. Each row in this table corresponds to one `boatCountID`. Each unique `boatCountID` represents an instantaneous boat count—multiple instantaneous counts were taken during a single visit to a lake (aka a single sample).

`boatCount` is the number of boats that were counted during the instantaneous count in question.

More details about the methods used for boat sampling can be found in the metadata doc, "fishscapesCreel.Boats.20200512.docx."

Primary key: `boatCountID`

Variables:

```
dbListFields(con, "creel_boat_samples")
```

```
##  [1] "projectID"      "lakeID"         "siteID"         "sampleID"
##  [5] "boatCountID"    "dateSample"     "dateTimeSample" "gear"
##  [9] "enteredBy"      "metadataID"     "updateID"
```

CREEL_FISH

Each row in this table is a single fish, with the primary key being `fishID`. Each `creelID` can have more than one fish (i.e. if an interviewed group of anglers caught more than one fish, which they often did).

Ashley notes that sometimes, the creel survey crew caught fish themselves, but those fish were entered into the FISH_INFO table, not this one.

`fishLength` is recorded for each fish, as well as three ID columns: `species`, `speciesCode`, and `speciesCode_wiDNR`.

`creelID` connects to CREEL_INFO, and `projectID` connects to PROJECTS. `updateID`, of course, connects to UPDATE_METADATA.

Primary key: `fishID`

Variables:

```
dbListFields(con, "creel_fish")
```

```
##  [1] "projectID"         "creelID"           "fishID"
##  [4] "creelNumber"       "fishNumber"        "species"
##  [7] "speciesCode_wiDNR" "speciesCode"       "fishLength"
## [10] "notes"             "enteredBy"         "metadataID"
## [13] "updateID"
```

CREEL_INFO

Table containing survey responses to WI DNR creel survey questions: target fish, info on the expedition, percent effort on certain species, etc. `creelID` is composed of the `sampleID` and the `creelNumber` and is the primary key, with each fishing party getting a separate ID. More info about interview procedures can be found in the associated metadata docs.

`sampleID` joins to CREEL_SAMPLES. If outgoing and return interviews were conducted, their times are recorded in `timeOutgoingInterview` and `timeReturnInterview`, while `outgoingInterview` and `returnInterview` give a binary indication of whether or not these interviews were conducted.

I won't go over the rest of the questions here—the metadata document contains the full interview checklist/data sheet.

Primary key: `creelID`

Variables:

```
dbListFields(con, "creel_info")
```

```
##  [1] "projectID"                       "sampleID"
##  [3] "creelID"                         "creelNumber"
##  [5] "timeOutgoingInterview"           "timeReturnInterview"
##  [7] "outgoingInterview"               "returnInterview"
##  [9] "declinedInterview"               "partyActivity"
## [11] "numberInParty"                   "vehicleUsed"
## [13] "timeLeftLanding"                 "timeNotFishing"
## [15] "licenseType"                     "guidedTrip"
## [17] "completeTrip"                    "tournament"
## [19] "notes"                           "enteredBy"
## [21] "expectedLengthTripHours"         "outTarget1"
## [23] "outTarget1Percent"               "outTarget1Expect"
## [25] "outTarget2"                      "outTarget2Percent"
## [27] "outTarget2Expect"                "outTarget3"
## [29] "outTarget3Percent"               "outTarget3Expect"
## [31] "walleyePercentEffort"            "walleyeNumberCaught"
## [33] "walleyeNumberKept"               "walleyeReleaseCode"
## [35] "muskellungePercentEffort"        "muskellungeNumberCaught"
## [37] "muskellungeNumberKept"           "muskellungeReleaseCode"
## [39] "northernPikePercentEffort"       "northernPikeNumberCaught"
## [41] "northernPikeNumberKept"          "northernPikeReleaseCode"
## [43] "largemouthBassPercentEffort"     "largemouthBassNumberCaught"
## [45] "largemouthBassNumberKept"        "largemouthBassReleaseCode"
## [47] "smallmouthBassPercentEffort"     "smallmouthBassNumberCaught"
## [49] "smallmouthBassNumberKept"        "smallmouthBassReleaseCode"
## [51] "panfishGeneralEffort"            "panfishGeneralNumber"
## [53] "panfishGeneralNumberKept"        "panfishGeneralReleaseCode"
## [55] "bluegillPercentEffort"           "bluegillNumberCaught"
## [57] "bluegillNumberKept"              "bluegillReleaseCode"
## [59] "yellowPerchPercentEffort"        "yellowPerchNumberCaught"
## [61] "yellowPerchNumberKept"           "yellowPerchReleaseCode"
## [63] "blackCrappiePercentEffort"       "blackCrappieNumberCaught"
## [65] "blackCrappieNumberKept"          "blackCrappieReleaseCodes"
## [67] "additionalSpeciesName"           "additionalSpeciesPercentEffort"
## [69] "additionalSpeciesNumberCaught"   "additionalSpeciesNumberKept"
## [71] "additionalSpeciesReleaseCode"    "additionalSpecies2Name"
## [73] "additionalSpecies2PercentEffort" "additionalSpecies2NumberCaught"
## [75] "additionalSpecies2NumberKept"    "additionalSpecies2ReleaseCode"
## [77] "metadataID"                      "updateID"
```

CREEL_INTERVIEW

Responses to additional interview questions designed by the Fishscapes team (in addition to the WI DNR questions in CREEL_INFO). As in that table, the `creelID` is composed from the `sampleID` and `creelNumber`, with `creelID` serving as the primary key for this table. CREEL_INFO is more complete that CREEL_INTERVIEW–that is, all creelID's in CREEL_INTERVIEW also exist in CREEL_INFO, but not the other way around. This is because the WI DNR questions were asked first, with the Rutgers/Fishscapes questions asked afterwards only if the angler consented to a more in-depth interview.

`sampleID` joins to CREEL_SAMPLES, while `creelID` is shared among this table and CREEL_INFO, and further joins to CREEL_FISH. Several other columns contain data for the various questions asked (see more details in the metadata doc).

Primary key: `creelID`

Variables:

```
dbListFields(con, "creel_interview")
```

```
##  [1] "projectID"            "sampleID"             "creelID"
##  [4] "creelNumber"          "timeInterview"        "dataRecorder"
##  [7] "placeStaying"         "locLakeAny"           "locLakeThis"
## [10] "travelTime"           "homeZip"              "sourceFriends"
## [13] "sourceStore"          "sourceDNR"            "sourceForums"
## [16] "sourceApps"           "sourcePersonal"       "sourceOther"
## [19] "sourceOtherName"      "rankFriends"          "rankStore"
## [22] "rankDNR"              "rankForums"           "rankApps"
## [25] "rankPersonal"         "rankOther"            "tellFriendsLastTime"
## [28] "tellStoreLastTime"    "tellDNRLastTime"      "tellForumsLastTime"
## [31] "tellAppsLastTime"     "tellOtherLastTime"    "tellOtherNameLastTime"
## [34] "tellFriends"          "tellStore"            "tellDNR"
## [37] "tellForums"           "tellApp"              "tellOther"
## [40] "tellOtherName"        "fishingDays"          "fishingDaysLakeYear"
## [43] "species1Name"         "fishingDaysSpecies1"  "species2Name"
## [46] "fishingDaysSpecies2"  "species3Name"         "fishingDaysSpecies3"
## [49] "species4Name"         "fishingDaysSpecies4"  "lakesPastWeek"
## [52] "lakesPastMonth"       "lakesPastYear"        "satisfactionOverall"
## [55] "satisfactionNumber"   "satisfactionSize"     "lifeRevolves"
## [58] "trophySpots"          "mostEnjoyable"        "significantIncome"
## [61] "specialGear"          "enjoyable"            "noTimeHobbies"
## [64] "maximizeNumbers"      "noAlternative"        "friendsFish"
## [67] "releaseFish"          "loseFriends"          "bigSmall"
## [70] "otherLeisure"         "keepFishing"          "stocked"
## [73] "knowWhenStocked"       "whenStocked"           "whenDecided"
## [76] "alternativeLakes"      "secondChoice"          "whyThisLake"
## [79] "whyStop"               "metadataID"            "updateID"
```

CREEL_SAMPLES

Each row is a single creel survey done at a single lake on a single date. This table contains all types of creel samples, including trailer, boat, and interview samples. `sampleID` identifies the sample. `sampleGroup` classifies samples by when/for which project they were taken. `effort` is the amount of time spent on the survey, and `effortUnits` describes how that effort is measured.

`useEffort` is a choice of whether the data from that sample should be included in CPUE calculations. It's typically decided in the moment based on whether something disqualifying happened. Like one time our fishing poles broke and only one person fished for the whole time, so the data was weird and probably shouldn't be used in calculations.

Primary key: `sampleID`

Variables:

```
dbListFields(con, "creel_samples")
```

```
##  [1] "siteID"         "sampleID"       "dayOfYear"      "dateSet"
##  [5] "dateSample"     "dateTimeSet"    "dateTimeSample" "crew"
##  [9] "gear"           "sampleGroup"    "effort"         "effortUnits"
## [13] "useEffort"      "weather"        "comments"       "metadataID"
## [17] "updateID"
```

CREEL_TRAILERS

This table contains further details about the trailers summarized in aggregate in CREEL_TRAILER_SAMPLES. Whereas in that table each row was an instantaneous trailer count, in this table, each row is a single trailer.

`trailerCountID` links to CREEL_TRAILER_SAMPLES, while `trailerID` is the primary key for this table (although it has some duplicates right now—see GH #103.

`wisconsinPlates` is self-explanatory and applies to each individual trailer.

Primary key: `trailerID`

Variables:

```
dbListFields(con, "creel_trailers")
```

```
## [1] "projectID"       "trailerCountID"  "trailerID"       "trailerNum"
## [5] "wisconsinPlates" "metadataID"      "updateID"
```

CREEL_TRAILER_SAMPLES

One row per trailer sample. `sampleID`'s here are a subset of those listed in CREEL_SAMPLES, but this table contains further information specific to trailer samples only. Each row in this table corresponds to one `trailerCountID`. Each unique `trailerCountID` represents an instantaneous trailer count—multiple instantaneous counts were taken during a single visit to a lake (aka a single sample). From the metadata doc: "Instantaneous counts were completed when arriving at the field site, when departing, and on every hour in between. In an 8-hour shift (from 5:30 to 13:30 or from 13:30 to 21:30), ten instantaneous vehicle counts were recorded."

`trailerCount` is the number of trailers present during that instantaneous count. `vehiclePresent` is a binary version of `trailerCount`: 1 if any vehicles were present and 0 if no vehicles were present.

The weather columns denote severe weather and are binary.

Primary key: `trailerCountID`

Variables:

```
dbListFields(con, "creel_trailer_samples")
```

```
##  [1] "projectID"      "lakeID"         "siteID"         "sampleID"      
##  [5] "trailerCountID" "dateSample"     "dateTimeSample" "gear"          
##  [9] "trailerCount"   "vehiclePresent" "heavyWind"      "heavyRain"     
## [13] "heavyClouds"    "notes"          "enteredBy"      "metadataID"    
## [17] "updateID"
```

CREW

CREW is in long format, after some changes made by Kaija in summer 2020. Each year is a unique combination of person, year, role, and location. There is no primary key. Kaija does not know how she feels about this reformat: she was all for it initially, but now she has doubts. It's a bit confusing. Unfortunately, for this table to be really normalized, we'd have to have a whole separate "people" table, give each person an ID, and then match the ID's to locations and years. We've basically decided that that's more effort than we care to invest in the CREW table, since it's pretty much never used for analyses.

`ID` is unique to each year/semester/location combo. It's a holdover from the wide format of the table.

Note: this table does not have a primary key.

Variables:

```
##  [1] "ID"          "year"        "semester"    "location"    "role"       
##  [6] "firstname"   "lastname"    "nickname"    "affiliation" "notes"      
## [11] "updateID"
```

DRY_MASS_EQUATIONS

This table contains equations, sourced from literature, to calculate the dry mass of invertebrates and other food items. These are used primarily in FISH_DIETS.

`OTU` corresponds to the `otu` column in the OTU table (confusing, I know) and represents the most specific taxonomic ranking available for the organism in question. `headWidth_a` and `bodyLength_a` are coefficients represented by "a" in the `equation` column—each organism should only have `headWidth_a` *or* `bodyLength_a`. Same goes for `headWidth_b` and `bodyLength_b`. `reference` gives the literature reference for the equation, and `referenceEquation` refers to the number that refers to the equation *in its paper of origin.*

Primary key: OTU

Variables:

```
##  [1] "OTU"                "headWidth_a"        "headWidth_b"       
##  [4] "bodyLength_a"       "bodyLength_b"       "bodyLength_b1"     
##  [7] "equation"           "reference"          "referenceEquation" 
## [10] "comments"           "metadataID"         "updateID"
```

FISH_DIETS

Measurements, ID, and mass of fish diet items.

Note that in 2011-2016, there are often multiple rows per fish per dietItem, because Brian Weidel wanted to enter the data as specifically as possible while still averaging diet items of the same size. This procedure is explained in more detail in the fish diets metadata document, and it is the reason that we've included `entryOrder` as part of the composite primary key. After 2016, the combination of `fishID` and `dietItem` is a unique identifier.

Primary key: `fishID` * `dietItem` * `entryOrder`

Variables:

```
##  [1] "fishID"              "entryOrder"          "lakeID"             
##  [4] "dateSample"          "species"             "dietItem"           
##  [7] "dietItemCount"       "dietItemBodyLength"  "dietItemHeadWidth"  
## [10] "otherLength"         "dietItemRangeLower"  "dietItemRangeHigher"
## [13] "dryMass_bodylength"  "dryMass_headwidth"   "dryMass_other"      
## [16] "totalDryMass"        "dietProcessor"       "comments"           
## [19] "metadataID"          "updateID"
```

FISH_INFO

Corresponding samples are in FISH_SAMPLES, joined by `sampleID`.

This is a behemoth of a table. So many columns! Each row is a fish. Different columns apply to different projects/fish/methods, so most of them are NA for any given individual fish.

`species` joins to the `otu` column in OTU. When there were multiple anglers in a boat, caughtBy gives the initials of the angler who caught that fish. (See Fishscapes.Angling.20180625.docx for a more detailed description of how to calculate per-angler CPUE for Fishscapes samples).

New fixes are coming soon to the tag* and clip* columns, so I won't describe those in detail for now.

Primary key: `fishID`

Variables:

```
##  [1] "projectID"               "sampleID"                "fishID"                 
##  [4] "fishNum"                 "species"                 "fishLength"             
##  [7] "standardLength"          "fishWeight"              "caughtBy"               
## [10] "jumperDescription"       "useTagMarkRecap"         "clipApply"              
## [13] "clipRecapture"           "floyApply"               "floyRecapture"          
## [16] "pitApply"                "pitRecapture"            "sex"                    
## [19] "mortality"               "removed"                 "otolithSampled"         
## [22] "tissueSampled"           "dietSampled"             "stomachRemoved"         
## [25] "gillArchRemoved"         "pectoralFinRemoved"      "gonadRemoved"           
## [28] "leftEyeRemoved"          "finClipCollected"        "photo"                  
## [31] "gonadWeight"             "rectalTemp"              "gonadSqueeze"           
## [34] "sexualStage_MaierScale" "gpsWaypoint"             "finClipBox"             
## [37] "spineSample"             "scaleSample"             "comments"               
## [40] "metadataID"              "updateID"
```

FISH_MORPHOMETRICS

This table was created to contain bluegill morphometric measurements from Chelsea Bishop's 2020 dissertation. The github repo with the code for this project is here.

`fishID` connects to FISH_INFO. `imageFile` refers to images stored in Chelsea's photo folder on Box.

The table is in long format, with one row per parameter (i.e. many rows for each fish). This table contains both conventional and geometric morphometric measurements, so some of the `parameter` values are lengths, while others are X/Y coordinates for morphometric landmarks. For these, numbers indicate correspondence (e.g. X1 goes with Y1). `parameterValue` contains the value of the parameter, and `parameterUnit` gives units for parameters that have them. Landmarks measured in x and y coordinates are unitless.

`replicate` differentiates the same landmarks measured more than once for the same fish: this happened because Chelsea measured several fish more than once in order to estimate her measurement error.

More details about the measurement protocols can be found in the metadata document, “blgMorphologyDOC.20180626.docx”.

Primary key: `fishID` * `parameter` *`replicate`

Variables:

```
## [1] "fishID"         "imageFile"      "parameter"      "parameterValue"
## [5] "parameterUnit"  "processedBy"    "replicate"      "metadataID"
## [9] "updateID"
```

FISH_OTOLITHS

Otolith ID and measurements. This table is in need of restructuring and checking; see GitHub issue #134.

Variables:

```
##  [1] "fishID"                 "wellNumber"             "otolith"
##  [4] "parameter"              "year"                   "annulusNumber"
##  [7] "paramValue"             "unit"                   "interpreter"
## [10] "interpretationNumber"   "interpretationDateTime" "metadataID"
## [13] "confidence"             "comments"               "updateID"
```

FISH_SAMPLES

Corresponding data are in FISH_INFO, joined by `sampleID`.

Unlike SAMPLES, this table includes both `dateSet`/`dateTimeSet` and `dateSample`/`dateTimeSample`. This is to account for minnow traps, which are deployed over a certain period of time.

`crew` gives the people on the boat when the samples were taken. `gear` gives the type of fishing gear used. `effortUnits` gives the units of `effort`, which may be as simple as “hours” or something like “angler_hours” (roughly number of anglers * number of hours). `useCPUE` tells whether CPUE (catch per unit effort) is an appropriate metric for the given sample.

Primary key: `sampleID`

Variables:

```
##  [1] "lakeID"              "siteID"              "sampleID"
##  [4] "dayOfYear"           "dateSet"             "dateSample"
##  [7] "dateTimeSet"         "dateTimeSample"      "crew"
## [10] "gear"                "sampleGroup"         "effort"
## [13] "effortUnits"         "distanceShocked"     "useCPUE"
## [16] "useSampleMarkRecap" "comments"            "metadataID"
## [19] "updateID"
```

FISH_YOY

Corresponding samples are in FISH_SAMPLES, joined by `sampleID`.

FISH_YOY is analogous to FISH_INFO, but specifically for the Largemouth Bass YOY survey conducted in 2016-2017 (project 29). Data from this survey are kept separate from other fish data because the survey was conducted by swimming transects. Fish were observed visually or through a GoPro, and their length was estimated, but they were not held in hand. Therefore, we don’t have length, weight, etc. measurements for them.

`YOY` is binary: was the fish a YOY or not. `fishLength` gives the lower bound for the fish’s length; `fishLengthMax` gives the upper bound.

Note that there are often breaks in the sequences of fish numbers for a given sampleID. You should not assume that because the highest `fishNum` is e.g. 400 that there were 400 fish in the sample. Numbering was somewhat haphazard, with an eye mainly toward not repeating numbers. Missing numbers do not indicate missing fish.

Primary key: `fishID`

Variables:

```
##  [1] "projectID"     "siteID"        "sampleID"      "fishID"
##  [5] "fishNum"       "species"       "YOY"           "plus1"
##  [9] "fishLengthMin" "fishLengthMax" "comments"      "metadataID"
## [13] "updateID"
```

FLIGHTS

During the Fishscapes project, some data was collected through airplane flights above lakes. FLIGHTS contains flight-level information, encompassing many samples (one sample per lake)–information on each sample can be found in FLIGHTS_SAMPLES.

Each row of this table corresponds to a single flight. `flightID` identifies each flight (only three so far). `boatCount` gives the total number of boats counted across all of the lakes counted in `lakeCount`. `crew` tells who manned the flight.

More information about flight protocols can be found in the flights metadata document, "fishscapesCreel.Flight.20200410.docx".

Primary key: `flightID`

Variables:

```
##  [1] "projectID"      "flightID"       "dateSample"     "dateTimeSample"
##  [5] "boatCount"      "lakeCount"      "crew"           "comments"
##  [9] "metadataID"     "updateID"
```

FLIGHTS_INFO

One row per boat, so several rows per sampleID (or 0 rows per sampleID if no boats were found on that lake in that sample). You can think of this table as analogous to FISH_INFO, while FLIGHTS_SAMPLES is analogous to FISH_SAMPLES.

`sampleID` links to FLIGHTS_SAMPLES. `boatID` is unique to each boat and is composed of `sampleID` and `boatNum`. `fishing` and `moving` are binary, and `vesselType` identifies the type of boat.

Primary key: `boatID`

Variables:

```
## [1] "projectID"  "sampleID"   "boatID"     "boatNum"    "vesselType"
## [6] "fishing"    "moving"     "updateID"
```

FLIGHTS_SAMPLES

In contrast to FLIGHTS, where each row is one flight over many lakes, FLIGHTS_SAMPLES gives one row to each flight/lake combination.

`flightID` links to FLIGHTS. `siteID` links to SITES. lakeID links to LAKES. `boatCount` is the number of boats counted on that lake from the flight.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"      "lakeID"         "siteID"         "flightID"      
##  [5] "sampleID"       "dateSample"     "dateTimeSample" "gear"          
##  [9] "boatCount"      "metadataID"     "updateID"
```

GC

Gas chromatography. This table is in need of restructuring and checking; see GitHub issue #118.

Variables:

```
##  [1] "projectID"          "sampleID"           "lakeID"            
##  [4] "siteName"           "dateSample"         "dateTimeSample"    
##  [7] "depthClass"         "depthTop"           "depthBottom"       
## [10] "subsampleClass"     "subsampleDateTime" "runID"             
## [13] "runDate"            "replicate"          "metadataID"        
## [16] "runName"            "comments"           "CH4PeakArea"       
## [19] "CO2PeakArea"        "CH4ppm"             "CO2ppm"            
## [22] "CH4_uM"             "CO2_uM"             "updateID"
```

ISOTOPE_BATCHES

Stable isotope analysis batch information: processing lab, date sent, date received, etc.

Primary key: batchID

Variables:

```
## [1] "batchID"           "IsotopeLab"        "dateSent"         
## [4] "dateReceived"      "sampleDescription" "instrument"       
## [7] "CEST_User"         "updateID"
```

ISOTOPE_RESULTS

Results of stable isotope analysis of various types of samples. isotopeID joins to the various ISOTOPE_SAMPLES_* tables, depending on the letter in the isotopeID. This table, and the associated samples tables, are in need of cleanup; see GitHub issue #133.

Primary key: isotopeID * batchID * replicate

Variables:

```
##  [1] "isotopeID"  "sampleWt"   "d13C"       "d15N"       "d2H"       
##  [6] "d18O"       "percentC"   "percentN"   "percentH"   "dateRun"   
## [11] "lab"        "batchID"    "replicate"  "metadataID" "comments"  
## [16] "flag"       "updateID"
```

ISOTOPE_SAMPLES_BENTHIC_INVERTS

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: isotopeID

Variables:

```
##  [1] "projectID"      "isotopeID"      "sampleID"       "lakeID"        
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"    
##  [9] "depthTop"       "depthBottom"    "taxa"           "batchID"       
## [13] "comments"       "updateID"
```

ISOTOPE_SAMPLES_DIC

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "lakeID"        
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"    
##  [9] "depthTop"       "depthBottom"    "batchID"        "comments"      
## [13] "updateID"
```

ISOTOPE_SAMPLES_FISH

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
## [1] "projectID" "isotopeID" "fishID"    "lakeID"    "species"   "batchID"  
## [7] "comments"  "updateID"
```

ISOTOPE_SAMPLES_METHANE

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "dateSample"    
##  [5] "dateTimeSample" "lakeID"         "siteName"       "depthClass"    
##  [9] "depthTop"       "depthBottom"    "batchID"        "comments"      
## [13] "updateID"
```

ISOTOPE_SAMPLES_PERIPHYTON

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "lakeID"        
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"    
##  [9] "depthTop"       "depthBottom"    "source"         "batchID"       
## [13] "comments"       "metadataID"     "updateID"
```

ISOTOPE_SAMPLES_POC

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "lakeID"
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"
##  [9] "depthTop"       "depthBottom"    "filterVol"      "sampleType"
## [13] "sampleAmount"   "batchID"        "comments"       "updateID"
```

ISOTOPE_SAMPLES_WATER

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "lakeID"
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"
##  [9] "depthTop"       "depthBottom"    "batchID"        "comments"
## [13] "updateID"
```

ISOTOPE_SAMPLES_ZOOPS

Samples taken for stable isotope analysis. Results can be found in ISOTOPE_RESULTS. This table is in need of cleanup; see GitHub issue #133.

Primary key: `isotopeID`

Variables:

```
##  [1] "projectID"      "sampleID"       "isotopeID"      "lakeID"
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"
##  [9] "depthTop"       "depthBottom"    "batchID"        "comments"
## [13] "updateID"
```

LAKES

Contains general information about lakes used for various studies in the database. `lakeID` is a unique code for each lake (usually 2 letters, but sometimes more), used in sampleID's. `lakeName` is, obviously, the name of the lake. `state`, `county`, and `city` give information for finding the lake on the map. `surfaceArea` and `maxDepth` are pretty self-explanatory.

`lat` and `long` are, ideally, the lat/long coordinates for the center of the lake. The gold standard for the lat/long coordinates is the ones shown in the lake finder app, but some lakes are hard or impossible to find using the app. If the lake can't be found using the app, the second choice is coordinates obtained by clicking the center of the lake on Google Maps. Last resort is coordinates of a non-central point in the lake, like the boat launch or some random part of the lake's area. `latLongSource` indicates where the lat/long coordinates come from.

`WBIC` is the Wisconsin DNR Water Body ID Code, which you can use to look up a lake here.

Primary key: `lakeID`

Variables:

```
##  [1] "lakeID"        "lakeName"      "state"         "county"
##  [5] "city"          "surfaceArea"   "maxDepth"      "lat"
##  [9] "long"          "latLongSource" "WBIC"          "comments"
## [13] "updateID"
```

LAKES_GIS

Watershed info for study lakes. `lakeID` is a foreign key for [`LAKES`]`$lakeID`.

Primary key: `lakeID`

Variables:

```
##  [1] "lakeID"                      "watershedID"
##  [3] "water"                       "urban_openSpace"
##  [5] "urban_lowintensity"          "urban_mediumintensity"
##  [7] "urban_highintensity"         "barrenLand"
##  [9] "deciduousForest"             "evergreenForest"
## [11] "mixedForest"                 "shrub_scrub"
## [13] "grassland_herbaceous"        "pasture_hay"
## [15] "cultivatedCrops"             "woodyWetlands"
## [17] "emergentHerbaceousWetlands" "totalArea"
## [19] "updateID"
```

LAKE_BATHYMETRY

Bathymetric profiles of study lakes. `lakeID` is a foreign key for [`LAKES`]`$lakeID`.

Primary key: `lakeID` * `depth_m`

Variables:

```
##  [1] "lakeID"              "lakeName"             "depth_m"
##  [4] "area_m2"             "sedimentArea3D_m2"    "layerVolume_m3"
##  [7] "volumeToBottom_m3"   "volumeToBottom3D_m3" "source"
## [10] "metadataID"          "updateID"
```

LIMNO_PROFILES

DO, pH, temp, etc. at descending depths.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"
##  [9] "depthBottom"    "temp"           "DOmgL"          "DOsat"
## [13] "SpC"            "pH"             "ORP"            "PAR"
## [17] "metadataID"     "comments"       "updateID"
```

LIPID_EXTRACTIONS

Results of lipid extractions from the samples in LIPID_SAMPLES. `lipidID` is a foreign key for [`LIPID_SAMPLES`]`$lipidID`.

Primary key: `extractionID`

Variables:

```
## [1] "extractionID"      "extractionDate"    "lipidID"
## [4] "extractionWt_mg"   "lipidWt_mg"        "metadataID"
## [7] "totalFilterVol_mL" "comments"          "updateID"
```

LIPID_SAMPLES

Sampling and processing info for lipid samples from zoops, inverts, etc.

Primary key: `lipidID`

Variables:

```
##  [1] "projectID"       "sampleID"        "lipidID"         "lakeID"         
##  [5] "siteName"        "dateSample"      "depthClass"      "depthTop"       
##  [9] "depthBottom"     "sampleType"      "taxa"            "filterVolume_mL"
## [13] "comments"        "updateID"
```

METADATA

METADATA contains metadataID's and corresponding descriptions, or information about how to find extended descriptions if they are saved in separate metadata files. (Almost) every table has a `metadataID` column, which joins to this table.

`dateCreated` is supposed to be the date the file was created *and* the date that the entry was added to the METADATA table, although in practice these are not usually the same date. Consider revising/clarifying philosophy on this.

`metadata` is a brief metadata description. This field is especially important if there is no corresponding metadata document, e.g. if the metadata is pretty short and can easily just be contained in this table. Some metadataID's with documents also have a description in this field; if there's a document, then it's optional.

`file` gives the file name of the external metadata document corresponding to each `metadataID`, if it has one.

IMPORTANT NOTE: The file name does not have to match the metadataID!!! This trips people up all the time, and it tripped me (Kaija) up immensely when I started working with this database. The idea is that the metadataID should (almost) never have to change, but if you need to change the name of a document, that's no problem.

This can get confusing because *often*, the metadataID *does* line up with the name of the document. For example, the metadataID "hyperstability.20180521" corresponds to the document "hyperstability.20180521.docx". Logical! But you can't assume that will always be the case. So don't be like me and find a metadataID in some table, go into Box, search for a file called [thatmetadataID].docx, and then panic when you can't find the document. Instead, you have to do the following:

Find a metadataID that you want to get info about —> open the METADATA table —> find your metadataID in METADATA —> find the file name in the `file` column —> go find that file in Box or wherever it is.

"But how do I find the file?" you ask. Aha! That's where the `filePath` column comes in. It tells you where the file is located. Most of them live in "Box/MFE/Database/Metadata Files", but there's an occasional exception.

Primary key: `metadataID`

Variables:

```
## [1] "metadataID"  "dateCreated" "metadata"    "file"        "filePath"   
## [6] "updateID"
```

MOLECULAR_SAMPLE

Samples to be used for DNA/RNA extraction.

Primary key: `molecularID`

Variables:

```
##  [1] "projectID"       "sampleID"        "molecularID"     "siteName"
##  [5] "dateSample"      "dateTimeSample"  "depthClass"      "depthTop"
##  [9] "depthBottom"     "sampleType"      "filterVol"       "boxNum"
## [13] "cellNum"         "remainingSample" "metadataID"      "comments"
## [17] "updateID"
```

OTU

The OTU table is designed to be a general-purpose reference for taxonomy across the database. There are several places in the database where we might identify an organism to family/species/etc, and that identification should be joinable to an entry in OTU (by the "otu" column), which would then give us the higher-level taxonomic information for that organism.

Note that the values in the "otu" column of OTU are pretty diverse. Some of them include life stage information (e.g. "coleoptera_larvae"), and some of them include habitat/not-strictly-taxonomic information such as "coleoptera_terrestrial". Different info like that is relevant to different database tables. For instance, in a fish diet it might be helpful to distinguish between aquatic and terrestrial coleoptera, while in the BENTHIC_INVERTS table, coleoptera would be assumed to be larvae. We've deliberately left the "otu" column as a mishmosh of that sort of designation, adding rows when needed instead of replacing them, so that it can serve as a central location for matching common names and specific classifications to higher-level taxonomic information.

For fish species, the common name generally goes in the "otu" slot. Fish species that have a common abbreviation, such as LMB for largemouth bass, have that noted in the "abbreviation" slot. Abbreviations should conform to the WI DNR fish abbreviations.

Primary key: `otu`

Variables:

```
dbListFields(con, "otu")
```

```
##  [1] "habitat"      "grouping"     "supergroup"   "orderTax"     "family"
##  [6] "subfamily"    "tribe"        "genus"        "species"      "commonName"
## [11] "abbreviation" "otu"          "updateID"
```

PIEZOMETERS_INSTALL

Depth, site, and other specs on installation.

Primary key: `siteID`

Variables:

```
##  [1] "siteID"             "installDate"        "removalDate"
##  [4] "piezLength_m"       "piezDiameter_m"     "heightAboveSeds_m"
##  [7] "insertionDepth_m"   "waterDepth_m"       "screenHeight_m"
## [10] "screenLength_m"     "screenInterval_m"   "conductivity_m_day"
## [13] "metadataID"         "comments"           "updateID"
```

PIEZOMETERS_LAKE

In-lake piezometer readings.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"            "sampleID"             "lakeID"
##  [4] "siteName"             "dateSample"           "dateTimeSample"
##  [7] "depthClass"           "depthTop"             "depthBottom"
## [10] "lakeLevel_m"          "wellLevel_m"          "pumped"
## [13] "wellLevelCorrected_m" "hydraulicHead_m"      "metadataID"
## [16] "comments"             "flag"                 "updateID"
```

PIEZOMETERS_SENSORS

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"               "sampleID"
##  [3] "lakeID"                  "siteName"
##  [5] "dateSample"              "dateTimeSample"
##  [7] "depthClass"              "depthTop"
##  [9] "depthBottom"             "wellLevel_m"
## [11] "wellLevelCorrected_m"    "chainLength_m"
## [13] "eyeBoltBelowTopOfPiez_m" "sensorDepth_m"
## [15] "metadataID"              "comments"
## [17] "updateID"
```

PIEZOMETERS_SURVEYING

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"                 "sampleID"
##  [3] "lakeID"                    "siteName"
##  [5] "surveyTrackID"             "dateSample"
##  [7] "dateTimeSample"            "depthClass"
##  [9] "depthTop"                  "depthBottom"
## [11] "topOfPiezAboveLakeLevel_m" "metadataID"
## [13] "comments"                  "updateID"
```

PIEZOMETERS_UPLAND

Upland piezometer readings.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"               "sampleID"
##  [3] "lakeID"                  "siteName"
##  [5] "dateSample"              "dateTimeSample"
##  [7] "depthClass"              "depthTop"
##  [9] "depthBottom"             "wellLevel_m"
## [11] "pumped"                  "wellLevelCorrected_m"
## [13] "wellHeightAboveGround_m" "waterTable_m"
## [15] "metadataID"              "comments"
## [17] "updateID"
```

PRIMARY_PRODUCTION_BENTHIC

Lab incubations to measure benthic primary production (GPP, NPP).

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"                  "sampleID"
##  [3] "lakeID"                     "siteName"
##  [5] "dateSample"                 "dateTimeSample"
##  [7] "depthClass"                 "depthTop"
##  [9] "depthBottom"                "benthicRespiration_mgC_m2_h"
## [11] "benthicNPP_mgC_m2_h"        "benthicGPP_mgC_m2_h"
## [13] "incubationDuration_h"       "metadataID"
## [15] "comments"                   "updateID"
```

PROJECTS

Contains information about projects. Joined to most other database tables by `projectID`. A "project" could be a dissertation or thesis project, a special survey, a routine survey, a particular grant, etc.

Most of the fields here are pretty straightforward. `projectLead` is the person who was in charge of the project; may be a PI or a student/tech/someone else. `responsiblePI` is the PI that the projectLead reported to, ultimately. This helps to distinguish e.g. whether the project was ultimately a Jones lab project, or a Cary project, or something else. This shouldn't be taken as any super-meaningful gospel: there are plenty of collaborative projects where e.g. both Stuart and Chris were heavily involved, and we just kinda picked one of the names to put into the `responsiblePI` field.

`startYear` is the year the project started. We don't have an `endYear` column because we knew we would forget to update it, or it would be imprecise. But at least `startYear` can help contextualize when you're likely to find data for that project.

Soon, there will be more detailed project descriptions in the MFE Yearbook document.

Primary key: `projectID`

Variables:

```
## [1] "projectID"     "projectName"   "description"   "projectLead"
## [5] "responsiblePI" "startYear"     "updateID"
```

PUBLICATIONS_PRESENTATIONS

Contains information about publications and/or presentations made by members of the group. This table is old and may be modified or removed in the future.

This table does not have a primary key.

Variables:

```
## [1] "date"             "leadAuthor"       "journal_location" "title"
## [5] "type"             "citation"         "updateID"
```

RHODAMINE

Rhodamine measurements (after releases, recorded in RHODAMINE_RELEASE).

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"
##  [9] "depthBottom"    "rhodID"         "ppb"            "metadataID"
## [13] "comments"       "updateID"
```

RHODAMINE_RELEASE

Release metadata: date, crew, volume released, etc.

Primary key: `rhodID`

Variables:

```
##  [1] "projectID"         "siteID"            "lakeID"           
##  [4] "siteName"          "rhodID"            "rhodReleaseDate"  
##  [7] "depthClass"        "depthTop"          "depthBottom"      
## [10] "crew"              "rhodReleaseType"   "rhodReleaseVolume"
## [13] "metadataID"        "comments"          "updateID"
```

SAMPLES

Corresponding data are in many different tables.

This is the default table for sample information. Samples go here unless they need to go somewhere else. Most of the samples here are for water stuff as opposed to fish, bugs, etc., but that's a very rough characterization.

`siteID` is a combination of the 2-letter lake ID and the site name within the lake. `dateTimeSample` is the same as `dateSample` but with the addition of the time. More information about the specific columns, and about the composition of `sampleID` strings, can be found in the "Database rules and vocabulary" document.

`metadataID`'s in SAMPLES should refer to the metadata that describes the *sampling* protocol, rather than any sample processing protocols.

Primary key: `sampleID`

Variables:

```
##  [1] "lakeID"         "siteID"         "sampleID"       "dateSample"    
##  [5] "dateTimeSample" "depthClass"     "depthTop"       "depthBottom"   
##  [9] "crew"           "weather"        "comments"       "metadataID"    
## [13] "updateID"
```

SEDIMENT

Mass and organic matter content.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"      
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"      
##  [9] "depthBottom"    "wetMass"        "dryMass"        "ashedMass"     
## [13] "percentOrganic" "metadataID"     "comments"       "updateID"
```

SED_TRAP_DATA

Data from sediment trap surveys. Corresponding samples are stored in SED_TRAP_SAMPLES.

This table contains results of nutrient analysis (P, C, N) for samples collected in sediment traps. As of its creation, it contains data from 2013-2018. It is joined to SED_TRAP_SAMPLES by `sampleID`.

`parameterValue` is in mg (for each of the three parameters). Trap internal diameter is 10.3 cm.

Variables:

```
##  [1] "projectID"      "sampleID"       "totalVolmL"     "volFilteredmL" 
##  [5] "parameter"      "parameterValue" "flag"           "comments"
##  [9] "metadataID"     "updateID"
```

SED_TRAP_SAMPLES

Samples from sediment trap surveys. Corresponding data are stored in SED_TRAP_DATA.

This table contains information about deployment and sampling of sediment traps. Like minnow traps, sediment traps are deployed and then retrieved some time later. As a result, we had to include `dateSet/dateTimeSet` as well as `dateSample/dateTimeSample`. Just like with FISH_SAMPLES, this means that we needed a new table and couldn't include these more generically in SAMPLES.

Raw versions of the data included a trapID column in addition to the siteID, but because sediment traps were always deployed at/near the DeepHole, we've opted here to make the trapID into the `siteName`, e.g. "Sed1".

Primary key: `sampleID` + `parameter`

Variables:

```
##  [1] "lakeID"         "projectID"      "siteName"       "sampleID"
##  [5] "dateTimeSet"    "dateTimeSample" "depthClass"     "depthTop"
##  [9] "depthBottom"    "comments"       "metadataID"     "updateID"
```

SITES

This is one of the central database tables, with nearly every other table referring back to it either explicitly (with a `siteID` or `siteName` column) or implicitly (through the `sampleID` column).

Primary key: `siteID`

Variables:

```
## [1] "siteID"   "lakeID"   "siteName" "lat"      "long"     "UTM"      "updateID"
```

STAFF_GAUGES

Instantaneous staff gauge readings.

Primary key: `sampleID`

Variables:

```
##  [1] "projectID"        "sampleID"         "lakeID"           "siteName"
##  [5] "dateSample"       "dateTimeSample"   "depthClass"       "depthTop"
##  [9] "depthBottom"      "waterHeight"      "waterHeightUnits" "waterHeight_m"
## [13] "metadataID"       "comments"         "updateID"
```

TPOC_DEPOSITION

Atmospheric carbon deposition via collectors on the lake surface.

Variables:

```
##  [1] "projectID"                "sampleID"
##  [3] "lakeID"                   "siteName"
##  [5] "dateSample"               "dateTimeSample"
##  [7] "depthClass"               "depthTop"
##  [9] "depthBottom"              "tPOCdepGreater35_mgC_m2_d"
## [11] "tPOCdepLess35_mgC_m2_d"   "incubationDuration_h"
## [13] "metadataID"               "comments"
## [15] "updateID"
```

UNITS

Information on the units (where applicable) and descriptions for all of the variables in all tables. The `description` field is similar to some of my text descriptions in this document (though with a less narrative structure). Also includes `units` for numeric columns, like mg, ug, etc.

`tableName` gives the name of the table, and `colName` gives the name of the variable/column. So, each *column* in a database table has its own *row* in UNITS.

Tables are generally listed in alphabetical order, though that has broken down a little bit with recent tables. I'll redo the alphabetizing at some point; it doesn't strike me as a priority. Similarly, columns are generally listed in the order they appear in the corresponding database tables, and I've tried to preserve that order when adding rows by adding them in the right place. But please don't take that as gospel—some of the rows may be out of order, and you should check before using them programmatically.

Primary key: `tableName` * `colName`

Variables:

```
## [1] "tableName"   "colName"     "units"       "description" "updateID"
```

UPDATE_METADATA

This table will save you when you're hunting through old database updates trying to figure out why the heck Kaija changed that one thing back in 2020. Here's how it works:

The most important field is the `updateID`. This is joined to every (I think?) other table in the database. Each row of every other table has an updateID that refers to a row in this table. This table will tell you when the update happened (`updateDate`) and what the change was (`updateMetadata`), e.g. how many rows were added or changed, why the change was made, etc.

`rawFile` is the name of the raw csv file containing the data immediately after the change was made. `parentBoxFolder` tells you where, usually within "rawFiles_csv", the csv file is stored. Most of my raw csv files are in folders corresponding to their database updates within "GitHubIssues_Fix".

`sourceCode` gives the name of the script used to make the change in question. `verNumber` notes which database version this update corresponds to, joining to VERSION_HISTORY.

Note: I had trouble parsing the name of this table at first. It's *not* a table devoted to information about updates to the METADATA table (although updates to METADATA will end up having a row here, because like any other updates, they have associated updateIDs!). Think of it as a table for metadata *about* updates.

Variables:

```
## [1] "updateID"        "updateDate"      "verNumber"       "rawFile"
## [5] "parentBoxFolder" "sourceCode"      "updaterInitials" "updateMetadata"
```

VERSION_HISTORY

This is the highest-level meta table and the only one whose updates aren't tracked in UPDATE_METADATA. Each time someone makes a database update, they add a row to VERSION_HISTORY, and a brief description of what went on in the update.

`dbDateCreate` is, obviously, the date when the database was created. `verNumber` is the version number. Version numbers have three decimal places, e.g. 3.5.0. The first number changes each year. The second number changes with most updates within the year. For example, I just pushed v 3.5.0, which included the addition of SED_TRAP_SAMPLES, SED_TRAP_DATA, and BENTHIC_INVERT_SAMPLES, as well as minor changes to a handful of other tables. A collection of fixes like that constitutes a relatively major update, so I opted for 3.5.0 instead of 3.4.1. The third number differentiates relatively minor fixes and isn't used very often. For example, at the time of writing this, I've just heard from Randi that she has some data that needs to be added to the database, but just a couple days ago I pushed a database update. If that data urgently needs to be added and can't wait for the next update, I will add it and create database version 3.5.1, because only one or two tables will be affected and it's just a single fix, probably with one updateID.

`updater` shows who performed the database update. "AR" is Alex Ross, "KS" is Katie Saunders, "RN" is Randi Notte, and "KG" is Kaija Gahm.

`dbTitle` is a just-for-fun name for each database version. It's not really necessary, since there's a new name for every version number, regardless of the size of the update. But we generally have a new theme for the names each year. 2019-2020 is "passive-aggressive email subject lines" and features such gems as "Gas is Out", "Gas is Back On", and "Speed Limits Semicolon Swan".

`status` is pretty understandable. "current" is the current version; other versions are "archived".

`updateInfo` gives a *general* overview of the updates that were performed. I've tried (and failed) to keep these pretty short. This field shouldn't be counted on as a detailed overview of all the fixes performed in that update—check UPDATE_METADATA for that.

Primary key: `verNumber`

Variables:

```
## [1] "dbDateCreate" "verNumber"    "updater"      "dbTitle"      "status"      
## [6] "updateInfo"
```

WATER_CHEM

WATER_CHEM contains combined data from what used to be the NUTRIENTS, DOC, and POC tables. The table is in long format, with one row per parameter*sample. The `parameter` column indicates which parameter is being measured, and `parameterValue` holds its value, in units of ug/L (note that this is a change for some nutrients from their previous tables, which stored measurements in mg/L).

The old DOC table was even more of a "long" table, in the sense that each row was a single replicate, so there were ~2 rows per sample (usually 2 replicates taken). As of December 2020, KG has written a DOC quality control script that takes averages of those DOC replicates before putting them into WATER_CHEM. The resulting `parameterValues` for DOC, then, are averages, with some exceptions for singleton samples.

After the quality control process, DOC samples are given a `QCcode` that describes whether the averaged value is within the error bounds of past samples from the same lake and site, and whether the two replicates that created that average were abnormally far apart. Translations of the QCcodes can be found in the metadata document "DOC_QCcodes.pdf". `QCcode` is NA for `parameter`s other than DOC.

Variables:

```
dbListFields(con, "water_chem")
```

```
##  [1] "projectID"      "lakeID"         "sampleID"       "dateSample"    
##  [5] "parameter"      "parameterValue" "QCcode"         "flag"          
##  [9] "comments"       "metadataID"     "updateID"
```

ZOOPS_ABUND_BIOMASS

Zooplankton samples, counts, biomass.

Variables:

```
##  [1] "projectID"           "sampleID"            "lakeID"             
##  [4] "siteName"            "dateSample"          "dateTimeSample"     
##  [7] "depthClass"          "depthTop"            "depthBottom"        
## [10] "taxa"                "count"               "meanMass_ug"        
## [13] "abundance_num_m3"    "biomass_gDryMass_m3" "metadataID"         
## [16] "comments"            "updateID"
```

ZOOPS_COEFFICIENTS

Primary key: `taxa`

Variables:

```
## [1] "taxa"      "slope"     "intercept" "equation"  "comments"  "updateID"
```

ZOOPS_LENGTHS

Measured lengths of sampled zooplankton.

Primary key: `zoopID`

Variables:

```
##  [1] "projectID"      "sampleID"       "zoopID"         "lakeID"        
##  [5] "siteName"       "dateSample"     "dateTimeSample" "depthClass"    
##  [9] "depthTop"       "depthBottom"    "taxa"           "length"        
## [13] "width"          "mass"           "eggs"           "metadataID"    
## [17] "updateID"
```

ZOOPS_PRODUCTION

Production calculations following Plante and Downing 1989.

Variables:

```
##  [1] "projectID"            "lakeID"               "siteID"              
##  [4] "yearSample"           "taxa"                 "production"          
##  [7] "prodSD"               "seasonalSD"           "production_m3"       
## [10] "prodSD_m3"            "seasonalSD_m3"        "production_eggRatio"
## [13] "metadataID"           "comments"             "updateID"
```

ZOOPS_SUBSAMPLE

Zooplankton subsamples taken to facilitate counting.

Variables:

```
##  [1] "projectID"      "sampleID"       "lakeID"         "siteName"      
##  [5] "dateSample"     "dateTimeSample" "depthClass"     "depthTop"      
##  [9] "depthBottom"    "wtEmpty"        "wtFull"         "wtSubsample"   
## [13] "metadataID"     "comments"       "updateID"
```