Counting mismatches

Suman Sahil, Steve Simon

Creation date: 2019-10-16

List of tables

```
##            tbl_name
## 1       demog_table
## 2    baseline_table
## 3 three_month_table
## 4    one_year_table
## 5         sex_table
## 6    migraine_table
## 7       group_table
```

The examples shown here will use the database longitudinal_examples_db. Here is a list of the tables in this database.

Count of baseline_table

– SQL code

```
select count(*) as n_baseline
  from baseline_table
```

– SQL output

```
##   n_baseline
## 1        401
```

Here's the count of the number of records in baseline_table.

Count of three_month_table

- SQL code

```
select count(*) as n_mo3
  from three_month_table
```

- SQL output

```
##   n_mo3
## 1   326
```

And here's the count of the number of records in three_month_table.

Count inner join

– SQL code

```
select count(*) as n_inner_join
  from baseline_table as bas
  inner join three_month_table as mo3
  on bas.id=mo3.id
```

– SQL output

```
##   n_inner_join
## 1          326
```

HEre is the count of the inner join.

Count left join

– SQL code

```
select count(*) as n_left_join
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
```

– SQL output

```
##   n_left_join
## 1         401
```

Here is the count of the left join.

Count by gender (1/3)

– SQL code

```
select sex, count(*) as n_sex
  from demog_table
  group by sex
```

– SQL output

```
##   sex n_sex
## 1   0    64
## 2   1   337
```

Here is the count of males and females, but it uses the number code.

Count by gender (2/3)

– SQL code

```
select sex_label, count(*) as n_sex
  from demog_table
  left join sex_table
  on sex=sex_code
  group by sex_label
```

– SQL output

```
##   sex_label n_sex
## 1    Female   337
## 2      Male    64
```

Here is the count of males and females, using the sex_label field that is only available after the merge.

Count by gender (3/3)

– SQL code

```
select sex_label, count(sex) as n_sex
  from sex_table
  left join demog_table
  on sex=sex_code
  group by sex_label
```

– SQL output

```
##   sex_label n_sex
## 1    Female   337
## 2      Male    64
## 3   Unknown     0
```

There is an unmatched label (9=unknown) and if you wanted to include that in the table of counts, switch the order of tables and use count(sex) rather than count(*). There will be one row in the database for the mismatch, but since that will have a missing value for sex, the count function will not register for that record, producing a count of 0.

Who, exactly, is mismatched?

– SQL code

```
select bas.id as mismatched_id
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
  where mo3.id is null
  limit 4
```

– SQL output

```
##   mismatched_id
## 1           100
## 2           101
## 3           104
## 4           139
```

If you want a list of ids where two tables fail to match, look for a null value in one of the tables. There are 75 ids, so I could not print out all of them on this slide.

Listing unmatched labels

– SQL code

```
select sex_label as umatched_label
  from sex_table
  left join demog_table
  on sex=sex_code
  where sex is null
```

– SQL output

```
##   umatched_label
## 1        Unknown
```

You can use the same approach to look for unmatched labels or unmatched codes. Here is ab example of looking for unmatched labels.

Listing unmatched codes

– SQL code

```
select sex as umatched_code
  from demog_table
  left join sex_table
  on sex=sex_code
  where sex_code is null
```

– SQL output

```
## [1] umatched_code
## <0 rows> (or 0-length row.names)
```

By swapping the order of tables (or changing to a right join) and making a few other small changes, you can find codes in the demography database that do not have a corresponding label.

Compute change score

– SQL code

```
select pk1, pk2, pk2-pk1 as change_score
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
  limit 5
```

– SQL output

```
##     pk1  pk2 change_score
## 1 10.75   NA           NA
## 2  9.50   NA           NA
## 3 16.00   NA           NA
## 4 32.50 44.0         11.5
## 5 16.50 17.5          1.0
```

You can compute values across different tables. This example shows a change score. Notice that the change score is null if either value is null.

Compute average change score

– SQL code

```
select
  round(avg(pk1), 1) as bas_mean,
  round(avg(pk2), 1) as mo3_mean,
  round(avg(pk2-pk1), 1) as mean_change
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
```

– SQL output

```
##   bas_mean mo3_mean mean_change
## 1     26.5     21.6        -4.7
```

You can even average these values. Notice that the mean change is not exactly equal to the difference in the two means. That would normally be the case, but remember that the baseline average is an average that includes 75 patients wihtout a corresponding three month value. You could restrict the data so that the baseline average is only computed for those patients who have a matching three month value. You could do this using an inner join or by restricting the data to cases where pk2 is not null.

Compare drop outs (1/3)

– SQL code

```
select avg(pk1) as mean_baseline, count(*) as n
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
```

– SQL output

```
##   mean_baseline   n
## 1      26.50998 401
```

An important comparison is what the baseline values look like for the drop outs compared to those who stay in the study. This code shows how to get an overall mean baseline score.

Compare drop outs (2/3)

– SQL code

```
select avg(pk1) as mean_baseline, count(*) as n
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
  where mo3.id is null
```

– SQL output

```
##   mean_baseline  n
## 1      27.36667 75
```

Here is the mean for those who dropped out.

Compare drop outs (3/3)

– SQL code

```
select avg(pk1) as mean_baseline, count(*) as n
  from baseline_table as bas
  left join three_month_table as mo3
  on bas.id=mo3.id
  where mo3.id is not null
```

– SQL output

```
##   mean_baseline   n
## 1      26.31288 326
```

Here is the mean for those who did not drop out.

Summary

- Counts are very important
  - Before AND after joins
- How to select ids of mismatched values
- How to find mismatched category labels
- How to compute change scores
- Comparing averages of drop outs to others

Homework (1/2)

- Use the same database used in this video. It is available as longitudinal_example_db.sqlite on the Canvas website or you can find it on the Insights platform.
- Count the number of records after an inner join of baseline_table and year_one_table. Compare this to the number of records in the year_one_table.

Homework (2/2)

- Compute the average pk score at baseline, the average score at one year, and the average change score.
- Find and list the two labels in migraine_table that do not correspond to any codes in demog_table.
- Show that there are no unmatched labels or unmatched codes for group_table.