In [1]:
```
library(tidyverse)
library(nycflights13)
```

```
── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──

✓ ggplot2 3.3.3     ✓ purrr   0.3.4
✓ tibble  3.0.6     ✓ dplyr   1.0.4
✓ tidyr   1.1.2     ✓ stringr 1.4.0
✓ readr   1.4.0     ✓ forcats 0.5.1

── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
```

- What are the primary and foreign keys in the two tables below?
- Show that they are keys!

In [2]:
```
# US panda births
us_born_pandas = read_csv('data/us_born_pandas.csv')

# Current pandas in the United States
us_current_pandas = read_csv('data/us_current_pandas.csv')
```

```
── Column specification ────────────────────────────────────────────────────────
cols(
  name = col_character(),
  birth_date = col_character(),
  birth_location = col_character()
)


── Column specification ────────────────────────────────────────────────────────
cols(
  name = col_character(),
  location = col_character(),
  sex = col_character()
)
```

In [3]:
```
us_born_pandas
us_current_pandas
```

A spec_tbl_df: 15 × 3

name	birth_date	birth_location
<chr>	<chr>	<chr>
Hua Mei	8/21/99	San Diego Zoo
Mei Sheng	8/19/03	San Diego Zoo
Su Lin	8/2/05	San Diego Zoo
Yun Zi	8/5/09	San Diego Zoo
Zhen Zhen	8/1/07	San Diego Zoo
Xiao Liwu	7/29/12	San Diego Zoo
Mei Lan	9/6/06	Atlanta Zoo
Xi Lan	8/30/08	Atlanta Zoo
Po	10/3/10	Atlanta Zoo
Mei Lun	7/15/13	Atlanta Zoo
Mei Huan	7/15/13	Atlanta Zoo
Tai Shan	7/9/05	Smithsonian National Zoo
Bao Bao	8/23/13	Smithsonian National Zoo
Bei Bei	8/22/15	Smithsonian National Zoo
Xiao Qi Ji	8/21/20	Smithsonian National Zoo

A spec_tbl_df: 8 × 3

name	location	sex
<chr>	<chr>	<chr>
Le Le	Memphis Zoo	male
Ya Ya	Memphis Zoo	female
Lun Lun	Atlanta Zoo	female
Mei Lan	Atlanta Zoo	male
Yang Yang	Atlanta Zoo	male
Tian Tian	Smithsonian National Zoo	male
Mei Xiang	Smithsonian National Zoo	female
Xiao Qi Ji	Smithsonian National Zoo	male

In [4]:
```
# a proof that name is a primary key
us_current_pandas %>%
    count(name)

us_born_pandas %>%
    count(name)
```

A spec_tbl_df: 8 × 2

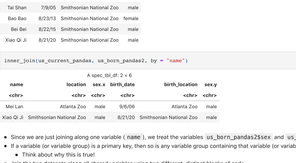
	name	n
	<chr>	<int>
1	Le Le	1
2	Lun Lun	1
3	Mei Lan	1
4	Mei Xiang	1
5	Tian Tian	1
6	Xiao Qi Ji	1
7	Ya Ya	1
8	Yang Yang	1

A spec_tbl_df: 15 × 2

	name	n
	<chr>	<int>
1	Bao Bao	1
2	Bei Bei	1
3	Hua Mei	1
4	Mei Huan	1
5	Mei Lan	1
6	Mei Lun	1
7	Mei Sheng	1
8	Po	1
9	Su Lin	1
10	Tai Shan	1
11	Xi Lan	1
12	Xiao Liwu	1
13	Xiao Qi Ji	1
14	Yun Zi	1
15	Zhen Zhen	1

Inner Join

Observations are matched whenever the keys are equal



Before we inner join the two datasets, answer the following:

- How many observations in the resulting dataset?
- How many variables in the resulting dataset?
- Will there be missing values? If so, where?
- How would you describe the resulting dataset in words?

In [7]:
```
inner_join(us_current_pandas, us_born_pandas, by = "name")
```

A spec_tbl_df: 2 × 5

name	location	sex	birth_date	birth_location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo

In [8]:
```
# if we don't specify a key, inner_join will use all shared variable names
inner_join(us_current_pandas, us_born_pandas)
```

```
Joining, by = "name"
```

A spec_tbl_df: 2 × 5

name	location	sex	birth_date	birth_location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo

In [9]:
```
# we can also use the pipe
# (pipe always uses the previous result as the first argument of the next line)
us_current_pandas %>%
    inner_join(us_born_pandas, by = "name")
```

A spec_tbl_df: 2 × 5

name	location	sex	birth_date	birth_location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo

In [11]:
```
# what if the datasets share more than variable?
(us_born_pandas2 = read_csv('data/us_born_pandas2.csv'))
```

```
── Column specification ────────────────────────────────────────────────────────
cols(
  name = col_character(),
  birth_date = col_character(),
  birth_location = col_character(),
  sex = col_character()
)
```

A spec_tbl_df: 15 × 4

name	birth_date	birth_location	sex
<chr>	<chr>	<chr>	<chr>
Hua Mei	8/21/99	San Diego Zoo	female
Mei Sheng	8/19/03	San Diego Zoo	female
Su Lin	8/2/05	San Diego Zoo	female
Yun Zi	8/5/09	San Diego Zoo	male
Zhen Zhen	8/1/07	San Diego Zoo	female
Xiao Liwu	7/29/12	San Diego Zoo	male
Mei Lan	9/6/06	Atlanta Zoo	male
Xi Lan	8/30/08	Atlanta Zoo	male
Po	10/3/10	Atlanta Zoo	female
Mei Lun	7/15/13	Atlanta Zoo	female
Mei Huan	7/15/13	Atlanta Zoo	female
Tai Shan	7/9/05	Smithsonian National Zoo	male
Bao Bao	8/23/13	Smithsonian National Zoo	female
Bei Bei	8/22/15	Smithsonian National Zoo	male
Xiao Qi Ji	8/21/20	Smithsonian National Zoo	male

In [12]:
```
inner_join(us_current_pandas, us_born_pandas2, by = "name")
```

A spec_tbl_df: 2 × 6

name	location	sex.x	birth_date	birth_location	sex.y
<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo	male
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo	male

- Since we are just joining along one variable (`name`), we treat the variables `us_born_pandas2$sex` and `us_current_pandas$sex` as two distinct variables when joining
- If a variable (or variable group) is a primary key, then so is any variable group containing that variable (or variable group)
  - Think about why this is true!
- Join the two datasets along all shared variables using two different, distinct blocks of code.

In [14]:
```
inner_join(us_current_pandas, us_born_pandas2, by = c('name', 'sex'))
```

A spec_tbl_df: 2 × 5

name	location	sex	birth_date	birth_location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo

In [15]:
```
inner_join(us_current_pandas, us_born_pandas2)
```

```
Joining, by = c("name", "sex")
```

A spec_tbl_df: 2 × 5

name	location	sex	birth_date	birth_location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	Atlanta Zoo	male	9/6/06	Atlanta Zoo
Xiao Qi Ji	Smithsonian National Zoo	male	8/21/20	Smithsonian National Zoo

In [16]:
```
# let's make things a little messier...
us_current_pandas2 <- rename(us_current_pandas, panda_name = name, sex_of_panda = sex)
```

A spec_tbl_df: 8 × 3

panda_name	location	sex_of_panda
<chr>	<chr>	<chr>
Le Le	Memphis Zoo	male
Ya Ya	Memphis Zoo	female
Lun Lun	Atlanta Zoo	female
Mei Lan	Atlanta Zoo	male
Yang Yang	Atlanta Zoo	male
Tian Tian	Smithsonian National Zoo	male
Mei Xiang	Smithsonian National Zoo	female
Xiao Qi Ji	Smithsonian National Zoo	male

In [17]:
```
# if the names of the variables aren't exactly the same, we have to specify this
inner_join(us_born_pandas2, us_current_pandas2, by = c("name" = "panda_name", "sex" = "sex_of_panda"))
```

A spec_tbl_df: 2 × 5

name	birth_date	birth_location	sex	location
<chr>	<chr>	<chr>	<chr>	<chr>
Mei Lan	9/6/06	Atlanta Zoo	male	Atlanta Zoo
Xiao Qi Ji	8/21/20	Smithsonian National Zoo	male	Smithsonian National Zoo

In [18]:
```
# why doesn't the following code work?
inner_join(us_born_pandas2, us_current_pandas2, by = c("panda_name" = "name", "sex_of_panda" = "sex"))
```

```
Error: Join columns must be present in data.
✖ Problem with `panda_name` and `sex_of_panda`.
Traceback:

1. inner_join(us_born_pandas2, us_current_pandas2, by = c(panda_name = "name",
 .     sex_of_panda = "sex"))
2. inner_join.data.frame(us_born_pandas2, us_current_pandas2, by = c(panda_name = "name",
 .     sex_of_panda = "sex"))
3. join_mutate(x, y, by, type = "inner", suffix = suffix, na_matches = na_matches,
 .     keep = keep)
4. join_cols(tbl_vars(x), tbl_vars(y), by = by, suffix = suffix,
 .     keep = keep)
5. standardise_join_by(by, x_names = x_names, y_names = y_names)
6. check_join_vars(by$x, x_names)
7. abort(c("Join columns must be present in data.", x = glue("Problem with {err_vars(missing)}.")))
8. signal_abort(cnd)
```

Attempt

- Use the `summarize()` and `filter()` commands on the `flights` dataset to get a dataset called `avg_delay_hour` containing the mean departure delay each hour of the year for flights whose `origin` is JFK.
  - Hint: First filter by flights with whose `origin` is JFK. The variables in the resulting dataset should be mean_delay, year, month, day, and hour
  - Is there a primary key?
- Use the `weather` dataset as a starting point to get a dataset called `JFK_weather` representing the weather at every hour at JFK airport.
  - Is there a primary key?
- Think about how you might join `avg_delay_hour` and `JFK_weather` to a new dataset `weather_and_delay`
  - What would be a good primary key for weather in this scenario?
  - How many rows are in `avg_delay_hour`, `JFK_weather`, and `weather_and_delay`? What happened?

In [ ]:
```
avg_delay_hour <- flights %>%
    filter(origin == "JFK") %>%
    group_by(year, month, day, hour) %>%
    summarize(mean_delay = mean(dep_delay, na.rm = TRUE)) %>%
    print()
```

In [ ]:
```
JFK_weather <- weather %>%
    filter(origin == "JFK") %>%
    print(width = Inf)
```