5. Bd-Capture Workflow

Cob Staines

2025-01-13

Table of contents

This tutorial is available as a [.qmd on Github](#).

Motivation

- Run through a complete workflow including: database connection, data discovery, data pulling, and data manipulation
- Demonstrate how to connect Capture data with sample data (in this case Bd qPCR results)

1

R

Let's run through a realistic data scenario from the beginning.

Suppose we are interested in Capture and Bd qPCR data. Specifically, we want to compare Bd qPCR results for juvenile-to-adult individuals, captured in 2015 or later, across species and sites.

Setup

```
# minimal packages for RIBBiTR DB data discovery
librarian::shelf(tidyverse, dbplyr, RPostgres, DBI, RIBBiTR-BII/ribbitrrr)

# establish database connection
dbcon <- hopToDB("ribbitr")
```

```
Connecting to database... Success!
```

```
# load table metadata
mdt <- tbl(dbcon, Id("public", "all_tables")) %>%
  filter(table_schema == "survey_data") %>%
  collect()

# load column metadata
mdc <- tbl(dbcon, Id("survey_data", "metadata_columns")) %>%
  filter(table_schema == "survey_data") %>%
  collect()
```

Data discovery and pulling

Looking at the survey_data schema diagram, we can browse to see which tables and columns we want. We can also consult the table or column metadata. The two observation tables with the primary data of interest are called "capture" and "bd_qpcr_results".

Point to support tables

```
# pointers for all tables of interest
db_bdqpcr = tbl(dbcon, Id("survey_data", "bd_qpcr_results"))
db_sample = tbl(dbcon, Id("survey_data", "sample"))
db_capture = tbl(dbcon, Id("survey_data", "capture"))
```

```
db_survey = tbl(dbcon, Id("survey_data", "survey"))
db_visit = tbl(dbcon, Id("survey_data", "visit"))
db_site = tbl(dbcon, Id("survey_data", "site"))
db_region = tbl(dbcon, Id("survey_data", "region"))
db_country = tbl(dbcon, Id("survey_data", "country"))
```

Join all data of interest

In this case we only want to consider cases for which he have both capture *and* Bd qPCR data. An inner join across the Bd, sample, and capture tables will keep only values which are common between them. Of these data, we want to return all supporting info, so we will left join to the remaining support tables.

```
# inner join capture and bd samples
# left join supporting tables
data_bd_capture = db_bdqpcr %>%
  inner_join(db_sample, by = "sample_id") %>%
  inner_join(db_capture, by = "capture_id") %>%
  left_join(db_survey, by = "survey_id") %>%
  left_join(db_visit, by = "visit_id") %>%
  left_join(db_site, by = "site_id") %>%
  left_join(db_region, by = "region_id") %>%
  left_join(db_country, by = "country_id")

# see what columns are available
colnames(data_bd_capture)
```

```
 [1] "result_id"                   "sample_id"
 [3] "sample_name_bd"              "detected"
 [5] "replicate"                   "replicate_count"
 [7] "replicate_results"          "average_ct"
 [9] "average_target_quant"       "total_qpcr_volume_uL"
[11] "qpcr_dilution_factor"       "volume_template_dna_uL"
[13] "extract_volume_uL"          "target_quant_per_swab"
[15] "average_its1_copies_per_swab" "swab_type"
[17] "standard_target_type"       "standard"
[19] "master_mix"                  "extraction_plate_name"
[21] "extraction_date"            "extraction_kit"
[23] "extraction_lab"             "qpcr_plate_name"
[25] "qpcr_well"                   "qpcr_plate_run"
[27] "qpcr_date"                   "qpcr_machine"
[29] "qpcr_lab"                    "comments_qpcr"
[31] "sample_name"                 "sample_type"
```

3

```
 [33] "capture_id"                      "taxon_capture"
 [35] "time_of_capture"                 "capture_trx_loc"
 [37] "microhabitat_type"              "body_temp_c"
 [39] "substrate_temp_c"               "svl_mm"
 [41] "body_mass_g"                     "life_stage"
 [43] "sex"                             "capture_animal_state"
 [45] "comments_capture"               "photo"
 [47] "photo_id"                        "microhab_moredetail"
 [49] "body_and_bag_mass_g"            "bag_mass_g"
 [51] "marked"                          "recapture_from_campaign_n_number"
 [53] "capture_utme"                    "capture_utmn"
 [55] "capture_type"                    "observer_capture"
 [57] "bag_id"                          "processor"
 [59] "cmr_id"                          "microhabitat_temperature"
 [61] "microhabitat_notes"             "tail_length_mm"
 [63] "buckets"                         "location_serdp"
 [65] "temp_gun"                        "clearcut"
 [67] "number_of_mites"                "flir"
 [69] "tad_stage"                       "survey_id"
 [71] "microhabitat_wet"               "capture_utm_zone"
 [73] "capture_latitude"               "capture_longitude"
 [75] "start_time"                      "end_time"
 [77] "detection_type"                 "duration_minutes"
 [79] "observers_survey"               "comments_survey"
 [81] "description"                     "survey_quality"
 [83] "transect"                        "number_observers"
 [85] "visit_id"                        "start_timestamp"
 [87] "end_timestamp"                  "date"
 [89] "time_of_day"                     "campaign"
 [91] "visit_status"                    "comments_visit"
 [93] "site_id"                         "visit_lab"
 [95] "site"                            "site_utm_zone"
 [97] "site_utme"                       "site_utmn"
 [99] "area_sqr_m"                      "site_code"
[101] "site_elevation_m"               "depth_m"
[103] "topo"                            "wilderness"
[105] "site_comments"                  "region_id"
[107] "site_name_alt"                  "site_latitude"
[109] "site_longitude"                 "region"
[111] "country_id"                      "location_id"
[113] "time_zone"                       "country"
[115] "iso_country_code"
```

```
# we can also see which columns come from specified tables, for context
colnames(db_bdqpcr)
```

```
 [1] "result_id"                 "sample_id"
 [3] "sample_name_bd"            "detected"
 [5] "replicate"                 "replicate_count"
 [7] "replicate_results"         "average_ct"
 [9] "average_target_quant"      "total_qpcr_volume_uL"
[11] "qpcr_dilution_factor"      "volume_template_dna_uL"
[13] "extract_volume_uL"         "target_quant_per_swab"
[15] "average_its1_copies_per_swab" "swab_type"
[17] "standard_target_type"      "standard"
[19] "master_mix"                "extraction_plate_name"
[21] "extraction_date"           "extraction_kit"
[23] "extraction_lab"            "qpcr_plate_name"
[25] "qpcr_well"                 "qpcr_plate_run"
[27] "qpcr_date"                 "qpcr_machine"
[29] "qpcr_lab"                  "comments_qpcr"
```

Select columns of interest, filter to date

```
# pull data from database
data_bd_capture_2015 = data_bd_capture %>%
  # filter to dates of interest
  filter(date >= "2015-01-01") %>%
  # select columns of interest
  select(capture_id,
         taxon_capture,
         life_stage,
         svl_mm,
         body_mass_g,
         survey_id,
         cmr_id,
         sample_id,
         sample_name_bd,
         detected,
         average_ct,
         average_target_quant,
         target_quant_per_swab,
         comments_capture,
         comments_qpcr,
         date,
```

```
        site,
        region,
        country)
```

Explore # of filtered observations by life stage, then filter again

```
data_bd_capture_2015 %>%
  select(life_stage) %>%
  group_by(life_stage) %>%
  summarise(row_count = n()) %>%
  arrange(desc(row_count)) %>%
  collect()
```

```
# A tibble: 14 x 2
   life_stage               row_count
   <chr>                      <int64>
 1 adult                        23157
 2 <NA>                          4605
 3 juvenile                      4123
 4 tadpole                       1787
 5 subadult                       998
 6 aquatic_larvae                 573
 7 metamorph                      429
 8 larva                          417
 9 unknown                        367
10 terrestrial_development        330
11 larvae                         218
12 metamorphosed                   28
13 eggmass                          7
14 Unknown                          1
```

```
data_bd_capture_2015_life <- data_bd_capture_2015 %>%
  filter(life_stage %in% c("juvenile",
                           "subadult",
                           "adult"),
        !is.na(life_stage))
```

Pull data

```
# check out our SQL "shopping list"
sql_render(data_bd_capture_2015_life)
```

```
<SQL> SELECT
  "capture_id",
  "taxon_capture",
  "life_stage",
  "svl_mm",
  "body_mass_g",
  "survey_id",
  "cmr_id",
  "sample_id",
  "sample_name_bd",
  "detected",
  "average_ct",
  "average_target_quant",
  "target_quant_per_swab",
  "comments_capture",
  "comments_qpcr",
  "date",
  "site",
  "region",
  "country"
FROM (
  SELECT
    "bd_qpcr_results".*,
    "sample_name",
    "sample_type",
    "sample"."capture_id" AS "capture_id",
    "taxon_capture",
    "time_of_capture",
    "capture_trx_loc",
    "microhabitat_type",
    "body_temp_c",
    "substrate_temp_c",
    "svl_mm",
    "body_mass_g",
    "life_stage",
    "sex",
    "capture_animal_state",
    "comments_capture",
    "photo",
    "photo_id",
    "microhab_moredetail",
```

```
"body_and_bag_mass_g",
"bag_mass_g",
"marked",
"recapture_from_campaign_n_number",
"capture_utme",
"capture_utmn",
"capture_type",
"observer_capture",
"bag_id",
"processor",
"cmr_id",
"microhabitat_temperature",
"microhabitat_notes",
"tail_length_mm",
"buckets",
"location_serdp",
"temp_gun",
"clearcut",
"number_of_mites",
"flir",
"tad_stage",
"capture"."survey_id" AS "survey_id",
"microhabitat_wet",
"capture_utm_zone",
"capture_latitude",
"capture_longitude",
"start_time",
"end_time",
"detection_type",
"duration_minutes",
"observers_survey",
"comments_survey",
"description",
"survey_quality",
"transect",
"number_observers",
"survey"."visit_id" AS "visit_id",
"start_timestamp",
"end_timestamp",
"date",
"time_of_day",
"campaign",
"visit_status",
"comments_visit",
"visit"."site_id" AS "site_id",
```

```
      "visit_lab",
      "site",
      "site_utm_zone",
      "site_utme",
      "site_utmn",
      "area_sqr_m",
      "site_code",
      "site_elevation_m",
      "depth_m",
      "topo",
      "wilderness",
      "site_comments",
      "site"."region_id" AS "region_id",
      "site_name_alt",
      "site_latitude",
      "site_longitude",
      "region",
      "region"."country_id" AS "country_id",
      "location_id",
      "time_zone",
      "country",
      "iso_country_code"
    FROM "survey_data"."bd_qpcr_results"
    INNER JOIN "survey_data"."sample"
      ON ("bd_qpcr_results"."sample_id" = "sample"."sample_id")
    INNER JOIN "survey_data"."capture"
      ON ("sample"."capture_id" = "capture"."capture_id")
    LEFT JOIN "survey_data"."survey"
      ON ("capture"."survey_id" = "survey"."survey_id")
    LEFT JOIN "survey_data"."visit"
      ON ("survey"."visit_id" = "visit"."visit_id")
    LEFT JOIN "survey_data"."site"
      ON ("visit"."site_id" = "site"."site_id")
    LEFT JOIN "survey_data"."region"
      ON ("site"."region_id" = "region"."region_id")
    LEFT JOIN "survey_data"."country"
      ON ("region"."country_id" = "country"."country_id")
) AS "q01"
WHERE
  ("date" >= '2015-01-01') AND
  ("life_stage" IN ('juvenile', 'subadult', 'adult')) AND
  (NOT(("life_stage" IS NULL)))
```

```
# collect data
data_bd_capture_final <- data_bd_capture_2015_life %>%
  collect()

head(data_bd_capture_final)
```

```
# A tibble: 6 x 19
  capture_id      taxon_capture life_stage svl_mm body_mass_g survey_id cmr_id
  <chr>           <chr>         <chr>       <dbl>       <dbl> <chr>     <chr>
1 1dc9fa0d-13bd-4d~ rana_muscosa  adult          47          12 bd398b88~ 0ba8e~
2 26089c8a-50bd-43~ rana_muscosa  adult          63          40 04c974ef~ <NA>
3 d9d1ba1b-7c53-47~ lithobates_s~ adult          48          14 87b52157~ <NA>
4 75cbebdb-663c-46~ rana_muscosa  adult          54          15 1ff0ec2e~ a68f1~
5 9c2be8d0-665e-4d~ rana_muscosa  adult          51          17 57d19e16~ fad92~
6 4afccde9-0672-41~ rana_muscosa  adult          52          21 c332c44e~ 38b9f~
# i 12 more variables: sample_id <chr>, sample_name_bd <chr>, detected <dbl>,
#   average_ct <dbl>, average_target_quant <dbl>, target_quant_per_swab <dbl>,
#   comments_capture <chr>, comments_qpcr <chr>, date <date>, site <chr>,
#   region <chr>, country <chr>
```

These data are ready to be analyzed and visualized!

Disconnect

```
dbDisconnect(dbcon)
```

Python

Let's run through a realistic data scenario from the beginning.

Suppose we are interested in Capture and Bd qPCR data. Specifically, we want to compare Bd qPCR results for juvenile-to-adult individuals, captured in 2015 or later, across species and sites.

Setup

```
# minimal packages for RIBBiTR DB Workflow
import ibis
from ibis import _
import pandas as pd
import dbconfig
import db_access as db

# establish database connection
dbcon = ibis.postgres.connect(**dbconfig.ribbitr)

# load table metadata
mdt = dbcon.table(database = "public", name = "all_tables").to_pandas()

# load column metadata
mdc = (
  dbcon.table(database="public", name="all_columns")
  .filter(_.table_schema == 'survey_data')
  .to_pandas()
  )
```

Data discovery and pulling

Looking at the survey_data schema diagram, we can browse to see which tables and columns
we want. We can also consult the table or column metadata. The two observation tables
with the primary data of interest are called "capture" and "bd_qpcr_results".

Point to support tables

```
# Ponters for all tables
db_bdqpcr = dbcon.table('bd_qpcr_results', database='survey_data')
db_sample = dbcon.table('sample', database='survey_data')
db_capture = dbcon.table('capture', database='survey_data')
db_survey = dbcon.table('survey', database='survey_data')
db_visit = dbcon.table('visit', database='survey_data')
db_site = dbcon.table('site', database='survey_data')
db_region = dbcon.table('region', database='survey_data')
db_country = dbcon.table('country', database='survey_data')
```

Join all data of interest

In this case we only want to consider cases for which he have both capture *and* Bd qPCR
data. An inner join across the Bd, sample, and capture tables will keep only values which

are common between them. Of these data, we want to return all supporting info, so we will left join to the remaining support tables.

```
# Recursive joins
data_bd_capture = (
    db_bdqpcr
    .inner_join(db_sample, db_bdqpcr.sample_id == db_sample.sample_id)
    .inner_join(db_capture, db_sample.capture_id == db_capture.capture_id)
    .left_join(db_survey, db_capture.survey_id == db_survey.survey_id)
    .left_join(db_visit, db_survey.visit_id == db_visit.visit_id)
    .left_join(db_site, db_visit.site_id == db_site.site_id)
    .left_join(db_region, db_site.region_id == db_region.region_id)
    .left_join(db_country, db_region.country_id == db_country.country_id)
)

# see what columns are available
data_bd_capture.columns
```

```
['result_id', 'sample_id', 'sample_name_bd', 'detected', 'replicate', 'replicate_count', 're
```

```
# we can also see which columns come from specified tables, for context
db_bdqpcr.columns
```

```
['result_id', 'sample_id', 'sample_name_bd', 'detected', 'replicate', 'replicate_count', 're
```

Select columns of interest, filter to date

```
# capture table, select, filter
data_bd_capture_2015 = (
  data_bd_capture
  .filter(_.date >= '2015-01-01')
  .select([
    "capture_id",
    "taxon_capture",
    "life_stage",
    "svl_mm",
    "body_mass_g",
    "survey_id",
    "cmr_id",
    "sample_id",
    "sample_name_bd",
    "detected",
```

```
        "average_ct",
        "average_target_quant",
        "target_quant_per_swab",
        "comments_capture",
        "comments_qpcr",
        "date",
        "site",
        "region",
        "country"
    ])
  )
```

Explore # of filtered observations by life stage, then filter again

```
# count by life stage
life_stage_counts = (
  data_bd_capture_2015
  .group_by('life_stage')
  .aggregate(row_count=_.count())
  .order_by(_.row_count.desc())
  .to_pandas()
  )

print(life_stage_counts)
```

```
                 life_stage   row_count
0                     adult       23157
1                      None        4605
2                  juvenile        4123
3                   tadpole        1787
4                  subadult         998
5             aquatic_larvae         573
6                  metamorph         429
7                     larva         417
8                   unknown         367
9     terrestrial_development         330
10                    larvae         218
11              metamorphosed          28
12                   eggmass           7
13                   Unknown           1
```

```
# filter to desired life stages
data_bd_capture_2015_life = (
  data_bd_capture_2015
  .filter(_.life_stage.isin(['juvenile','subadult', 'adult']) & _.life_stage.notnull())
  )
```

Pull data

```
# check out our SQP "shopping list"
data_bd_capture_2015_life.compile()
```

```
'SELECT "t16"."capture_id", "t16"."taxon_capture", "t16"."life_stage", "t16"."svl_mm", "t16"
```

```
# pull data
data_bd_capture_final = data_bd_capture_2015_life.to_pandas()

# preview data
data_bd_capture_final.head()
```

```
                       capture_id  ... country
0  1dc9fa0d-13bd-4d95-8594-10bb15e58d65  ...     usa
1  26089c8a-50bd-43d3-8660-8895bdf21467  ...     usa
2  d9d1ba1b-7c53-470b-9f6d-7db851d4c932  ...     usa
3  75cbebdb-663c-46bd-a9ca-653893d6e8cd  ...     usa
4  9c2be8d0-665e-4dff-8bca-9cb0417729a5  ...     usa

[5 rows x 19 columns]
```

These data are ready to be analyzed and visualized!

Disconnect

```
# close connection
dbcon.disconnect()
```

<- 4. Table Joins | 6. Microclimate Workflow ->