5. Bd-Capture Workflow

Cob Staines

2025-02-21

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)

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

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

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

```
Connecting to 'ribbitr'... Success!
```

```r
# 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

```r
# 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"))
```

```r
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.

```r
# 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_detected"               "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"
```

```
 [33] "capture_id"                        "sample_subtype"
 [35] "taxon_capture"                     "time_of_capture"
 [37] "capture_trx_loc"                   "microhabitat_type"
 [39] "body_temp_c"                       "substrate_temp_c"
 [41] "svl_mm"                            "body_mass_g"
 [43] "life_stage"                        "sex"
 [45] "capture_animal_state"              "comments_capture"
 [47] "photo"                             "photo_id"
 [49] "microhabitat_detailed"             "body_and_bag_mass_g"
 [51] "bag_mass_g"                        "marked"
 [53] "recapture_from_campaign_n_number" "capture_utme"
 [55] "capture_utmn"                      "capture_type"
 [57] "observer_capture"                  "bag_id"
 [59] "processor"                         "cmr_id"
 [61] "microhabitat_temperature"          "microhabitat_notes"
 [63] "tail_length_mm"                    "buckets"
 [65] "location_serdp"                    "temp_gun"
 [67] "clearcut"                          "number_of_mites"
 [69] "flir"                              "tad_stage"
 [71] "survey_id"                         "microhabitat_wet"
 [73] "capture_utm_zone"                  "capture_latitude"
 [75] "capture_longitude"                 "start_time"
 [77] "end_time"                          "detection_type"
 [79] "duration_minutes"                  "observers_survey"
 [81] "comments_survey"                   "description"
 [83] "survey_quality"                    "transect"
 [85] "number_observers"                  "visit_id"
 [87] "start_timestamp"                   "end_timestamp"
 [89] "date"                              "time_of_day"
 [91] "campaign"                          "visit_status"
 [93] "comments_visit"                    "site_id"
 [95] "visit_lab"                         "project"
 [97] "site"                              "site_utm_zone"
 [99] "site_utme"                         "site_utmn"
[101] "area_sqr_m"                        "site_code"
[103] "site_elevation_m"                  "depth_m"
[105] "topo"                              "wilderness"
[107] "site_comments"                     "region_id"
[109] "site_name_alt"                     "site_latitude"
[111] "site_longitude"                    "geographic_area"
[113] "geographic_area_type"              "region"
[115] "country_id"                        "time_zone"
[117] "country"                           "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_detected"           "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: 12 x 2
   life_stage    row_count
   <chr>           <int64>
 1 adult             28045
 2 juvenile           4231
 3 tadpole            1845
 4 subadult           1009
 5 <NA>                734
 6 metamorph           429
 7 larva               417
 8 unknown             367
 9 larvae              218
10 metamorphosed        28
11 eggmass               7
12 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

```
# inspect 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",
    "sample_subtype",
    "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",
    "microhabitat_detailed",
    "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",
    "project",
    "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",
    "geographic_area",
    "geographic_area_type",
    "region",
    "region"."country_id" AS "country_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)))
```

```r
# 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_sierrae  adult          47          12 bd398b88~ 0ba8e~
2 26089c8a-50bd-43~ rana_sierrae  adult          63          40 04c974ef~ <NA>
3 d9d1ba1b-7c53-47~ rana_sylvati~ adult          48          14 87b52157~ <NA>
4 242b1923-ab4b-45~ hylodes_phyl~ adult          NA          NA 3359378a~ <NA>
5 75cbebdb-663c-46~ rana_sierrae  adult          54          15 1ff0ec2e~ a68f1~
6 9c2be8d0-665e-4d~ rana_muscosa  adult          51          17 57d19e16~ fad92~
# i 12 more variables: sample_id <chr>, sample_name_bd <chr>, detected <lgl>,
#   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
```

```python
# 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

```python
# 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.

```python
# 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
```

Filter to date, select columns of interest

```
# capture table, filter, select
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      28045
1        juvenile       4231
2         tadpole       1845
3        subadult       1009
4            None        734
5       metamorph        429
6           larva        417
7         unknown        367
8          larvae        218
9   metamorphosed         28
10        eggmass          7
11        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

```
# inspect our SQL "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  242b1923-ab4b-4553-a2b9-5a679ba94a39  ...  brazil
4  75cbebdb-663c-46bd-a9ca-653893d6e8cd  ...     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 ->