Course 2 Section 1.9 - Joins

Jiaying Wu

```r
library(tidyverse)
```

Bank data: left join

Below are two made-up data sets: customer and transaction.

- customer contains some personal information about a bank's customers.
- transaction contains spending transactions from some customers of the bank

```r
# Create bank customer info data
customer <- tibble(customer_id = c("C120463", "N4244552", "A907892", "Z337572", "D701453", "A285883"),
                   names = c("John", "Sam", "Mike", "Rachael", "Mary", "Will"),
                   sex = c("M", "F", "M", "F", "F", "M"),
                   educ = c("HS", "Bachelor", "PhD", "Bachelor", "HS", "HS"),
                   wage = c(1200, 900, 3100, 600, 500, 800))

# Print customer
customer
```

```
## # A tibble: 6 x 5
##   customer_id names   sex   educ      wage
##   <chr>       <chr>   <chr> <chr>    <dbl>
## 1 C120463     John    M     HS        1200
## 2 N4244552    Sam     F     Bachelor   900
## 3 A907892     Mike    M     PhD       3100
## 4 Z337572     Rachael F     Bachelor   600
## 5 D701453     Mary    F     HS         500
## 6 A285883     Will    M     HS         800
```

```r
# Create bank spending transaction data
transaction <- tibble(customer_id = c("D701453", "N4244552", "C120463", "A907892", "D701453", "C120463"
                      shop = c("JB Hifi", "Steakhouse", "Apple", "Coles", "Lobster Diner", "Dymocks", "
                      amount = c(300, 110, 3000, 80, 185, 40, 25, 15, 170))

# Print transaction
transaction
```

```
## # A tibble: 9 x 3
##   customer_id shop          amount
##   <chr>       <chr>          <dbl>
## 1 D701453     JB Hifi          300
## 2 N4244552    Steakhouse       110
## 3 C120463     Apple           3000
## 4 A907892     Coles             80
## 5 D701453     Lobster Diner    185
## 6 C120463     Dymocks           40
## 7 N4244552    Target            25
## 8 C120463     Netflix           15
## 9 A907892     Mecca            170
```

Q1: left join customer into transaction by the customer_id variable.

```
# Left join customer into transaction by customer_id
transaction_customer <- left_join(transaction, customer, by = "customer_id")

# Print transaction_customer
transaction_customer
```

```
## # A tibble: 9 x 7
##   customer_id shop          amount names sex   educ      wage
##   <chr>       <chr>          <dbl> <chr> <chr> <chr>    <dbl>
## 1 D701453     JB Hifi          300 Mary  F     HS         500
## 2 N4244552    Steakhouse       110 Sam   F     Bachelor   900
## 3 C120463     Apple           3000 John  M     HS        1200
## 4 A907892     Coles             80 Mike  M     PhD       3100
## 5 D701453     Lobster Diner    185 Mary  F     HS         500
## 6 C120463     Dymocks           40 John  M     HS        1200
## 7 N4244552    Target            25 Sam   F     Bachelor   900
## 8 C120463     Netflix           15 John  M     HS        1200
## 9 A907892     Mecca            170 Mike  M     PhD       3100
```

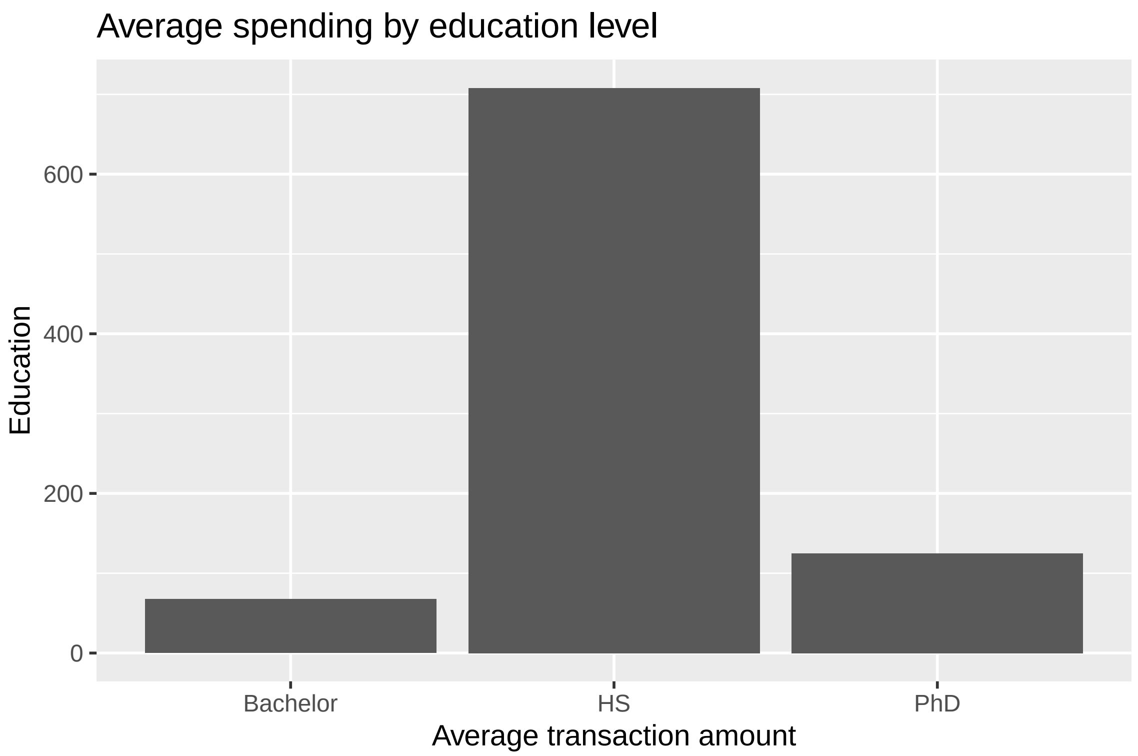
```
# Above join can also be done with the pipe operator (%>%)
transaction %>% left_join(customer, by = "customer_id")
```

```
## # A tibble: 9 x 7
##   customer_id shop          amount names sex   educ      wage
##   <chr>       <chr>          <dbl> <chr> <chr> <chr>    <dbl>
## 1 D701453     JB Hifi          300 Mary  F     HS         500
## 2 N4244552    Steakhouse       110 Sam   F     Bachelor   900
## 3 C120463     Apple           3000 John  M     HS        1200
## 4 A907892     Coles             80 Mike  M     PhD       3100
## 5 D701453     Lobster Diner    185 Mary  F     HS         500
## 6 C120463     Dymocks           40 John  M     HS        1200
## 7 N4244552    Target            25 Sam   F     Bachelor   900
## 8 C120463     Netflix           15 John  M     HS        1200
## 9 A907892     Mecca            170 Mike  M     PhD       3100
```

customer_id is the variable that connects both data sets together. Left joining customer into transaction using customer_id as the connecting variable add the variables names, sex, educ and wage into the transactions data.

Q2: One insight we can gather from this joined data set, which we have named transaction_customer, is the average amount spent by individuals of various education levels:

```r
# Bar plot of average spending by education
transaction_customer %>%
  # group data by education
  group_by(educ) %>%
  # compute average spending (by education)
  summarise(avg_amount = mean(amount)) %>%
  ungroup() %>%
  # aesthetic for bar plot
  ggplot(aes(x = educ, y = avg_amount)) +
  # bar is the visual element we want
  # stat = "identity" to tell R to use y aesthetic and not it's own count for the y aesthetic
  geom_bar(stat = "identity") +
  # label the graph
  labs(title = "Average spending by education level",
       y = "Education",
       x = "Average transaction amount")
```



Joining the bank's customers' personal information to data about spending transactions is different from joining data about transactions to the personal details of banking customers.

Put differently, a left join of customer into transaction produces a joined data set that differs from a left join of transaction into customer.

Q3: perform a left join of transaction into customer

```
# Left join transaction into customer by customer_id
customer %>%
  left_join(transaction, by = "customer_id")
```

```
## # A tibble: 11 x 7
##    customer_id names   sex   educ      wage shop          amount
##    <chr>       <chr>   <chr> <chr>    <dbl> <chr>          <dbl>
##  1 C120463     John    M     HS        1200 Apple           3000
##  2 C120463     John    M     HS        1200 Dymocks           40
##  3 C120463     John    M     HS        1200 Netflix           15
##  4 N4244552    Sam     F     Bachelor   900 Steakhouse       110
##  5 N4244552    Sam     F     Bachelor   900 Target            25
##  6 A907892     Mike    M     PhD       3100 Coles             80
##  7 A907892     Mike    M     PhD       3100 Mecca            170
##  8 Z337572     Rachael F     Bachelor   600 <NA>              NA
##  9 D701453     Mary    F     HS         500 JB Hifi          300
## 10 D701453     Mary    F     HS         500 Lobster Diner    185
## 11 A285883     Will    M     HS         800 <NA>              NA
```

Bank data: inner join

An inner join of customer and transaction keeps only observations with customer IDs that are in both the customer and transaction data.

Q1:

```
# Inner join customer and transaction by customer_id
customer %>%
  inner_join(transaction, by = "customer_id")
```

```
## # A tibble: 9 x 7
##   customer_id names sex   educ      wage shop          amount
##   <chr>       <chr> <chr> <chr>    <dbl> <chr>          <dbl>
## 1 C120463     John  M     HS        1200 Apple           3000
## 2 C120463     John  M     HS        1200 Dymocks           40
## 3 C120463     John  M     HS        1200 Netflix           15
## 4 N4244552    Sam   F     Bachelor   900 Steakhouse       110
## 5 N4244552    Sam   F     Bachelor   900 Target            25
## 6 A907892     Mike  M     PhD       3100 Coles             80
## 7 A907892     Mike  M     PhD       3100 Mecca            170
## 8 D701453     Mary  F     HS         500 JB Hifi          300
## 9 D701453     Mary  F     HS         500 Lobster Diner    185
```

For inner joins, it does not matter which data set comes first, i.e, whether customer or transaction is placed before %>% does not produce an informatively different inner joined data. The only difference that we would see is how the columns and rows are arranged.

Notice that the first left join data is the same as this inner join data. Execute the following code chunk to inner join transaction and FAANG. The resulting inner join data is shown below.

```r
# Create data of 2019 FAANG revenue
FAANG <- tibble(company = c("Facebook", "Amazon", "Apple", "Netflix", "Alphabet (formerly Google)"),
                revenue_billion = c(70.7, 280.5, 260.2, 20.16, 161.86))

# Print FAANG
FAANG
```

```
## # A tibble: 5 x 2
##   company                    revenue_billion
##   <chr>                                <dbl>
## 1 Facebook                              70.7
## 2 Amazon                               280.
## 3 Apple                                260.
## 4 Netflix                               20.2
## 5 Alphabet (formerly Google)           162.
```

Q2:

```r
# Inner join transaction and FAANG by shop = company
transaction %>%
  inner_join(FAANG, by = c("shop" = "company"))
```

```
## # A tibble: 2 x 4
##   customer_id shop    amount revenue_billion
##   <chr>       <chr>    <dbl>           <dbl>
## 1 C120463     Apple     3000           260.
## 2 C120463     Netflix     15            20.2
```

Joining data from nycflights13

Previously, we explored the flights data set from the nycflights13 package. Other data sets included in the nycflights13 package include airlines and airports. Execute the following code chunk to explore all 3 data sets.

```r
# Load nycflights13
library(nycflights13)
```

```r
# Head of flights
head(flights)
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## 4  2013     1     1      544            545        -1     1004           1022
## 5  2013     1     1      554            600        -6      812            837
## 6  2013     1     1      554            558        -4      740            728
## # ... with 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```r
# Head of airlines
head(airlines)
```

```
## # A tibble: 6 x 2
##   carrier name
##   <chr>   <chr>
## 1 9E      Endeavor Air Inc.
## 2 AA      American Airlines Inc.
## 3 AS      Alaska Airlines Inc.
## 4 B6      JetBlue Airways
## 5 DL      Delta Air Lines Inc.
## 6 EV      ExpressJet Airlines Inc.
```

```r
# Head of airports
head(airports)
```

```
## # A tibble: 6 x 8
##   faa   name                             lat   lon   alt    tz dst   tzone
##   <chr> <chr>                          <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport               41.1 -80.6  1044    -5 A     America/New_Y~
## 2 06A   Moton Field Municipal Airp~     32.5 -85.7   264    -6 A     America/Chica~
## 3 06C   Schaumburg Regional             42.0 -88.1   801    -6 A     America/Chica~
## 4 06N   Randall Airport                 41.4 -74.4   523    -5 A     America/New_Y~
## 5 09J   Jekyll Island Airport           31.1 -81.4    11    -5 A     America/New_Y~
## 6 0A9   Elizabethton Municipal Air~     36.4 -82.2  1593    -5 A     America/New_Y~
```

To add the full airline name to the flights data set, you can combine airlines to flights with the left_join() function, as shown in the following code chunk. Execute the following code chunks to left join airlines to flights and airports to flights.

```r
# Left join airlines into flights
flights %>% left_join(airlines, by = "carrier")
```

```
## # A tibble: 336,776 x 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 336,766 more rows, and 12 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   name <chr>
```

```r
# Left join airports into flights
flights %>% left_join(airports, by = c("origin" = "faa"))
```

```
## # A tibble: 336,776 x 26
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 336,766 more rows, and 18 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   name <chr>, lat <dbl>, lon <dbl>, alt <dbl>, tz <dbl>, dst <chr>,
## #   tzone <chr>
```