Wrangling Variables

Name: SOLUTIONS

Part 1

The Data

This is the data set you'll work with for every exercise in Part 1 of this handout: `songs`, 10 rows and 5 columns.

```
# A tibble: 10 x 5
   track_name      genre   release_year duration_sec streams_millions
   <chr>           <chr>          <dbl>        <dbl>             <dbl>
 1 Blinding Lights Pop             2020          200                87
 2 Bad Guy         Pop             2019          194                65
 3 Levitating      Pop             2020          203                72
 4 Sunflower       Hip-Hop         2018          158                58
 5 Perfect         Pop             2017          263                45
 6 Old Town Road   Hip-Hop         2019          113                90
 7 Shape of You    Pop             2017          233                55
 8 Circles         Indie           2019          215                30
 9 Ocean Eyes      Indie           2015          200                12
10 Heat Waves      <NA>            2020          238                68
```

Exercise 1

Suppose we run:

```
songs |>
  rename(
    music_genre = genre,
    stream_count_millions = streams_millions
  )
```

Show the changes made to the `songs` data frame below by writing on the values/variables and/or crossing rows and/or columns.

```
# A tibble: 10 x 5
   track_name      music_genre release_year duration_sec stream_count_millions
   <chr>           <chr>              <dbl>        <dbl>                  <dbl>
 1 Blinding Lights Pop                 2020          200                     87
 2 Bad Guy         Pop                 2019          194                     65
 3 Levitating      Pop                 2020          203                     72
 4 Sunflower       Hip-Hop             2018          158                     58
 5 Perfect         Pop                 2017          263                     45
 6 Old Town Road   Hip-Hop             2019          113                     90
 7 Shape of You    Pop                 2017          233                     55
 8 Circles         Indie               2019          215                     30
 9 Ocean Eyes      Indie               2015          200                     12
10 Heat Waves      <NA>                2020          238                     68
```

Exercise 2

Suppose we run:

```
songs |>
  select(track_name, release_year, streams_millions)
```

Show the changes made to the `songs` data frame below by writing on the values/variables and/or crossing rows and/or columns.

```
# A tibble: 10 x 3
   track_name      release_year streams_millions
   <chr>                  <dbl>            <dbl>
 1 Blinding Lights         2020               87
 2 Bad Guy                 2019               65
 3 Levitating              2020               72
 4 Sunflower               2018               58
 5 Perfect                 2017               45
 6 Old Town Road           2019               90
 7 Shape of You            2017               55
 8 Circles                 2019               30
 9 Ocean Eyes              2015               12
10 Heat Waves              2020               68
```

Exercise 3

Suppose we run:

```
songs |>
  filter(release_year >= 2019)
```

Shade the rows below that would remain after running this code.

```
# A tibble: 6 x 5
  track_name      genre   release_year duration_sec streams_millions
  <chr>           <chr>          <dbl>        <dbl>             <dbl>
1 Blinding Lights Pop             2020          200                87
2 Bad Guy         Pop             2019          194                65
3 Levitating      Pop             2020          203                72
4 Old Town Road   Hip-Hop         2019          113                90
5 Circles         Indie           2019          215                30
6 Heat Waves      <NA>            2020          238                68
```

How many rows would remain? 6

Exercise 4

Suppose we run:

```
songs |>
  filter(genre == "Pop" & streams_millions >= 60)
```

Shade the rows below that satisfy **both** conditions.

```
# A tibble: 5 x 5
  track_name      genre release_year duration_sec streams_millions
  <chr>           <chr>        <dbl>        <dbl>            <dbl>
1 Blinding Lights Pop           2020          200               87
2 Bad Guy         Pop           2019          194               65
3 Levitating      Pop           2020          203               72
4 Perfect         Pop           2017          263               45
5 Shape of You    Pop           2017          233               55
```

Answer:

```
# A tibble: 3 x 5
  track_name      genre release_year duration_sec streams_millions
  <chr>           <chr>        <dbl>        <dbl>            <dbl>
1 Blinding Lights Pop           2020          200               87
2 Bad Guy         Pop           2019          194               65
3 Levitating      Pop           2020          203               72
```

Exercise 5

Suppose we run:

```
specific_years <- c(2017, 2020)

songs |>
  filter(release_year %in% specific_years)
```

Shade the rows below whose `release_year` is in `specific_years`.

```
# A tibble: 5 x 5
  track_name      genre release_year duration_sec streams_millions
  <chr>           <chr>        <dbl>        <dbl>             <dbl>
1 Blinding Lights Pop           2020          200                87
2 Levitating      Pop           2020          203                72
3 Perfect         Pop           2017          263                45
4 Shape of You    Pop           2017          233                55
5 Heat Waves      <NA>          2020          238                68
```

How many rows would remain? 5

Exercise 6

Suppose we run:

```
songs |>
  filter(is.na(genre))
```

Shade the row(s) below that would remain after running the code above.

```
# A tibble: 1 x 5
  track_name genre release_year duration_sec streams_millions
  <chr>      <chr>        <dbl>        <dbl>            <dbl>
1 Heat Waves <NA>          2020          238               68
```

Exercise 7

Suppose we run:

```
songs |>
  mutate(
    viral_status = if_else(
      streams_millions >= 50,
      "Viral hit",
      "Not yet viral"
    )
  )
```

Fill in only the new `viral_status` column below (write over the blanks).

```
# A tibble: 10 x 6
   track_name      genre   release_year duration_sec streams_millions viral_status
   <chr>           <chr>          <dbl>        <dbl>            <dbl> <chr>
 1 Blinding Lights Pop             2020          200               87 Viral hit
 2 Bad Guy         Pop             2019          194               65 Viral hit
 3 Levitating      Pop             2020          203               72 Viral hit
 4 Sunflower       Hip-Hop         2018          158               58 Viral hit
 5 Perfect         Pop             2017          263               45 Not yet viral
 6 Old Town Road   Hip-Hop         2019          113               90 Viral hit
 7 Shape of You    Pop             2017          233               55 Viral hit
 8 Circles         Indie           2019          215               30 Not yet viral
 9 Ocean Eyes      Indie           2015          200               12 Not yet viral
10 Heat Waves      <NA>            2020          238               68 Viral hit
```

Exercise 8

Suppose we run:

```
songs |>
  mutate(
    stream_tier = case_when(
      streams_millions < 20  ~ "Under 20M",
      streams_millions < 50  ~ "20M to 49M",
      streams_millions >= 50 ~ "50M or more"
    )
  )
```

Fill in only the new `stream_tier` column below (write over the blanks).

```
# A tibble: 10 x 6
   track_name      genre   release_year duration_sec streams_millions stream_tier
   <chr>           <chr>          <dbl>        <dbl>             <dbl> <chr>
 1 Blinding Lights Pop             2020          200                87 50M or more
 2 Bad Guy         Pop             2019          194                65 50M or more
 3 Levitating      Pop             2020          203                72 50M or more
 4 Sunflower       Hip-Hop         2018          158                58 50M or more
 5 Perfect         Pop             2017          263                45 20M to 49M
 6 Old Town Road   Hip-Hop         2019          113                90 50M or more
 7 Shape of You    Pop             2017          233                55 50M or more
 8 Circles         Indie           2019          215                30 20M to 49M
 9 Ocean Eyes      Indie           2015          200                12 Under 20M
10 Heat Waves      <NA>            2020          238                68 50M or more
```

Exercise 9

Suppose we run:

```
songs |>
  group_by(genre) |>
  summarize(
    med_duration_sec = median(duration_sec),
    count = n()
  )
```

Use Exercise 2's table to work out the median duration and count of songs for each genre. The group labels are already filled in — write your answers over the blanks. Remember: the missing `genre` value forms its own group.

```
# A tibble: 4 x 3
  genre   med_duration_sec count
  <chr>              <dbl> <int>
1 Hip-Hop             136.     2
2 Indie               208.     2
3 Pop                 203      5
4 <NA>                238      1
```

Exercise 10

Suppose we run the following chain, **in order**:

```
songs |>
  filter(genre != "Indie") |>
  mutate(
    viral_status = if_else(
      streams_millions >= 50,
      "Viral hit",
      "Not yet viral"
    )
  ) |>
  group_by(viral_status) |>
  summarize(count = n())
```

1. Shade the rows below that survive the `filter()` step. (Hint: think about what `!=` does with the missing `genre` value, and check your answer with `is.na()` logic from Exercise 6 if you're not sure.)

```
# A tibble: 7 x 5
  track_name      genre   release_year duration_sec streams_millions
  <chr>           <chr>          <dbl>        <dbl>            <dbl>
1 Blinding Lights Pop             2020          200               87
2 Bad Guy         Pop             2019          194               65
3 Levitating      Pop             2020          203               72
4 Sunflower       Hip-Hop         2018          158               58
5 Perfect         Pop             2017          263               45
6 Old Town Road   Hip-Hop         2019          113               90
7 Shape of You    Pop             2017          233               55
```

2. List which tracks remain after `filter()`: Blinding Lights, Bad Guy, Levitating, Sunflower, Perfect, Old Town Road, Shape of You, NA

3. Fill in the final summary table:

```
# A tibble: 2 x 2
  viral_status  count
  <chr>         <int>
1 Not yet viral     1
2 Viral hit         6
```

Part 2 — Write the Code Yourself

In Part 1, you were given the code and had to work out the output by hand. Now it's reversed: you'll see the **input** data frame and the **output** data frame, and your job is to write the dplyr code that turns one into the other. There can be more than one correct way to write some of these — as long as your code produces the same output, it's right.

The Data

This is the data set for Part 2: `movies`, 10 rows and 5 columns.

```
# A tibble: 10 x 5
   title          genre     release_year runtime_min box_office_millions
   <chr>          <chr>            <dbl>       <dbl>                <dbl>
 1 Coco           Animation         2017         105                   72
 2 Moana          Animation         2016         107                   84
 3 Up             Animation         2009          96                   68
 4 Zootopia       Family            2016         108                   91
 5 Encanto        Animation         2021         102                   58
 6 Toy Story 4    Family            2019         100                   76
 7 Inside Out     Family            2015          95                   65
 8 Incredibles 2  Adventure         2018         118                   88
 9 Finding Dory   Adventure         2016          97                   62
10 Soul           <NA>              2020         100                   45
```

Exercise 11

Output: (only the column names changed — everything else is identical)

```
# A tibble: 10 x 5
   title         movie_genre release_year runtime_min box_office_m
   <chr>         <chr>              <dbl>       <dbl>        <dbl>
 1 Coco          Animation           2017         105           72
 2 Moana         Animation           2016         107           84
 3 Up            Animation           2009          96           68
 4 Zootopia      Family              2016         108           91
 5 Encanto       Animation           2021         102           58
 6 Toy Story 4   Family              2019         100           76
 7 Inside Out    Family              2015          95           65
 8 Incredibles 2 Adventure           2018         118           88
 9 Finding Dory  Adventure           2016          97           62
10 Soul          <NA>                2020         100           45
```

Write the code that produces this output from `movies`.

```
movies |>
  rename(movie_genre = genre, box_office_m = box_office_millions)
```

Exercise 12

Output:

```
# A tibble: 10 x 3
   title         release_year box_office_millions
   <chr>                <dbl>               <dbl>
 1 Coco                  2017                  72
 2 Moana                 2016                  84
 3 Up                    2009                  68
 4 Zootopia              2016                  91
 5 Encanto               2021                  58
 6 Toy Story 4           2019                  76
 7 Inside Out            2015                  65
 8 Incredibles 2         2018                  88
 9 Finding Dory          2016                  62
10 Soul                  2020                  45
```

Write the code that produces this output from `movies`.

```
movies |>
  select(title, release_year, box_office_millions)
```

Exercise 13

Output:

```
# A tibble: 4 x 5
  title         genre     release_year runtime_min box_office_millions
  <chr>         <chr>            <dbl>       <dbl>                <dbl>
1 Encanto       Animation         2021         102                   58
2 Toy Story 4   Family            2019         100                   76
3 Incredibles 2 Adventure         2018         118                   88
4 Soul          <NA>              2020         100                   45
```

Write the code that produces this output from `movies`.

```
movies |>
  filter(release_year >= 2018)
```

Exercise 14

Output:

```
# A tibble: 2 x 5
  title genre     release_year runtime_min box_office_millions
  <chr> <chr>            <dbl>       <dbl>               <dbl>
1 Coco  Animation         2017         105                  72
2 Moana Animation         2016         107                  84
```

Write the code that produces this output from `movies`. (Hint: two conditions have to be true at once.)

```
movies |>
  filter(genre == "Animation" & box_office_millions >= 70)
```

Exercise 15

Input:

```
# A tibble: 10 x 5
   title         genre     release_year runtime_min box_office_millions
   <chr>         <chr>            <dbl>       <dbl>                <dbl>
 1 Coco          Animation         2017         105                   72
 2 Moana         Animation         2016         107                   84
 3 Up            Animation         2009          96                   68
 4 Zootopia      Family            2016         108                   91
 5 Encanto       Animation         2021         102                   58
 6 Toy Story 4   Family            2019         100                   76
 7 Inside Out    Family            2015          95                   65
 8 Incredibles 2 Adventure         2018         118                   88
 9 Finding Dory  Adventure         2016          97                   62
10 Soul          <NA>              2020         100                   45
```

Output:

```
# A tibble: 4 x 5
  title        genre     release_year runtime_min box_office_millions
  <chr>        <chr>            <dbl>       <dbl>                <dbl>
1 Moana        Animation         2016         107                   84
2 Zootopia     Family            2016         108                   91
3 Toy Story 4  Family            2019         100                   76
4 Finding Dory Adventure         2016          97                   62
```

Write the code that produces this output from `movies`. (Hint: you'll need a vector of years and the `%in%` operator.)

```
specific_years <- c(2016, 2019)
movies |>
  filter(release_year %in% specific_years)
```

Exercise 16

Input:

```
# A tibble: 10 x 5
   title         genre     release_year runtime_min box_office_millions
   <chr>         <chr>            <dbl>       <dbl>                <dbl>
 1 Coco          Animation         2017         105                   72
 2 Moana         Animation         2016         107                   84
 3 Up            Animation         2009          96                   68
 4 Zootopia      Family            2016         108                   91
 5 Encanto       Animation         2021         102                   58
 6 Toy Story 4   Family            2019         100                   76
 7 Inside Out    Family            2015          95                   65
 8 Incredibles 2 Adventure         2018         118                   88
 9 Finding Dory  Adventure         2016          97                   62
10 Soul          <NA>              2020         100                   45
```

Output:

```
# A tibble: 1 x 5
  title genre release_year runtime_min box_office_millions
  <chr> <chr>        <dbl>       <dbl>                <dbl>
1 Soul  <NA>          2020         100                   45
```

Write the code that produces this output from `movies` while using `is.na()`.

```
movies |>
  filter(is.na(genre))
```

Exercise 17

Output:

```
# A tibble: 10 x 3
   title         box_office_millions hit_status
   <chr>                       <dbl> <chr>
 1 Coco                           72 Box-office hit
 2 Moana                          84 Box-office hit
 3 Up                             68 Modest run
 4 Zootopia                       91 Box-office hit
 5 Encanto                        58 Modest run
 6 Toy Story 4                    76 Box-office hit
 7 Inside Out                     65 Modest run
 8 Incredibles 2                  88 Box-office hit
 9 Finding Dory                   62 Modest run
10 Soul                           45 Modest run
```

Write the code that produces this output. (You're adding one new column, `hit_status`, based on a condition.)

```
movies |>
  select(title, box_office_millions) |>
  mutate(
    hit_status = if_else(
      box_office_millions >= 70,
      "Box-office hit",
      "Modest run"
    )
  )
```

Exercise 18

Output:

```
# A tibble: 10 x 3
   title         box_office_millions revenue_tier
   <chr>                       <dbl> <chr>
 1 Coco                           72 $50M to $79M
 2 Moana                          84 $80M or more
 3 Up                             68 $50M to $79M
 4 Zootopia                       91 $80M or more
 5 Encanto                        58 $50M to $79M
 6 Toy Story 4                    76 $50M to $79M
 7 Inside Out                     65 $50M to $79M
 8 Incredibles 2                  88 $80M or more
 9 Finding Dory                   62 $50M to $79M
10 Soul                           45 Under $50M
```

Write the code that produces this output. (You're adding one new column, `revenue_tier`, with three possible values.)

```
movies |>
  select(title, box_office_millions) |>
  mutate(
    revenue_tier = case_when(
      box_office_millions < 50 ~ "Under $50M",
      box_office_millions < 80 ~ "$50M to $79M",
      box_office_millions >= 80 ~ "$80M or more"
    )
  )
```

Exercise 19

Output:

```
# A tibble: 4 x 3
  genre     avg_runtime count
  <chr>           <dbl> <int>
1 Adventure        108.     2
2 Animation        102.     4
3 Family           101      3
4 <NA>             100      1
```

Write the code that produces this output. (You're grouping by one column and calculating two summary values per group.)

```
movies |>
  group_by(genre) |>
  summarize(
    avg_runtime = mean(runtime_min),
    count = n()
  )
```

Exercise 20

Output:

```
# A tibble: 2 x 2
  hit_status     count
  <chr>          <int>
1 Box-office hit     4
2 Modest run         3
```

Write the full chain of code — filter, then mutate, then group and summarize — that turns the input into this output. (Hint: check how many rows disappeared in the `filter()` step, and think about why. Does `!=` keep or drop a missing `genre` value?)

```
movies |>
  filter(genre != "Adventure") |>
  mutate(
    hit_status = if_else(
      box_office_millions >= 70,
      "Box-office hit",
      "Modest run"
    )
  ) |>
  group_by(hit_status) |>
  summarize(count = n())
```