INTRO TO R – DATA WRANGLING

R Workshop - 2

February 17, 2021 Prepared by Jay Cao / TDMDAL

Website: https://tdmdal.github.io/r-tutorial-201920-winter/

Rotman School of Management
UNIVERSITY OF TORONTO

Plan

“Data Scientists spend up to 80% of the time on data cleaning and 20 percent of their time on actual data analysis”

-- ?????

- Tidy Data (a way to organize data)
- Data manipulation (a continuation)
  - `summarise()` and `group_by()`
  - `_join()` datasets
- Again, we will focus on basics, underlying principles, and best practices

Tidy Data – Motivation (FANG Revenue Data)

- Q1. Is the below table organized well for easy analysis?
- Q2. In general, what's a good way to organize/structure data?

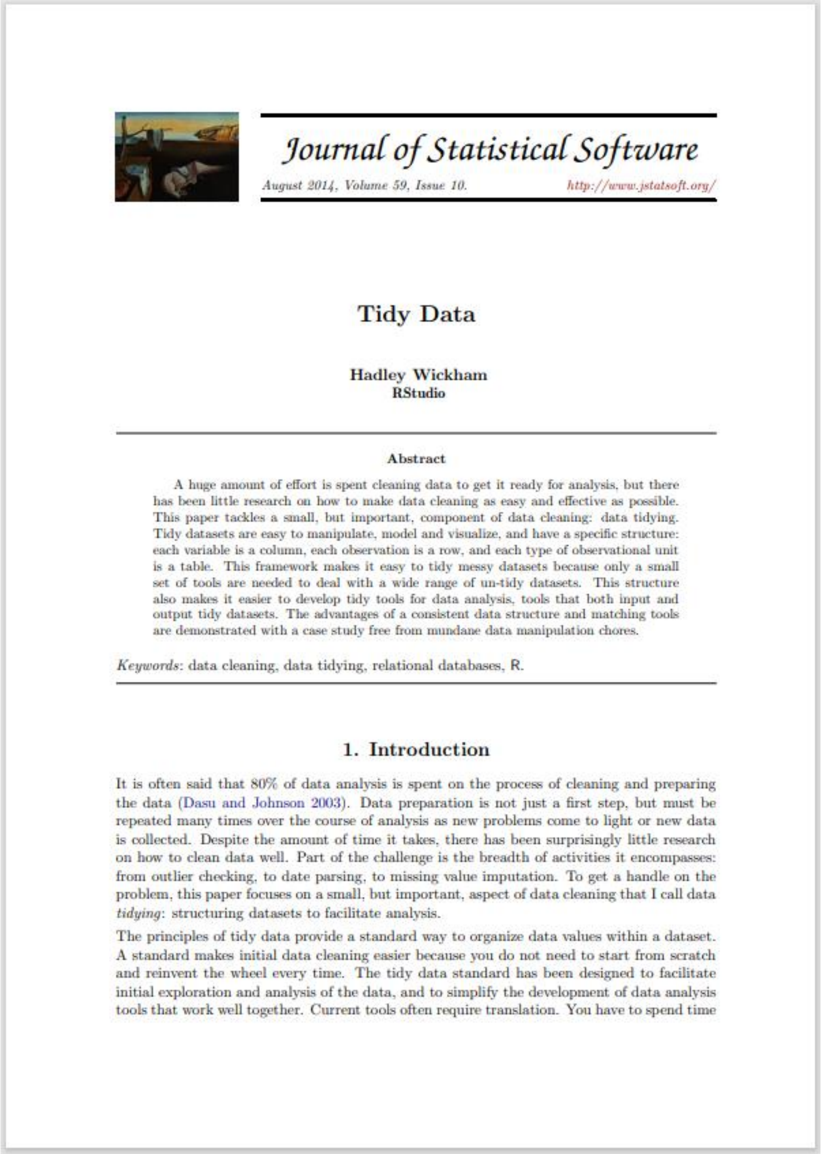
```
# A tibble: 4 x 6
  Ticker `Fiscal Year`    Q1    Q2    Q3    Q4
  <chr>          <int> <dbl> <dbl> <dbl> <dbl>
1 AMZN            2018 51.0  52.9  56.6  72.4 
2 FB              2018 12.0  13.2  13.7  16.9 
3 GOOG            2018 31.1  32.7  33.7  39.3 
4 NFLX            2018  3.70  3.91  4.00  4.19
```

Tidy Data

- A (**One**) way to organize **tabular** data

- Definition
  - Each **variable** forms a **column**.
  - Each **observation**, or **case**, forms a **row**.
  - Each **type of observational unit** forms a **table**

- Why tidy data
  - A great way to organize data for maintainability
  - Once in tidy data, it's easy to use the toolset from *tidyverse* to manipulate them



Journal of Statistical Software
August 2014, Volume 59, Issue 10. http://www.jstatsoft.org/

Tidy Data

Hadley Wickham
RStudio

Abstract

A huge amount of effort is spent cleaning data to get it ready for analysis, but there has been little research on how to make data cleaning as easy and effective as possible. This paper tackles a small, but important, component of data cleaning: data tidying. Tidy datasets are easy to manipulate, model and visualize, and have a specific structure: each variable is a column, each observation is a row, and each type of observational unit is a table. This framework makes it easy to tidy messy datasets because only a small set of tools are needed to deal with a wide range of un-tidy datasets. This structure also makes it easier to develop tidy tools for data analysis, tools that both input and output tidy datasets. The advantages of a consistent data structure and matching tools are demonstrated with a case study free from mundane data manipulation chores.

Keywords: data cleaning, data tidying, relational databases, R.

1. Introduction

It is often said that 80% of data analysis is spent on the process of cleaning and preparing the data (Dasu and Johnson 2003). Data preparation is not just a first step, but must be repeated many times over the course of analysis as new problems come to light or new data is collected. Despite the amount of time it takes, there has been surprisingly little research on how to clean data well. Part of the challenge is the breadth of activities it encompasses: from outlier checking, to date parsing, to missing value imputation. To get a handle on the problem, this paper focuses on a small, but important, aspect of data cleaning that I call data *tidying*: structuring datasets to facilitate analysis.

The principles of tidy data provide a standard way to organize data values within a dataset. A standard makes initial data cleaning easier because you do not need to start from scratch and reinvent the wheel every time. The tidy data standard has been designed to facilitate initial exploration and analysis of the data, and to simplify the development of data analysis tools that work well together. Current tools often require translation. You have to spend time

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1 / Why is it Messy?

- Values as column names
- Hard to retrieve data and analyze them in a consistent way
  - **how many treatments in total**
  - get average result by person
  - get average result by treatment
  - get overall average result

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1 / Why is it Messy?

- Values as column names
- Hard to retrieve data and analyze them in a consistent way
  - how many treatments in total
  - **get average result by person**
  - get average result by treatment
  - get overall average result

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1 / Why is it Messy?

- Values as column names
- Hard to retrieve data and analyze them in a consistent way
  - how many treatments in total
  - get average result by person
  - **get average result by treatment**
  - get overall average result

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 1 / Why is it Messy?

- Values as column names
- Hard to retrieve data and analyze them in a consistent way
  - how many treatments in total
  - get average result by person
  - get average result by treatment
  - **get overall average result**

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 2

	John Smith	Jane Doe	Mary Johnson
treatmenta	—	16	3
treatmentb	2	11	1

Table 2: The same data as in Table 1 but structured differently.

http://vita.had.co.nz/papers/tidy-data.html

The Tidy Version

name	trt	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

http://vita.had.co.nz/papers/tidy-data.html

The Tidy Version – Why is it Tidy

- All column-wise operations
  - how many treatments in total
  - get average result by person
  - get average result by treatment
  - get overall average result

name	trt	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

http://vita.had.co.nz/papers/tidy-data.html

Back to the FANG Revenue Data

```
# A tibble: 4 x 6
  Ticker `Fiscal Year`    Q1    Q2    Q3    Q4
  <chr>          <int> <dbl> <dbl> <dbl> <dbl>
1 AMZN            2018 51.0  52.9  56.6  72.4 
2 FB              2018 12.0  13.2  13.7  16.9 
3 GOOG            2018 31.1  32.7  33.7  39.3 
4 NFLX            2018  3.70  3.91  4.00  4.19
```

From Messy to Tidy (One Example)

	treatmenta	treatmentb
John Smith	—	2
Jane Doe	16	11
Mary Johnson	3	1

Table 1: Typical presentation dataset.

name	trt	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

pivot_longer()

```
# A tibble: 3 x 3
  name         treatmenta treatmentb
  <chr>             <dbl>      <dbl>
1 John Smith           NA          2
2 Jane Doe             16         11
3 Mary Johnson          3          1

pivot_longer(df_messy, -name,
             names_to = "treatment", values_to = "result")
```

https://tidyr.tidyverse.org/articles/pivot.html

pivot_longer()

```
# A tibble: 3 x 3
  name          treatmenta treatmentb
  <chr>              <dbl>      <dbl>
1 John Smith            NA          2
2 Jane Doe              16         11
3 Mary Johnson           3          1

pivot_longer(df_messy, -name,
             names_to = "treatment", values_to = "result")
```

https://tidyr.tidyverse.org/articles/pivot.html

pivot_longer()

```
# A tibble: 3 x 3
  name         treatmenta treatmentb
  <chr>             <dbl>      <dbl>
1 John Smith           NA          2
2 Jane Doe             16         11
3 Mary Johnson          3          1

pivot_longer(df_messy, -name,
             names_to = "treatment", values_to = "result")
```

https://tidyr.tidyverse.org/articles/pivot.html

pivot_longer()

```
# A tibble: 3 x 3
  name          treatmenta treatmentb
  <chr>              <dbl>      <dbl>
1 John Smith            NA          2
2 Jane Doe              16         11
3 Mary Johnson           3          1

pivot_longer(df_messy, -name,
             names_to = "treatment", values_to = "result")
```

https://tidyr.tidyverse.org/articles/pivot.html

pivot_longer()

```
# A tibble: 3 x 3
  name          treatmenta treatmentb
  <chr>              <dbl>      <dbl>
1 John Smith            NA          2
2 Jane Doe              16         11
3 Mary Johnson           3          1

pivot_longer(df_messy, -name,
             names_to = "treatment", values_to = "result")
```

https://tidyr.tidyverse.org/articles/pivot.html

pivot_longer() result

```
# A tibble: 6 x 3
  name         treatment  result
  <chr>        <chr>       <dbl>
1 John Smith   treatmenta     NA
2 John Smith   treatmentb      2
3 Jane Doe     treatmenta     16
4 Jane Doe     treatmentb     11
5 Mary Johnson treatmenta      3
6 Mary Johnson treatmentb      1
```

https://tidyr.tidyverse.org/articles/pivot.html

The inverse transformation: `pivot_wider()`

```
  name         treatment result
  <chr>        <chr>      <dbl>
1 John Smith   a             NA
2 Jane Doe     a             16
3 Mary Johnson a              3
4 John Smith   b              2
5 Jane Doe     b             11
6 Mary Johnson b              1

pivot_wider(df_tidy,
            names_from = treatment, values_from = result)
```

https://tidyr.tidyverse.org/articles/pivot.html

The inverse transformation: pivot_wider()

```
  name         treatment result
  <chr>        <chr>      <dbl>
1 John Smith   a             NA
2 Jane Doe     a             16
3 Mary Johnson a              3
4 John Smith   b              2
5 Jane Doe     b             11
6 Mary Johnson b              1

pivot_wider(df_tidy,
            names_from = treatment, values_from = result)
```

https://tidyr.tidyverse.org/articles/pivot.html

The inverse transformation: `pivot_wider()`

```
  name         treatment result
  <chr>        <chr>      <dbl>
1 John Smith   a             NA
2 Jane Doe     a             16
3 Mary Johnson a              3
4 John Smith   b              2
5 Jane Doe     b             11
6 Mary Johnson b              1

pivot_wider(df_tidy,
            names_from = treatment, values_from = result)
```

https://tidyr.tidyverse.org/articles/pivot.html

`pivot_wider()` result

```
# A tibble: 3 x 3
  name               a     b
  <chr>          <dbl> <dbl>
1 John Smith        NA     2
2 Jane Doe          16    11
3 Mary Johnson       3     1
```

https://tidyr.tidyverse.org/articles/pivot.html

Try Yourself: "Tidy up" the FANG revenue data

```
# A tibble: 16 x 4
   Ticker `Fiscal Year` Quarter Revenue
   <chr>          <int> <chr>     <dbl>
 1 AMZN            2018 Q1         51.0
 2 AMZN            2018 Q2         52.9
 3 AMZN            2018 Q3         56.6
 4 AMZN            2018 Q4         72.4
 5 FB              2018 Q1         12.0
 6 FB              2018 Q2         13.2
```

…

Load the raw "long" data from this URL: https://tdmdal.github.io/r-tutorial-202021-winter/data/fang_2018.csv

Many Ways of Being Messy :(

- Messy datasets have 5 common problems (Wickham, 2014)
    1. Column headers are values, not variable names.
    2. Multiple variables are stored in one column.
    3. Variables are stored in both rows and columns.
    4. Multiple types of observational units are stored in the same table.
    5. A single observational unit is stored in multiple tables.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 3

year	artist	time	track	date	week	rank
2000	2 Pac	4:22	Baby Don't Cry	2000-02-26	1	87
2000	2 Pac	4:22	Baby Don't Cry	2000-03-04	2	82
2000	2 Pac	4:22	Baby Don't Cry	2000-03-11	3	72
2000	2 Pac	4:22	Baby Don't Cry	2000-03-18	4	77
2000	2 Pac	4:22	Baby Don't Cry	2000-03-25	5	87
2000	2 Pac	4:22	Baby Don't Cry	2000-04-01	6	94
2000	2 Pac	4:22	Baby Don't Cry	2000-04-08	7	99
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-02	1	91
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-09	2	87
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-16	3	92
2000	3 Doors Down	3:53	Kryptonite	2000-04-08	1	81
2000	3 Doors Down	3:53	Kryptonite	2000-04-15	2	70
2000	3 Doors Down	3:53	Kryptonite	2000-04-22	3	68
2000	3 Doors Down	3:53	Kryptonite	2000-04-29	4	67
2000	3 Doors Down	3:53	Kryptonite	2000-05-06	5	66

Table 8: First fifteen rows of the tidied billboard dataset. The `date` column does not appear in the original table, but can be computed from `date.entered` and `week`.

http://vita.had.co.nz/papers/tidy-data.html

Messy Data – Example 3

year	artist	time	track	date	week	rank
2000	2 Pac	4:22	Baby Don't Cry	2000-02-26	1	87
2000	2 Pac	4:22	Baby Don't Cry	2000-03-04	2	82
2000	2 Pac	4:22	Baby Don't Cry	2000-03-11	3	72
2000	2 Pac	4:22	Baby Don't Cry	2000-03-18	4	77
2000	2 Pac	4:22	Baby Don't Cry	2000-03-25	5	87
2000	2 Pac	4:22	Baby Don't Cry	2000-04-01	6	94
2000	2 Pac	4:22	Baby Don't Cry	2000-04-08	7	99
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-02	1	91
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-09	2	87
2000	2Ge+her	3:15	The Hardest Part Of ...	2000-09-16	3	92
2000	3 Doors Down	3:53	Kryptonite	2000-04-08	1	81
2000	3 Doors Down	3:53	Kryptonite	2000-04-15	2	70
2000	3 Doors Down	3:53	Kryptonite	2000-04-22	3	68
2000	3 Doors Down	3:53	Kryptonite	2000-04-29	4	67
2000	3 Doors Down	3:53	Kryptonite	2000-05-06	5	66

Table 8: First fifteen rows of the tidied billboard dataset. The `date` column does not appear in the original table, but can be computed from `date.entered` and `week`.

http://vita.had.co.nz/papers/tidy-data.html

The Tidy Version

id	artist	track	time
1	2 Pac	Baby Don't Cry	4:22
2	2Ge+her	The Hardest Part Of ...	3:15
3	3 Doors Down	Kryptonite	3:53
4	3 Doors Down	Loser	4:24
5	504 Boyz	Wobble Wobble	3:35
6	98^0	Give Me Just One Nig...	3:24
7	A*Teens	Dancing Queen	3:44
8	Aaliyah	I Don't Wanna	4:15
9	Aaliyah	Try Again	4:03
10	Adams, Yolanda	Open My Heart	5:30
11	Adkins, Trace	More	3:05
12	Aguilera, Christina	Come On Over Baby	3:38
13	Aguilera, Christina	I Turn To You	4:00
14	Aguilera, Christina	What A Girl Wants	3:18
15	Alice Deejay	Better Off Alone	6:50

id	date	rank
1	2000-02-26	87
1	2000-03-04	82
1	2000-03-11	72
1	2000-03-18	77
1	2000-03-25	87
1	2000-04-01	94
1	2000-04-08	99
2	2000-09-02	91
2	2000-09-09	87
2	2000-09-16	92
3	2000-04-08	81
3	2000-04-15	70
3	2000-04-22	68
3	2000-04-29	67
3	2000-05-06	66

Table 13: Normalised billboard dataset split up into song dataset (left) and rank dataset (right). First 15 rows of each dataset shown; `genre` omitted from song dataset, `week` omitted from rank dataset.

http://vita.had.co.nz/papers/tidy-data.html

Tidy Data and Its Eco-system

- Tidyverse
  - “an opinionated collection of R packages designed for data science”
  - “All packages share an underlying design philosophy, grammar, and data structures.”

- Other tools in the eco-system (especially for finance)
  - tidyquant, a package for quantitative finance
  - tidyvert, a set of tidy tools for time series
    - **Forecasting: Principles and Practice** (3ed), a free book using this toolset

Plan

"Data Scientists spend up to 80% of the time on data cleaning and 20 percent of their time on actual data analysis"

-- ?????

- Tidy Data
- Data manipulation (a continuation)
  - `summarise() and group_by()`
  - `_join()` datasets
- Again, we will focus on basics, underlying principles, and best practices

Data manipulation: `dplyr()`

- Filter observations: `filter()`
- Select variables: `select()`
- Reorder rows: **`arrange()`**
- Create new variables: `mutate()`
- Collapse column values to a single summary: **`summarise()`**

- Group by: **`group_by()`**

The Employees Table

```
> employees %>% select(FirstName, LastName, Country)

# A tibble: 9 x 3
  FirstName LastName  Country
  <chr>     <chr>     <chr>
1 Nancy     Davolio   USA
2 Andrew    Fuller    USA
3 Janet     Leverling USA
4 Margaret  Peacock   USA
5 Steven    Buchanan  UK
…
```

Count Number of Employees By Country (1)

```
> employees %>% select(FirstName, LastName, Country) %>%
    group_by(Country)

# A tibble: 9 x 3
# Groups:   Country [2]
  FirstName LastName  Country
  <chr>     <chr>     <chr>
1 Nancy     Davolio   USA
2 Andrew    Fuller    USA
3 Janet     Leverling USA
…
```

Count Number of Employees By Country (2)

```
> employees %>% select(FirstName, LastName, Country) %>%
    group_by(Country) %>%
    summarise(count = n())

# A tibble: 2 x 2
  Country count
  <chr>   <int>
1 UK          4
2 USA         5
```

Count Number of Employees By Country (3)

```
> employees %>% select(FirstName, LastName, Country) %>%
    group_by(Country) %>%
    summarise(count = n()) %>%
    arrange(desc(count))

# A tibble: 2 x 2
  Country count
  <chr>   <int>
1 USA         5
2 UK          4
```

More on Data Aggregation

- `summarise()` works with many aggregation functions
  - Aggregation function: n input -> 1 output
  - e.g. `mean(), median(), min(), max(), first(), last(), n_distinct(),` …

- There are also Windows functions (useful for time series data)
  - Windows functions: n input -> n output
  - aggregation variations: `cumsum(), cummean(),`…
  - ranking and ordering: `rank(), percent_rank(),`…
  - offsets: `lead(), lag(),`…

https://dplyr.tidyverse.org/reference/summarise.html
https://dplyr.tidyverse.org/articles/window-functions.html

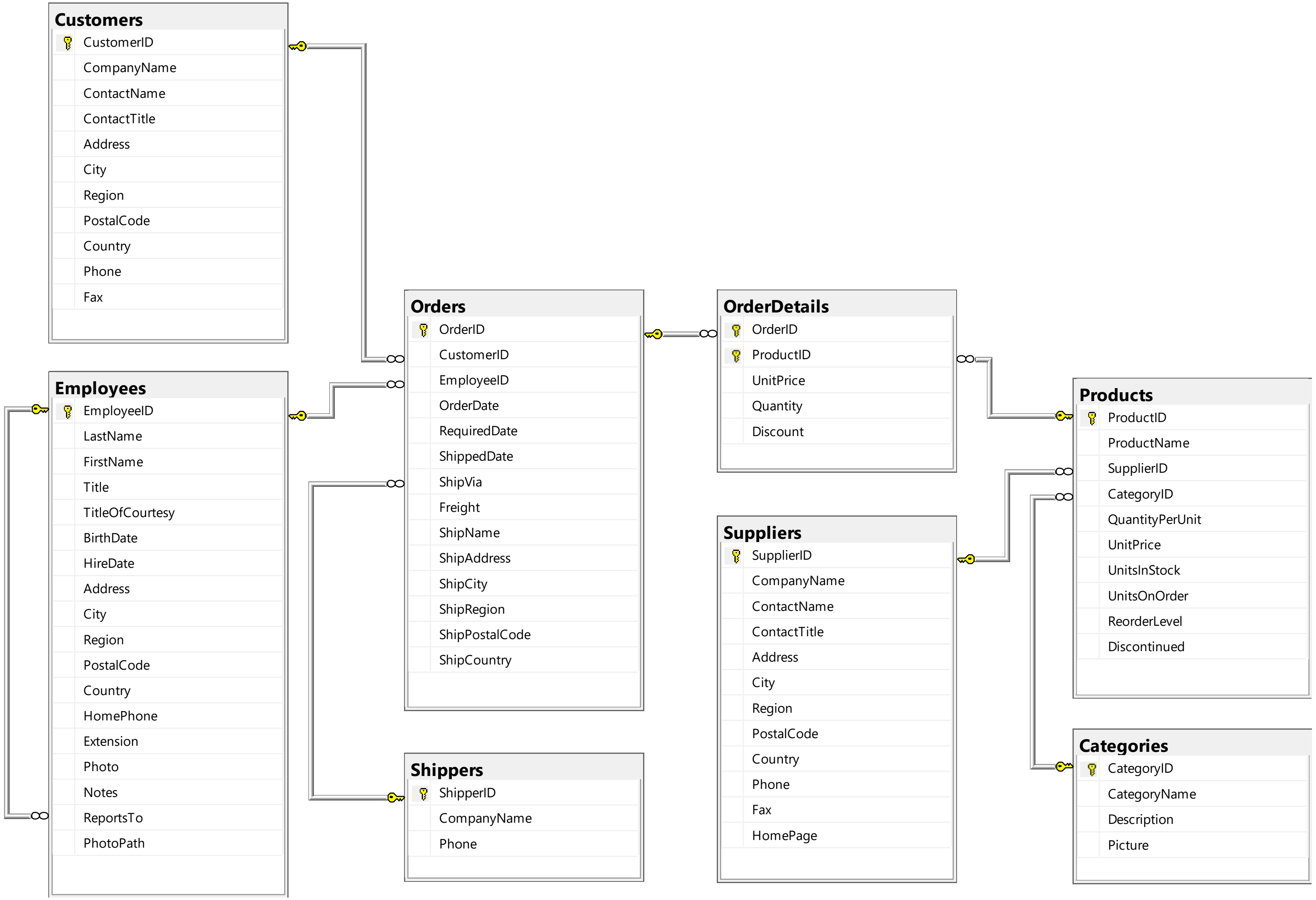
Plan

"Data Scientists spend up to 80% of the time on data cleaning and 20 percent of their time on actual data analysis"

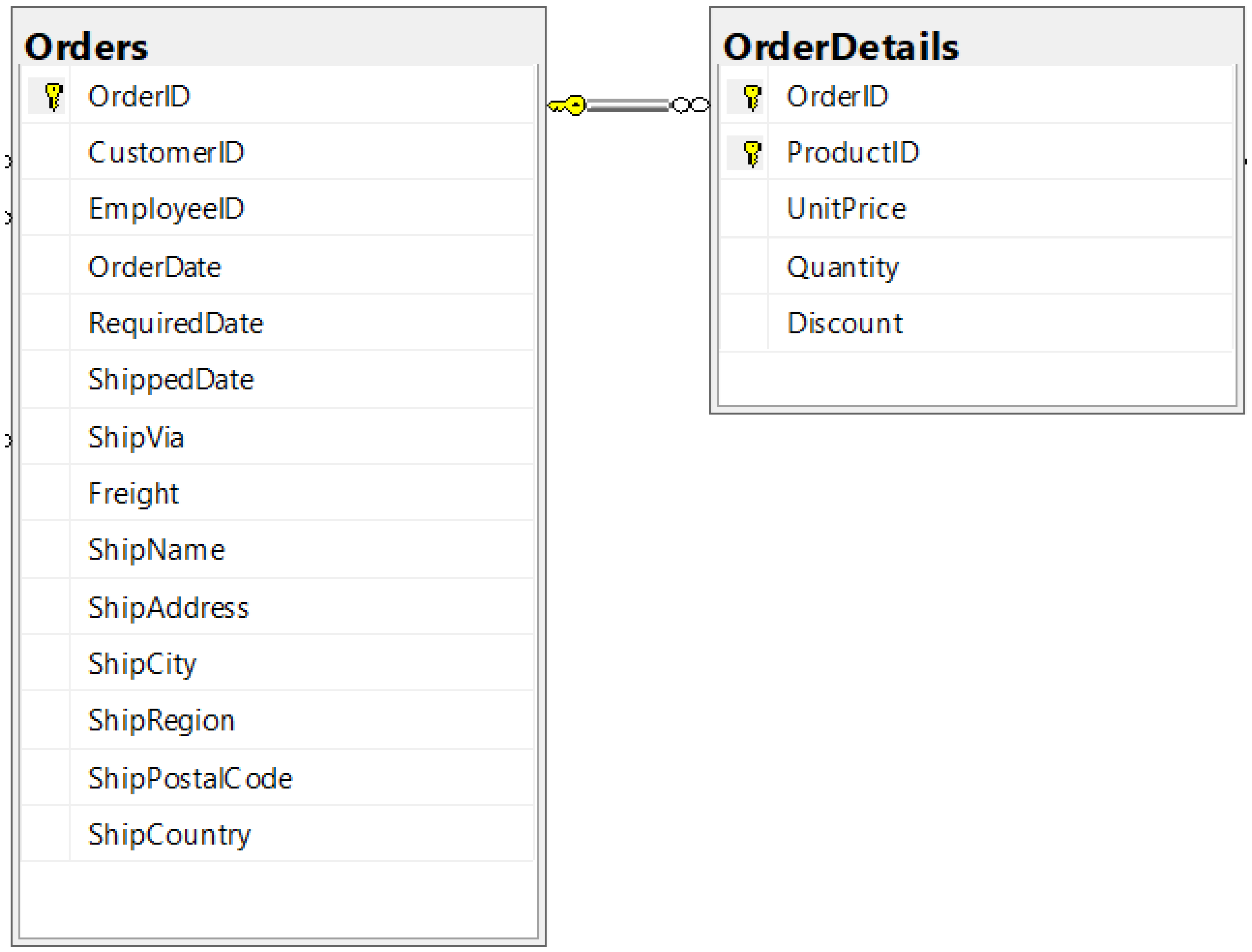
-- ?????

- Tidy Data
- Data manipulation (a continuation)
  - `summarise()` and `group_by()`
  - `_join()` datasets
- Again, we will focus on basics, underlying principles, and best practices

Motivation: Relation between Datasets/Tables

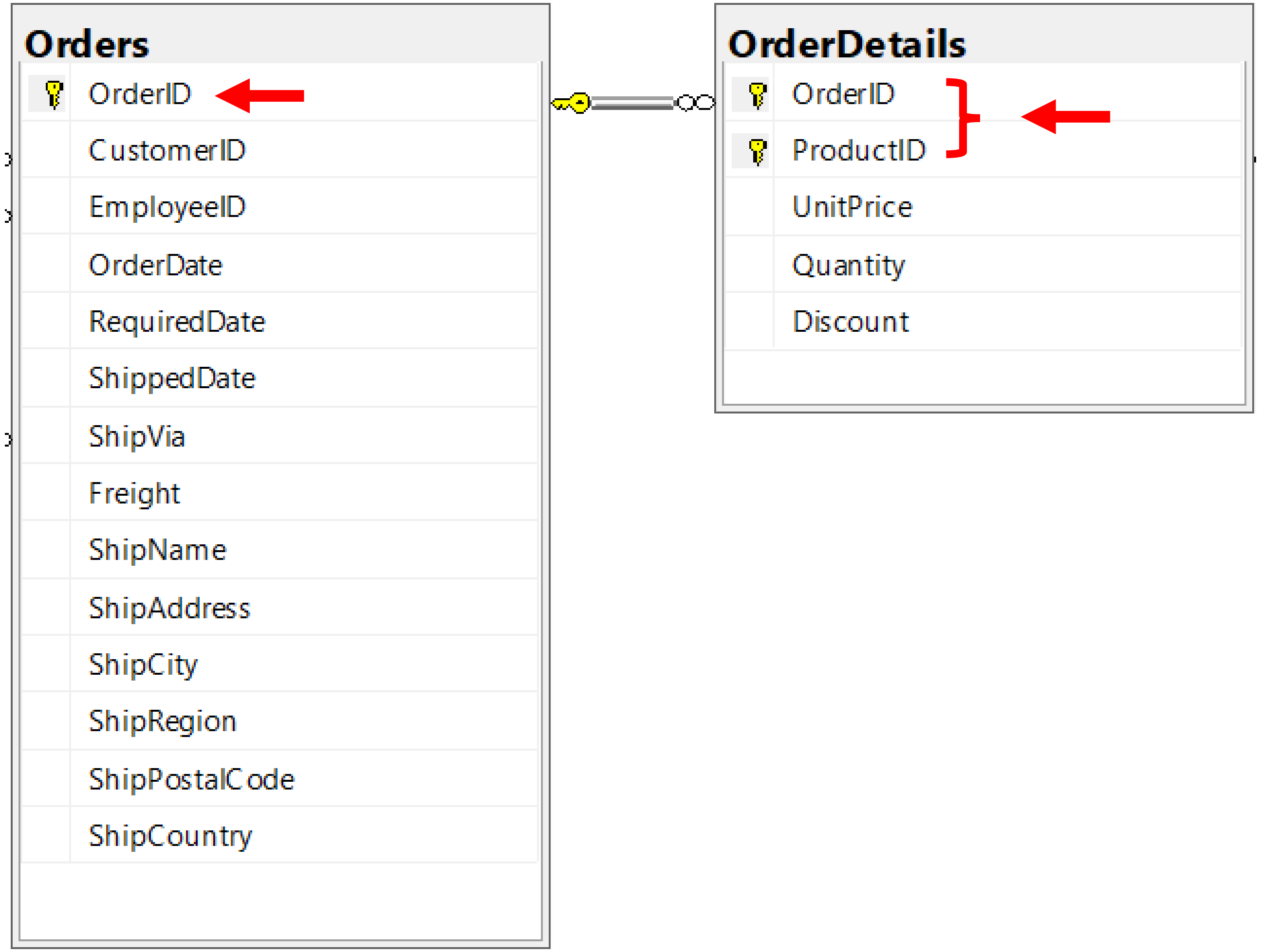


Relation between Datasets/Tables – Zoom In



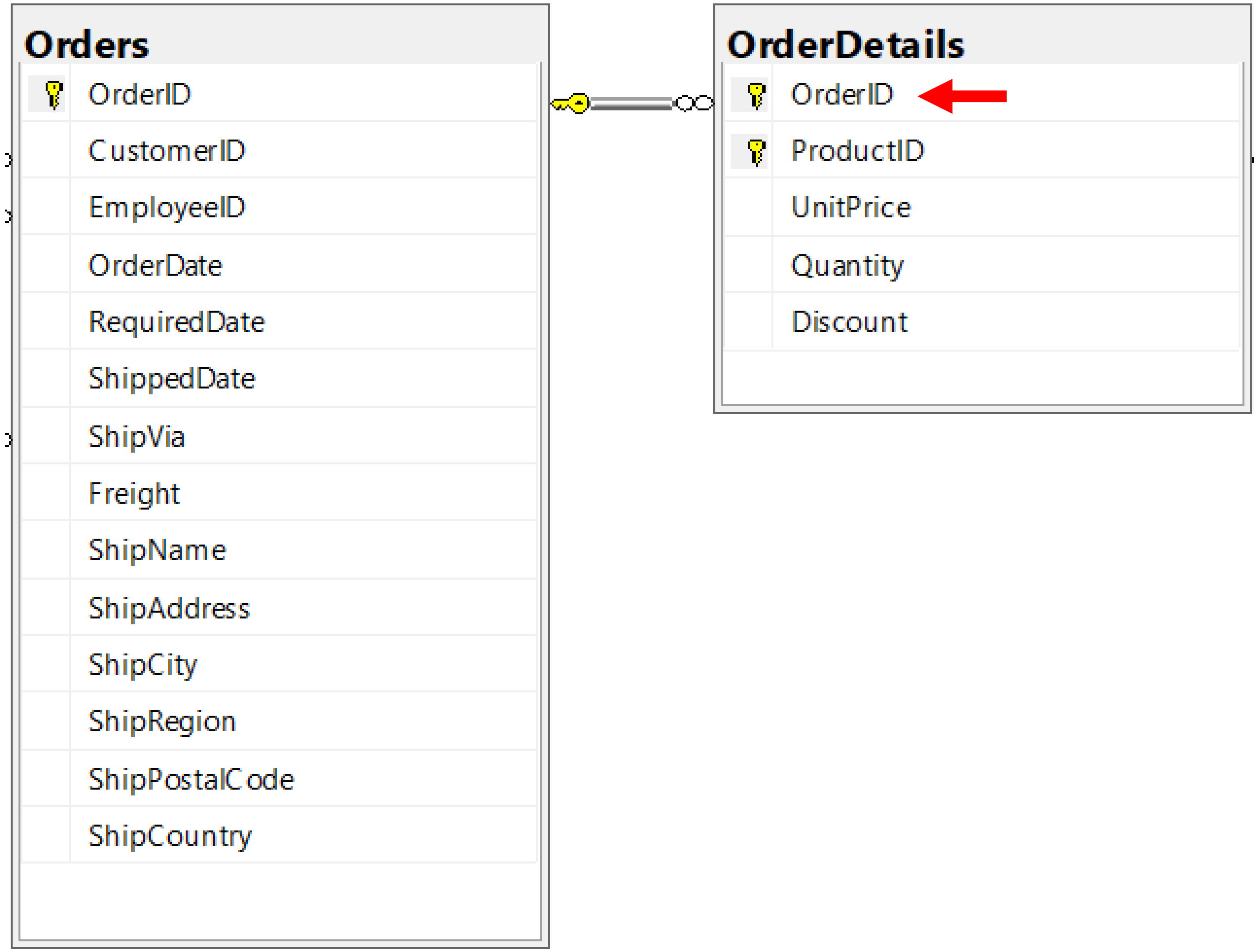
Relation between Datasets/Tables – Zoom In

- **Primary key**



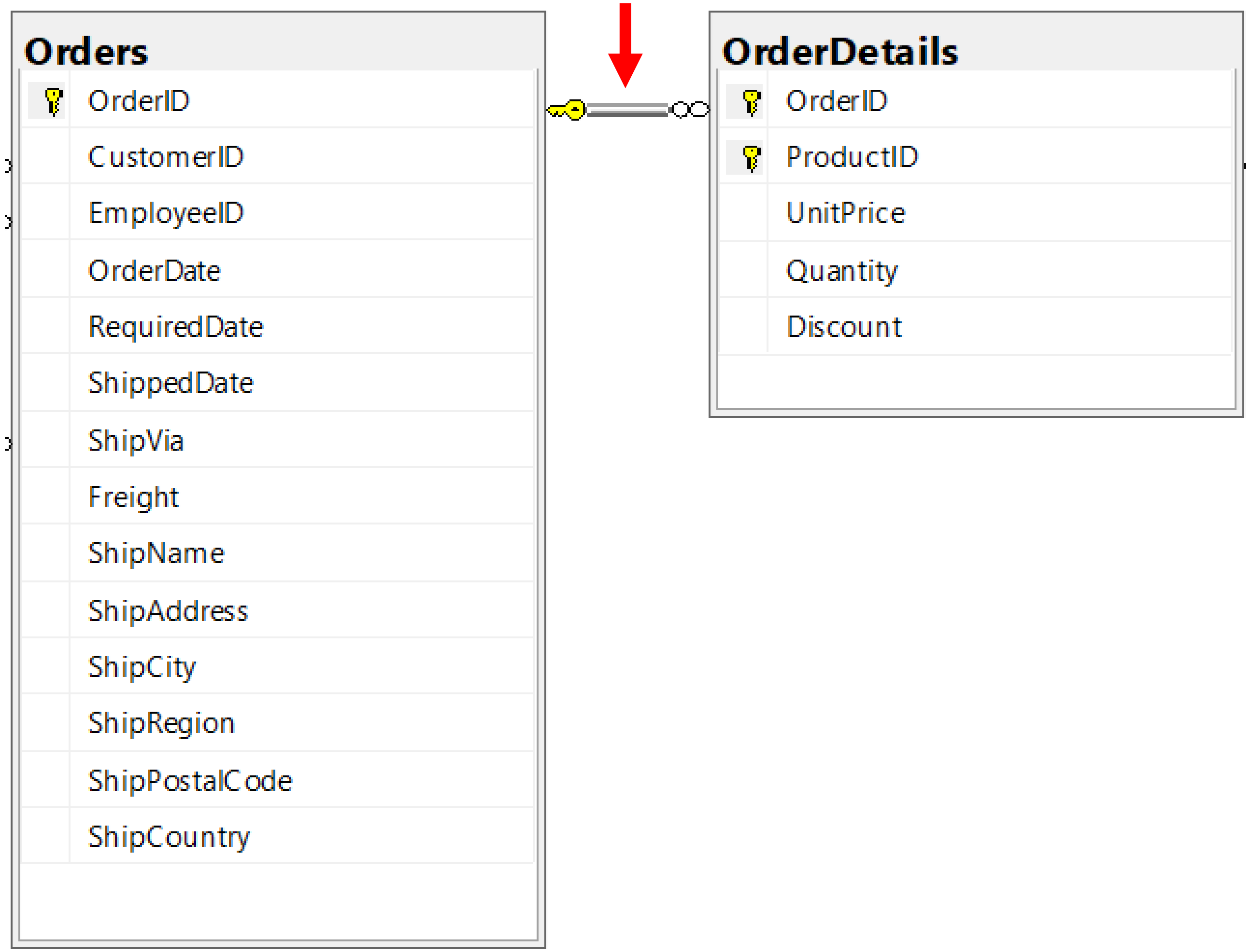
Relation between datasets/tables – Zoom In

- Primary key
- **Foreign key**

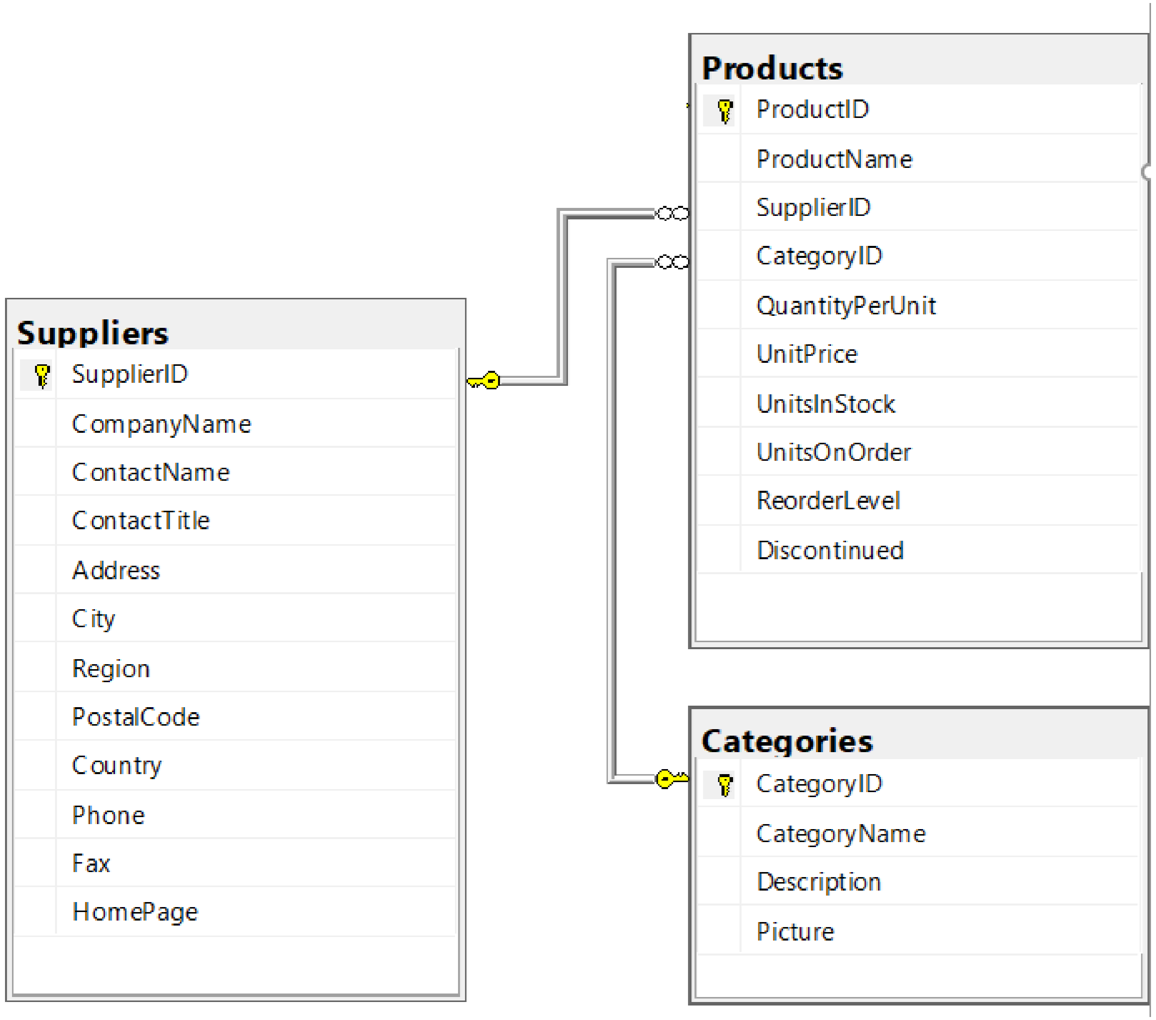


Relation between datasets/tables – Zoom In

- Primary key
- Foreign key
- **1-to-Many Relationship**



Relation between Tables – Another Example



Join – Inner Join

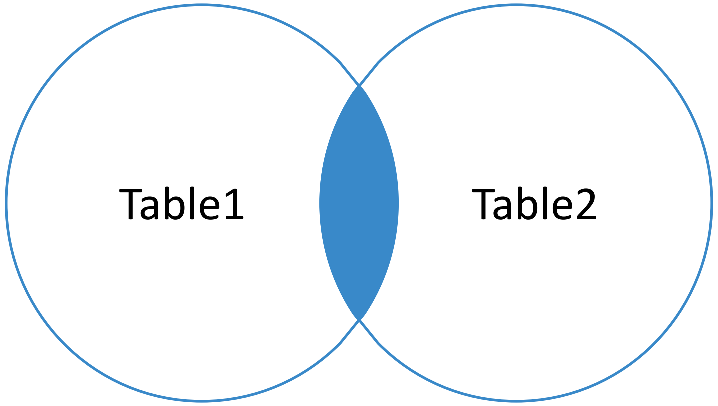


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
1	a	c
1	a	d

```
inner_join(Table1, Table2, by = c("pk" = "fk"))
```

Join – Left (Outer) Join

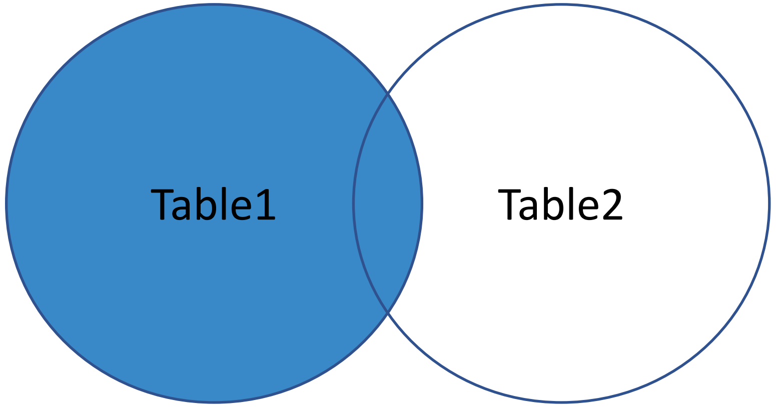


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
1	a	c
1	a	d
2	b	NA

```
left_join(Table1, Table2, by = c("pk" = "fk"))
```

Join - Left (Outer) Join With Exclusion

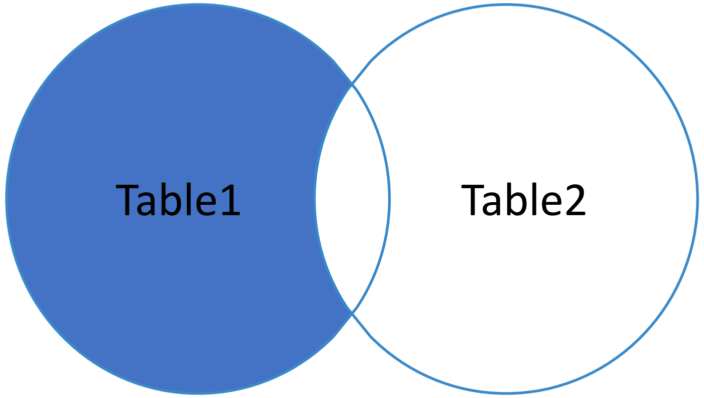


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
2	b	NA

```
Table1 %>%
  left_join(Table2, by = c("pk" = "fk")) %>%
  filter(is.na(t2c1))
```

More on Join Variations (learn them yourself)

- More join variation illustrations in the next few slides
  - right join, full join, …
- Read the “two-table verbs” vignette (in the *dplyr* package doc)
  - https://dplyr.tidyverse.org/articles/two-table.html
- Read the reference (see the “two table verbs” section)
  - https://dplyr.tidyverse.org/reference/index.html
- For data manipulation tasks in general
  - reading *dplyr* related articles are a good start, https://dplyr.tidyverse.org/articles/

Join – Right (Outer) Join*

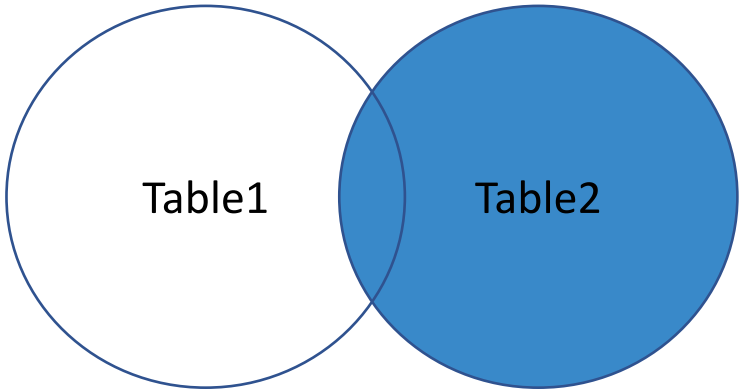


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
1	a	c
1	a	d
3	NA	e

```
right_join(Table1, Table2, by = c("pk" = "fk"))
```

Note: can use left_join as well.

Join - Right (Outer) Join With Exclusion*

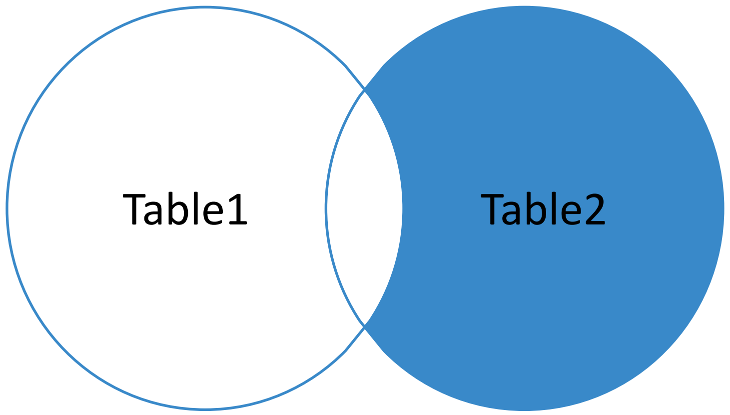


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
3	NA	e

```
Table1 %>%
  right_join(Table2, by = c("pk" = "fk")) %>%
  filter(is.na(t1c1))
```

Join – Full Outer Join

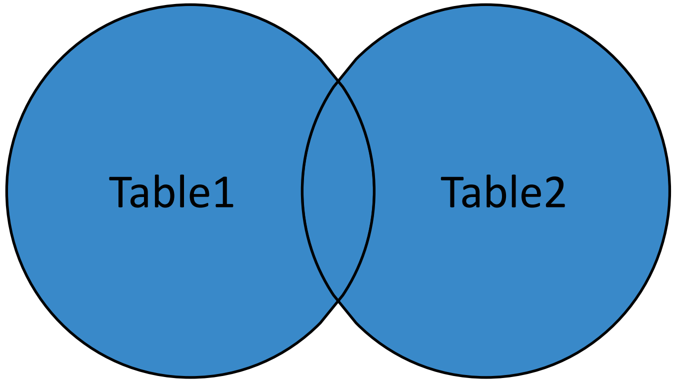


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```r
full_join(Table1, Table2, by = c("pk" = "fk"))
```

pk	t1c1	t2c1
1	a	c
1	a	d
2	b	NA
3	NA	e

Join – Full Outer Join With Exclusion

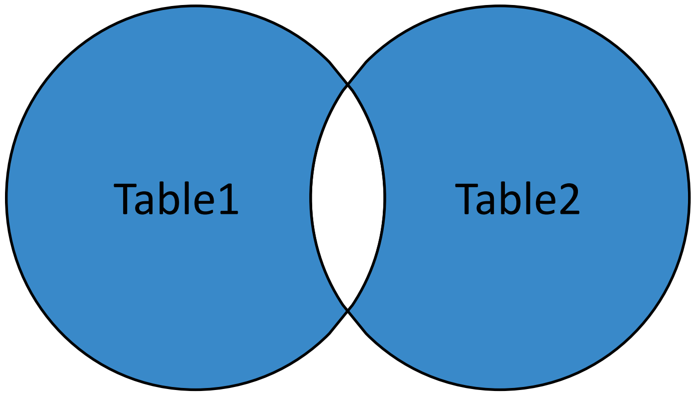


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	t2c1
2	b	NA
3	NA	e

```
Table1 %>%
  full_join(Table2, by = c("pk" = "fk")) %>%
  filter(is.na(t1c1) | is.na(t2c1))
```