Rotman

INTRO TO R – DATA WRANGLING

R Workshop - 2

February 3, 2020 Prepared by Jay Cao / TDMDAL

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

Rotman School of Management
UNIVERSITY OF TORONTO

Plan

- Tidy Data

- Data manipulation
  - Filter, Select, etc.
  - Join datasets

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**

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

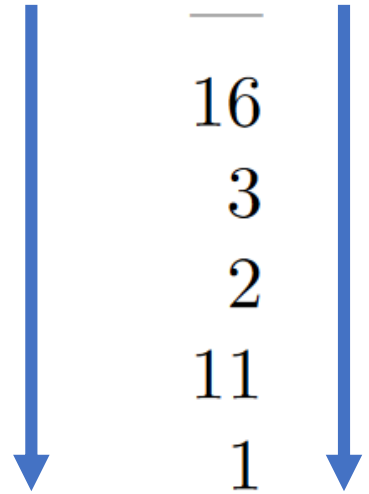
name	trt	result
John Smith	a	—
Jane Doe	a	16
Mary Johnson	a	3
John Smith	b	2
Jane Doe	b	11
Mary Johnson	b	1

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

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	98ˆ0	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

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")
```

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")
```

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")
```

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")
```

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")
```

`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
```

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)
```

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)
```

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)
```

`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
```

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

```
> 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
```

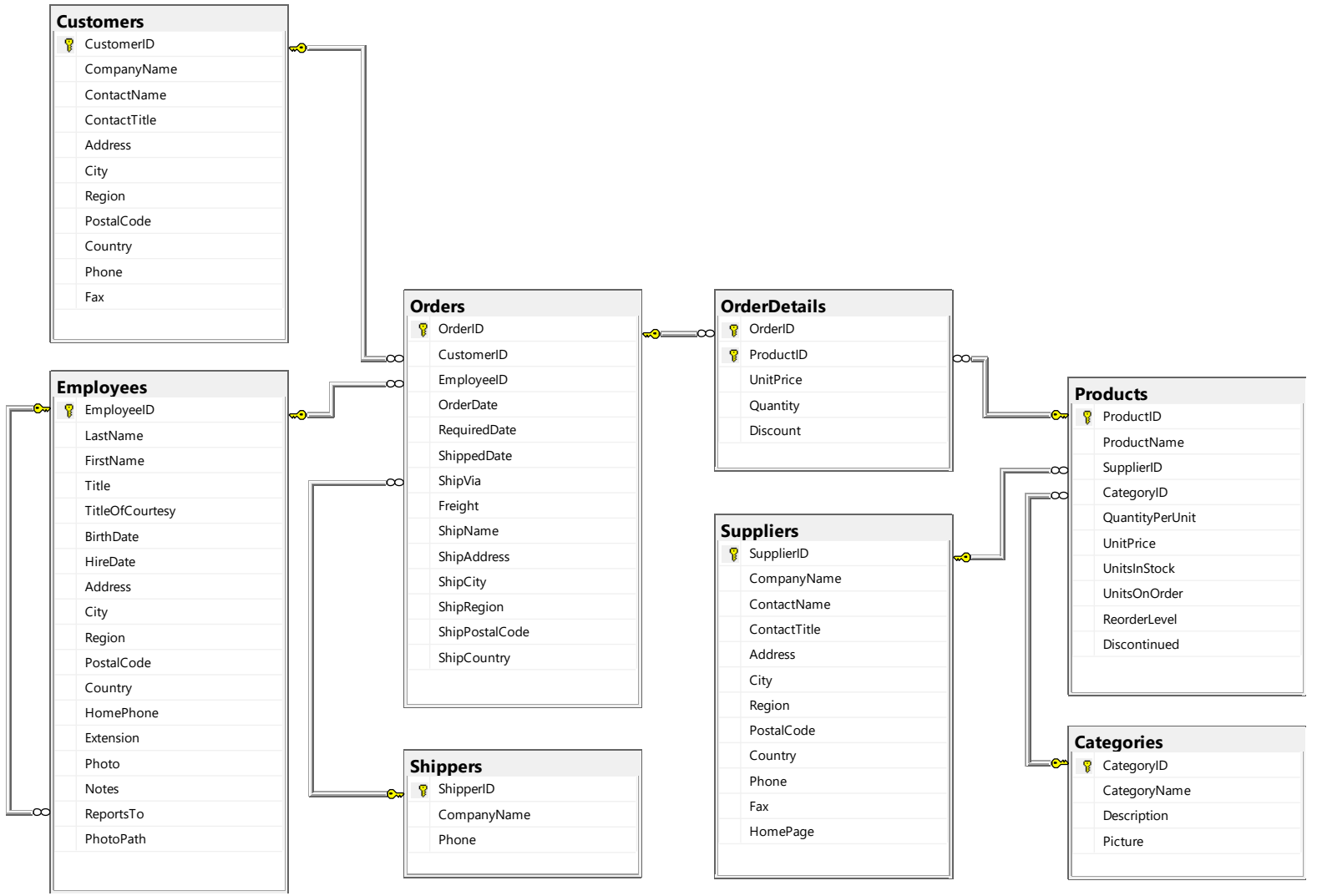
Count Number of Employees By Country

```
> 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
```

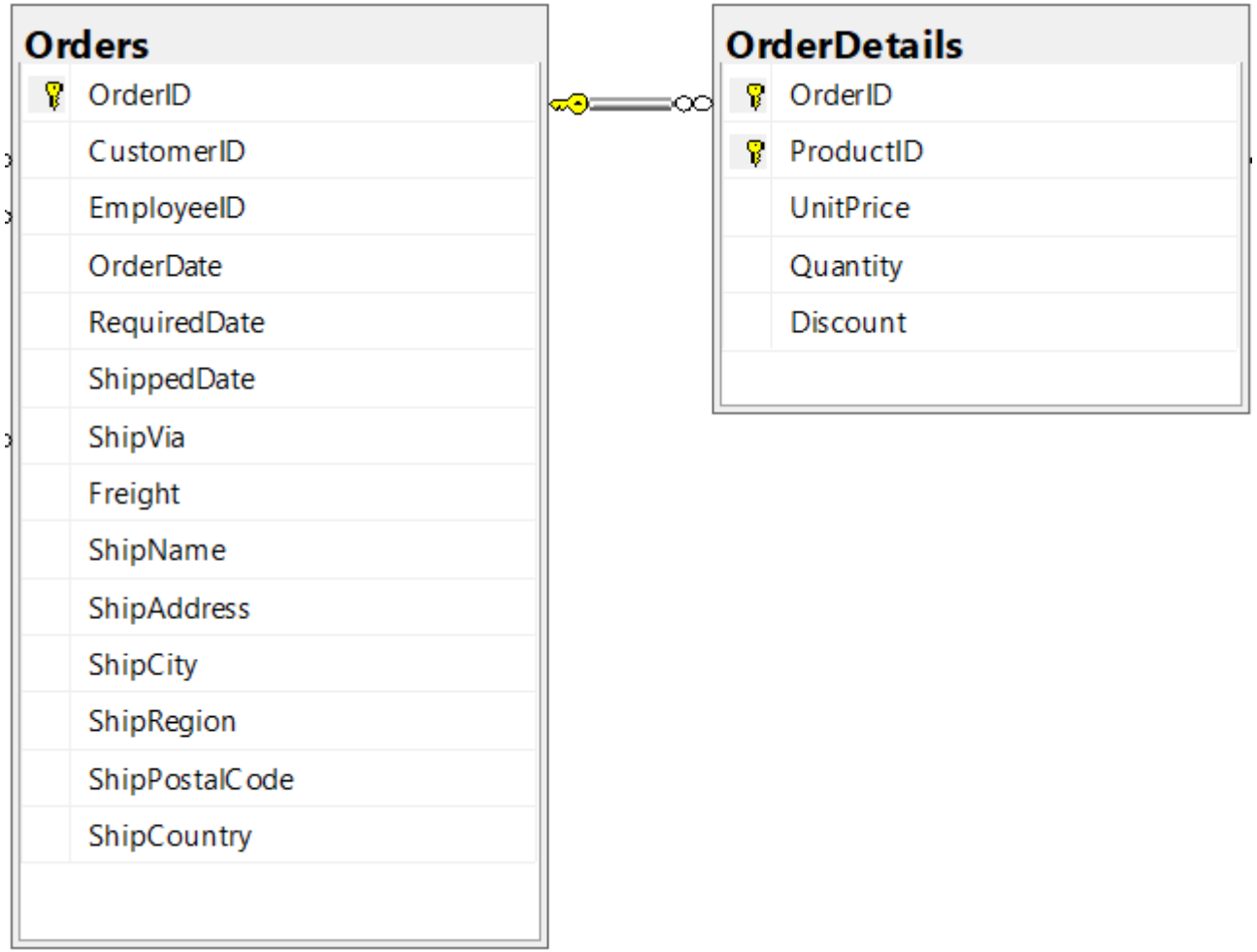
Count Number of Employees By Country

```
> 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
```

Relation between Datasets/Tables

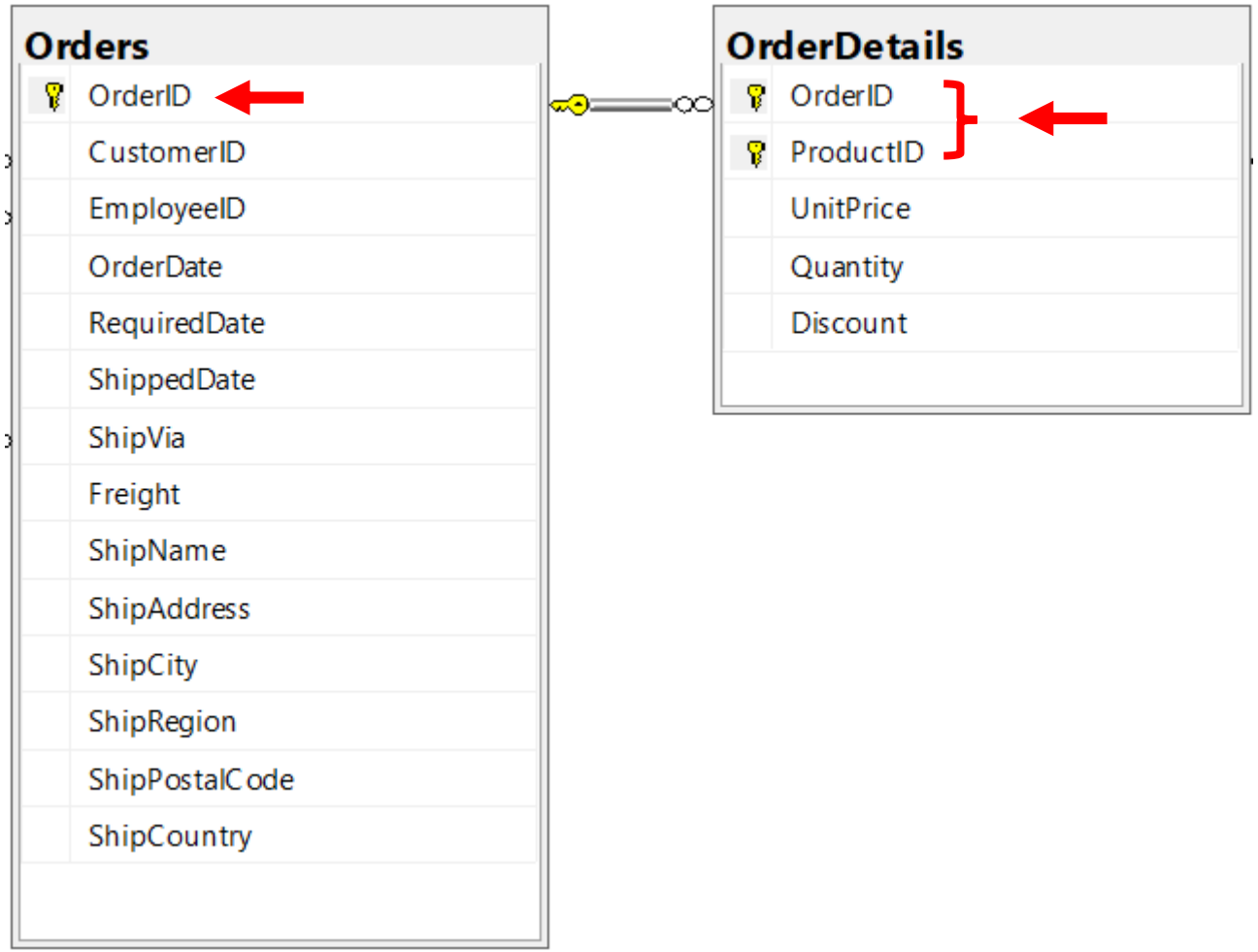


Relation between Datasets/Tables – Zoom In



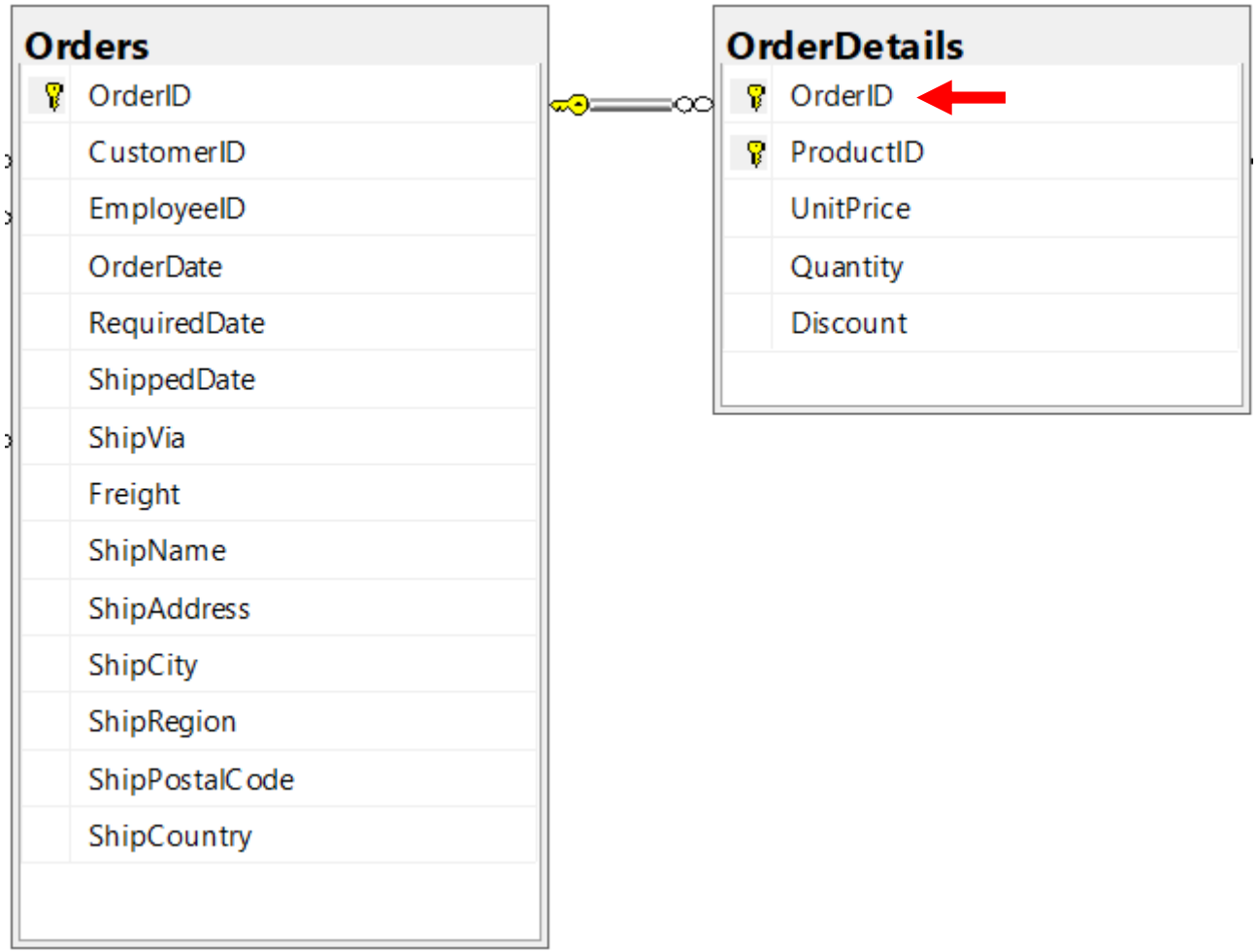
Relation between Datasets/Tables – Zoom In

- **Primary key**



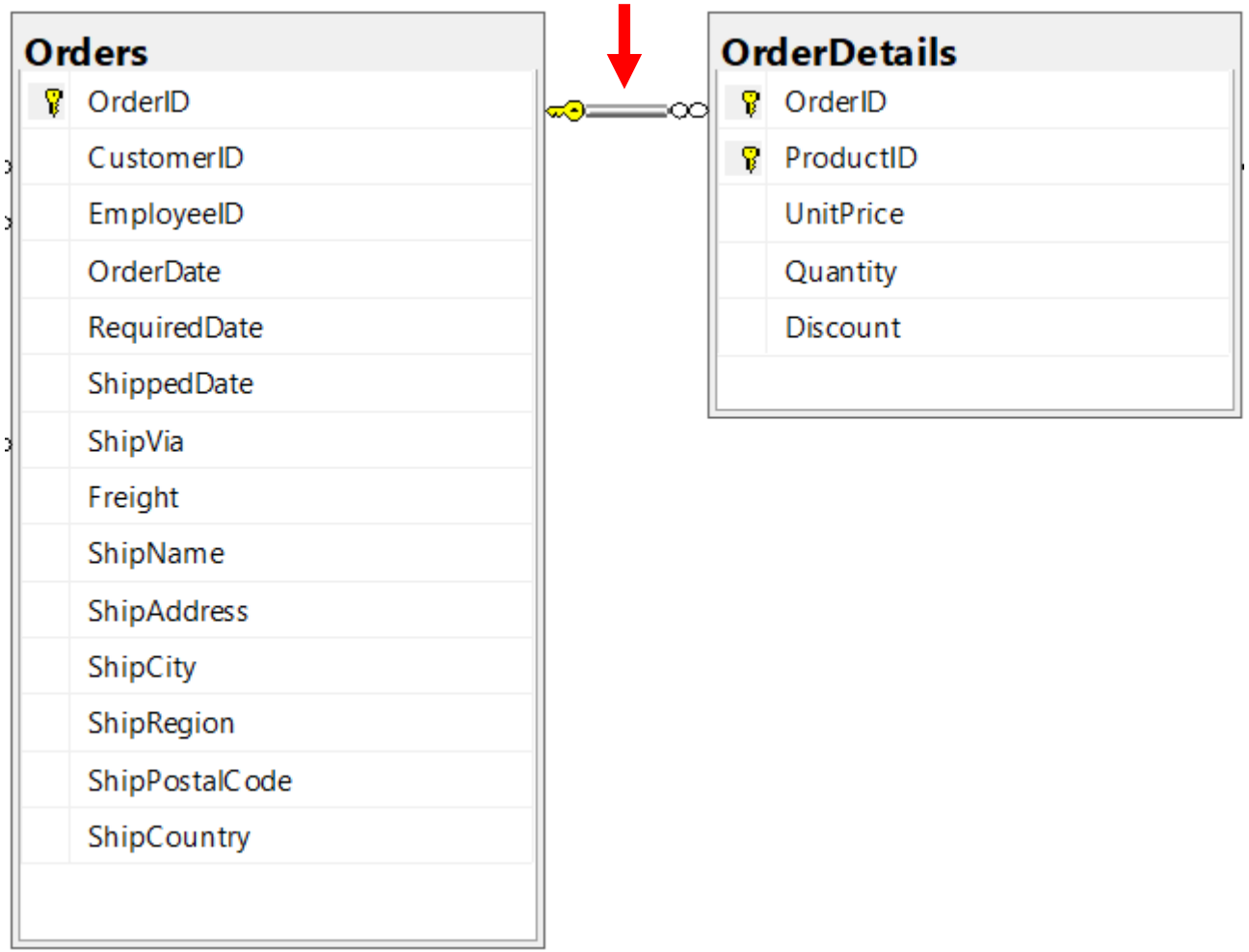
Relation between datasets/tables – Zoom In

- Primary key
- **Foreign key**

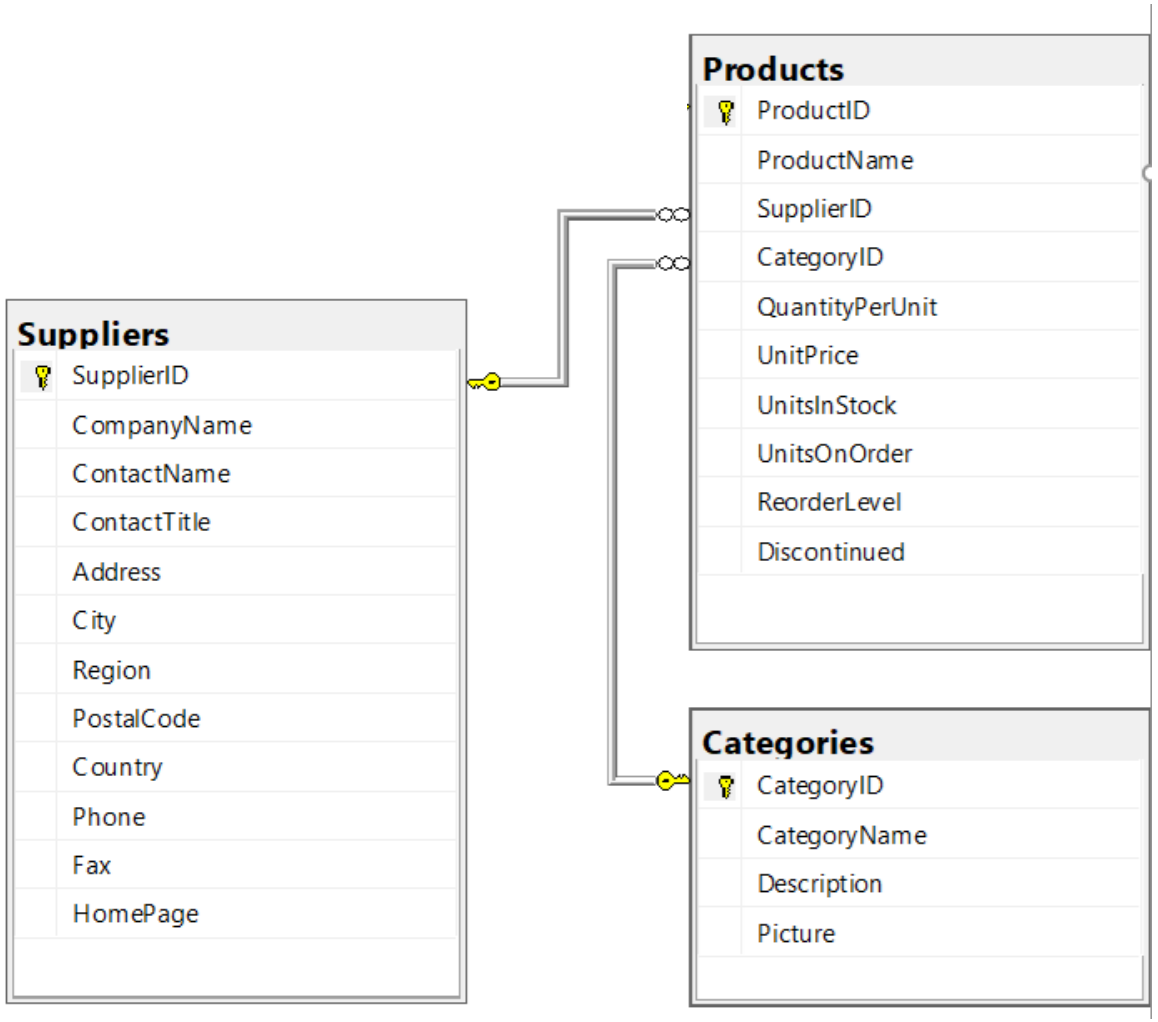


Relation between datasets/tables – Zoom In

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



Relation between Tables – Another Example



Join – Inner Join

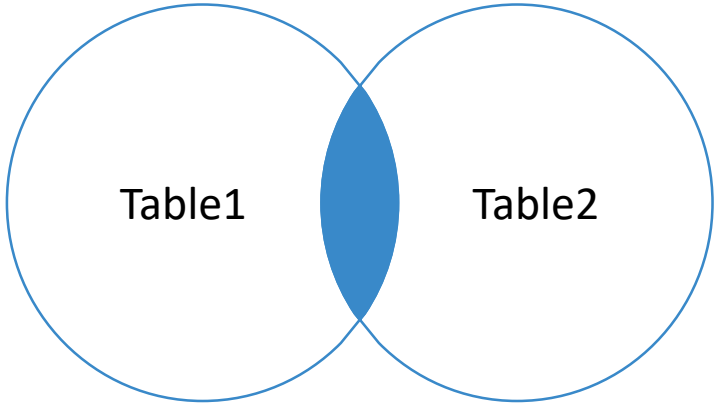


Table1

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

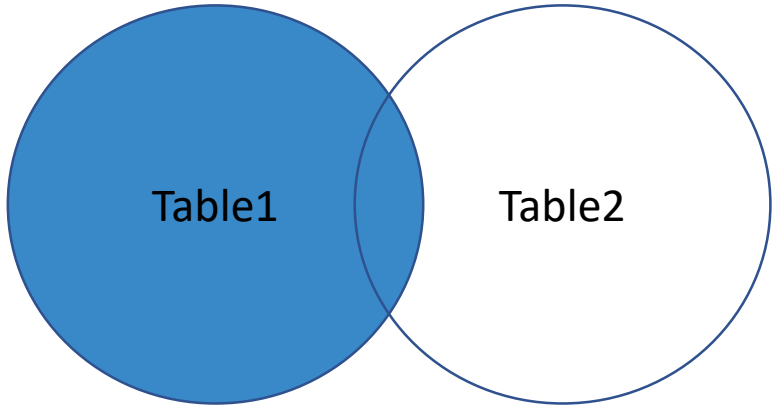


Table1

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

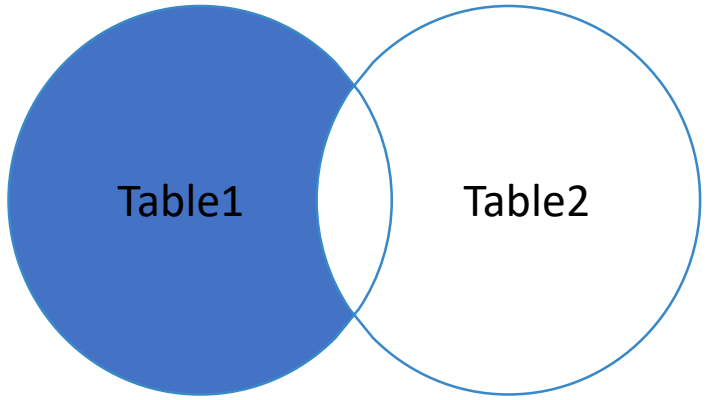


Table1

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))
```

Join – Right (Outer) Join*

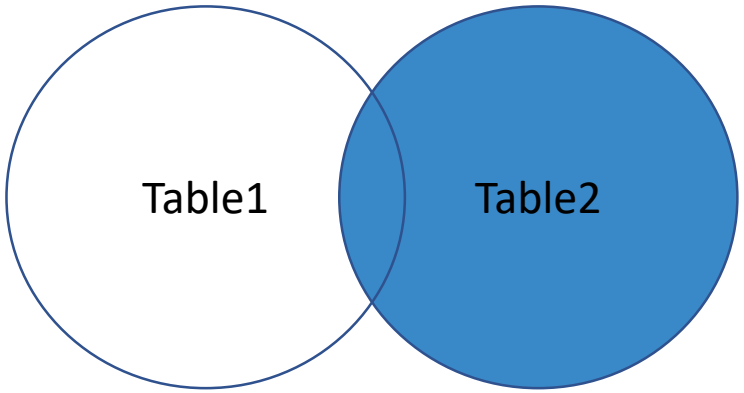


Table1

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*

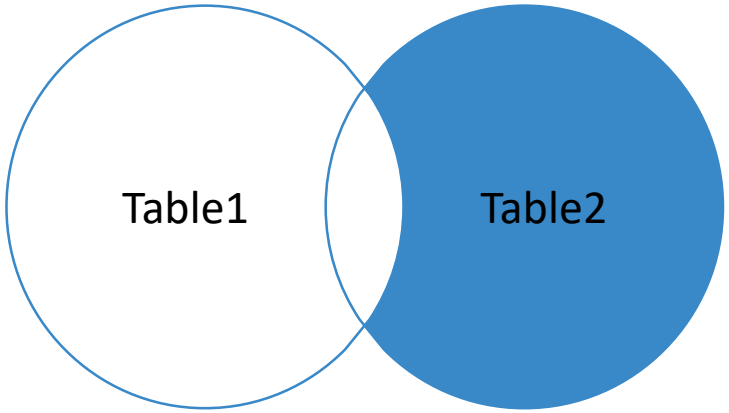


Table1

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

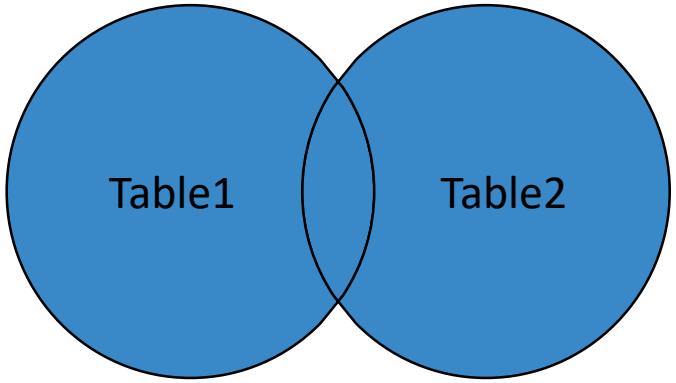


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
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

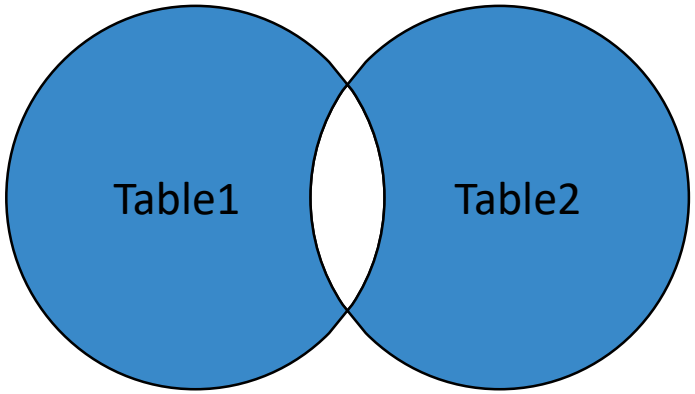


Table1

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))
```