Data Handling: Import, Cleaning and Visualisation

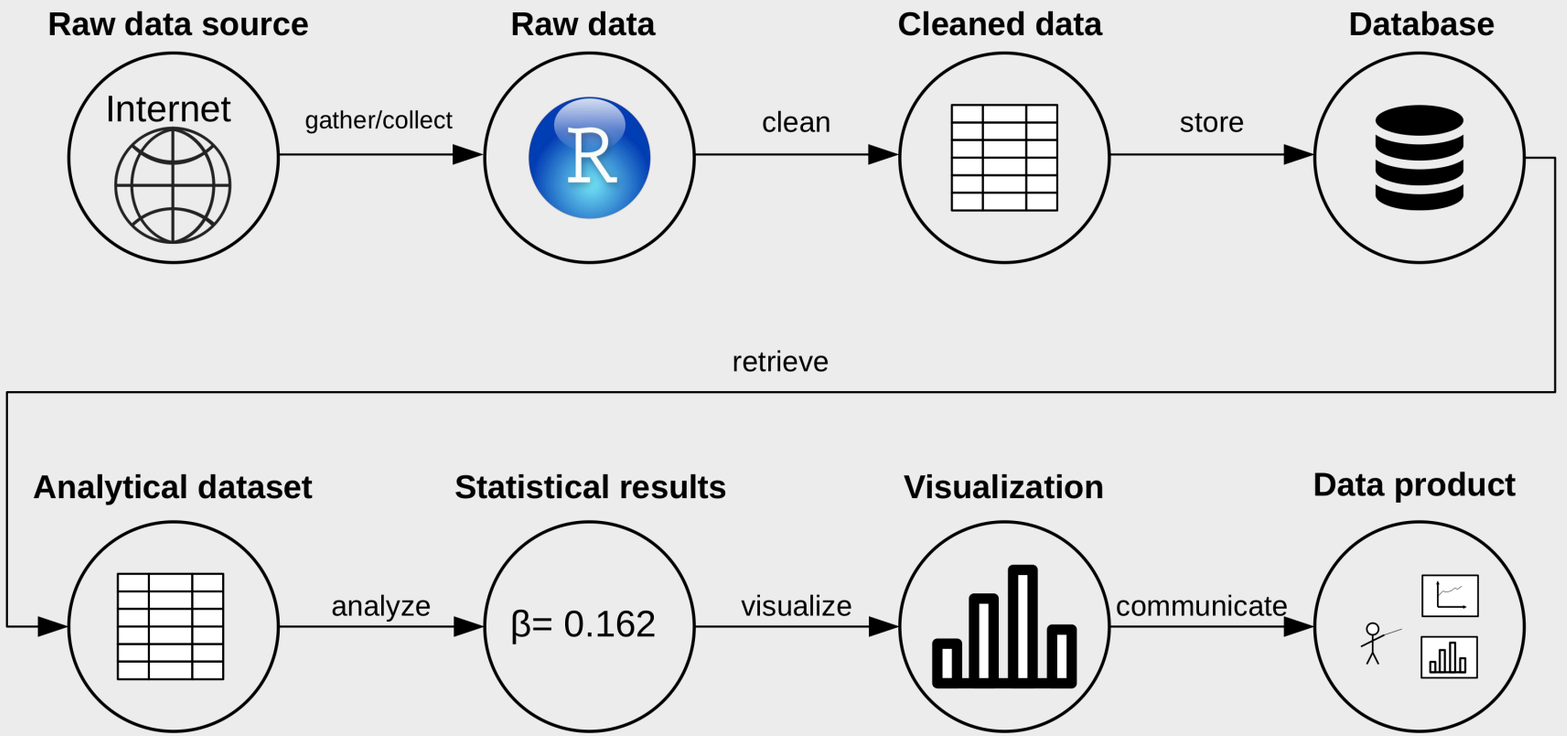
Lecture 8:
Basic Data Analysis with R

Dr. Aurélien Sallin

2024-11-21

Recap: Data Preparation

Data (science) pipeline



Data preparation/data cleaning

Goal of data preparation: Dataset is ready for analysis.

Key conditions:

1. Data values are consistent/clean within each variable.
2. Variables are of proper data types.
3. Dataset is in ‘tidy’ (in long format)!

“Garbage in garbage out (GIGO)”

Data preparation consists of five main steps

- **Tidy** data.
- **Reshape** datasets from wide to long (and vice versa).
- **Bind** or stack rows in datasets.
- **Join** datasets.
- **Clean** data.

A tidy dataset is tidy, when ...

1. Each variable is a column; each column is a variable.
2. Each observation is a row; each row is an observation.
3. Each value is a cell; each cell is a single value.

Reshaping

country	year	metric
x	1960	10
x	1970	13
x	2010	15
y	1960	20
y	1970	23
y	2010	25
z	1960	30
z	1970	33
z	2010	35

```
pivot_wider(names_from = “year”,
            names_prefix = “yr”,
            values_from = “metric”)
```

country	yr1960	yr1970	yr2010
x	10	13	15
y	20	23	25
z	30	33	35

```
pivot_longer(cols = yr1960:yr2010,
             names_to = “year”,
             names_prefix = “yr”
             values_to = “metric”)
```

Long and wide data. Source: Hugo Tavares

Stack/row-bind

ID	X	Y
1	a	50
2	b	10

ID	Z
3	M
4	O

ID	X	Z
5	c	P

ID	X	Y	Z
1	a	50	NA
2	b	10	NA
3	NA	NA	M
4	NA	NA	O
5	c	NA	P

Warm up

Reshaping: multiple/one/none answers correct

Consider the following data frame `schwiizerChuchi`. This dataset records the popularity ratings (on a scale of 1 to 10) of various Swiss dishes in different regions of Switzerland:

```
schwiizerChuchi <- data.frame(
  Region = c("Zurich", "Geneva", "Lucerne"),
  Fondue = c(8, 9, 7),
  Raclette = c(7, 8, 10),
  Rosti = c(9, 6, 8),
  Olma = c(10, 7, 8)
)
```

```
schwiizerChuchiLong <- pivot_longer(schwiizerChuchi,
                                    cols = c(Fondue, Raclette, Rosti, Olma),
                                    values_to = "Popularity",
                                    names_to = "Dish")
```

Which of the following statements is true?

- `nrow(schwiizerChuchiLong) == 12` returns `TRUE`
- `dim(schwiizerChuchiLong)` returns `c(3, 12)`
- `dim(schwiizerChuchi)` returns `c(3, 12)`
- `mean(schwiizerChuchiLong$Raclette) == 8.333`

Tidy data: essay question

Why is this data frame not tidy, and what would you do to make it tidy? Write down your reasoning in numbered steps. You can write down some exact code, some higher-level code concepts, or in plain text.

```
temp_location_data <- data.frame(
  temperature_location = c("22C_London", "18C_Paris", "25C_Rome")
)
```

Tidy data: essay question

Why is this data frame not tidy, and what would you do to make it tidy? Write down your reasoning in numbered steps. You can write down some exact code, some higher-level code concepts, or in plain text.

```
grades_data <- data.frame(
  Student = c("Johannes", "Hannah", "Igor"),
  Econ = c(5, 5.25, 4),
  DataHandling = c(4, 4.5, 5),
  Management = c(5.5, 6, 6)
)
```

Today

Goals of today's lecture

1. Understand the concept of merging datasets.
2. Perform basic data manipulation in `dplyr`.
3. First steps in exploratory data analysis.

Data Preparation: merging

Merging (Joining) datasets

- Combine data of two datasets in one dataset.
- Needed: Unique identifiers for observations (‘keys’).

Merging (joining) datasets

x

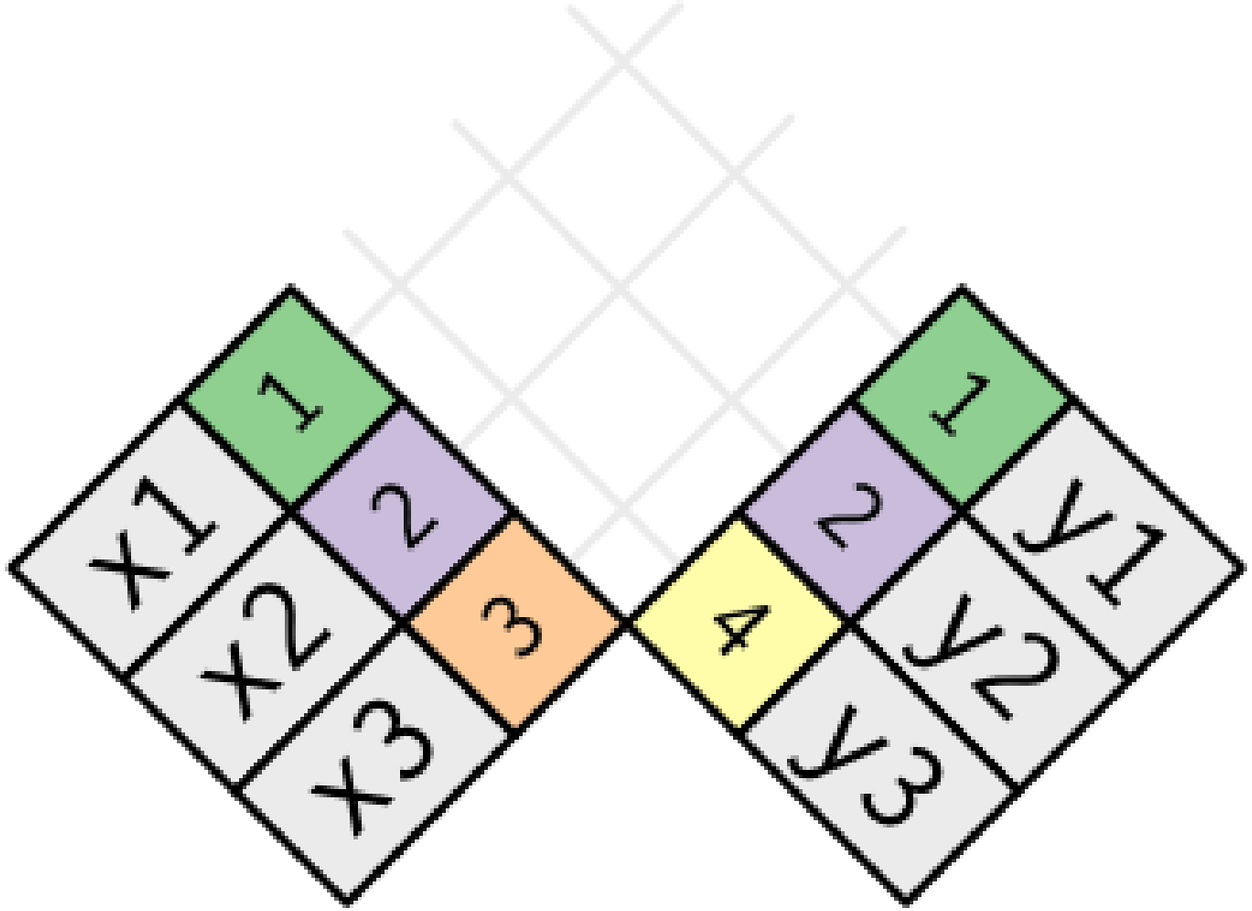
1	x1
2	x2
3	x3

y

1	y1
2	y2
4	y3

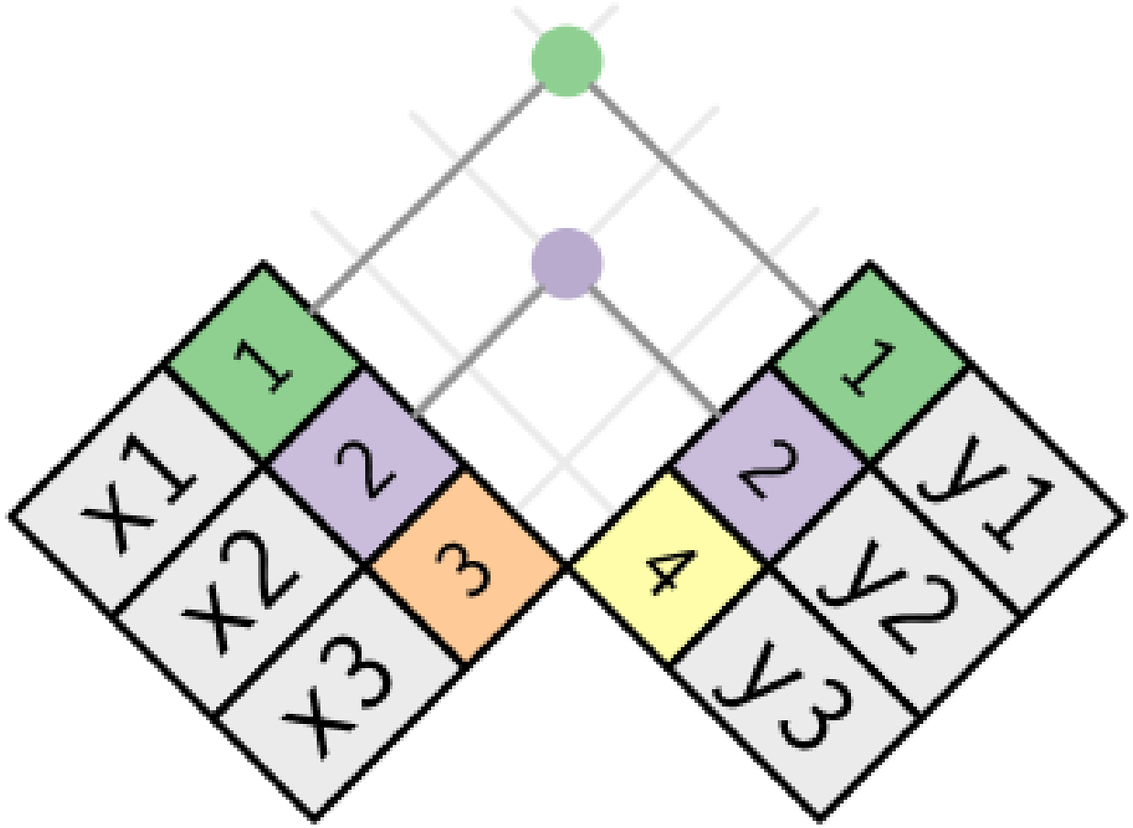
Join setup. Source: R4DS.

Merging (joining) datasets



Join setup. Source: R4DS.

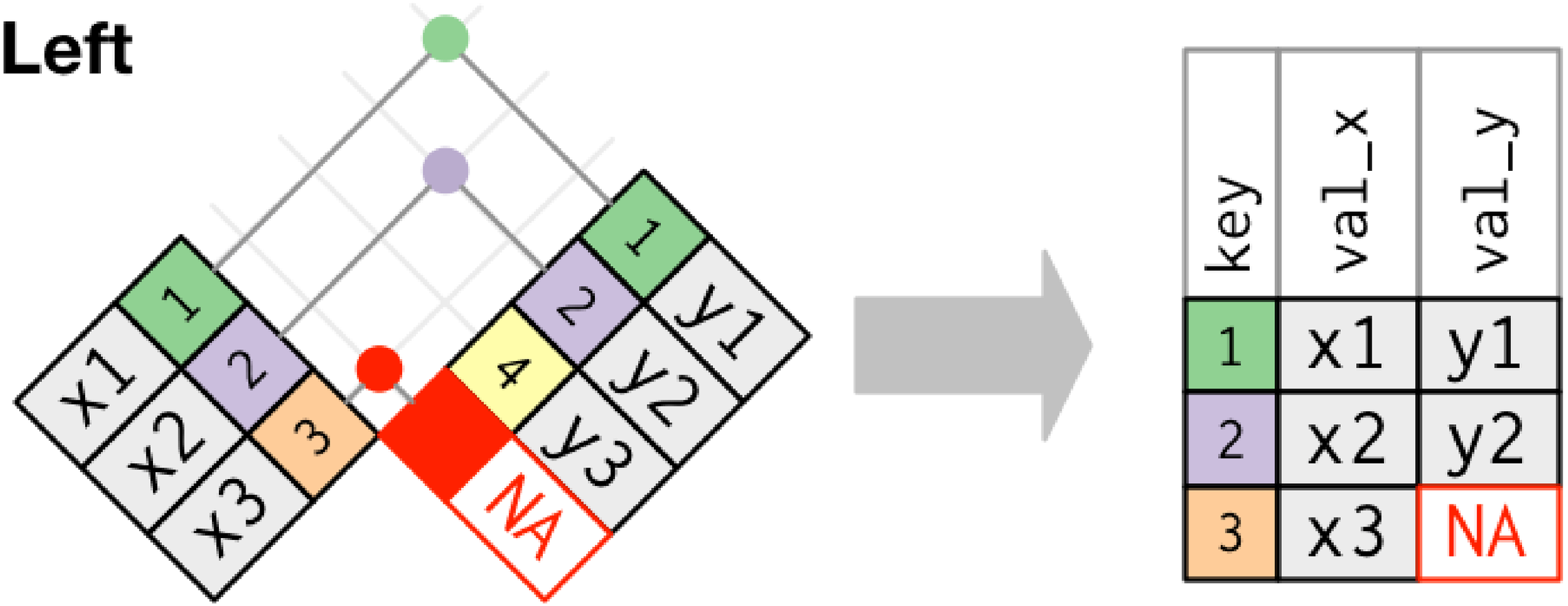
Merging (joining) datasets using an inner join



key	val_x	val_y
1	x1	y1
2	x2	y2

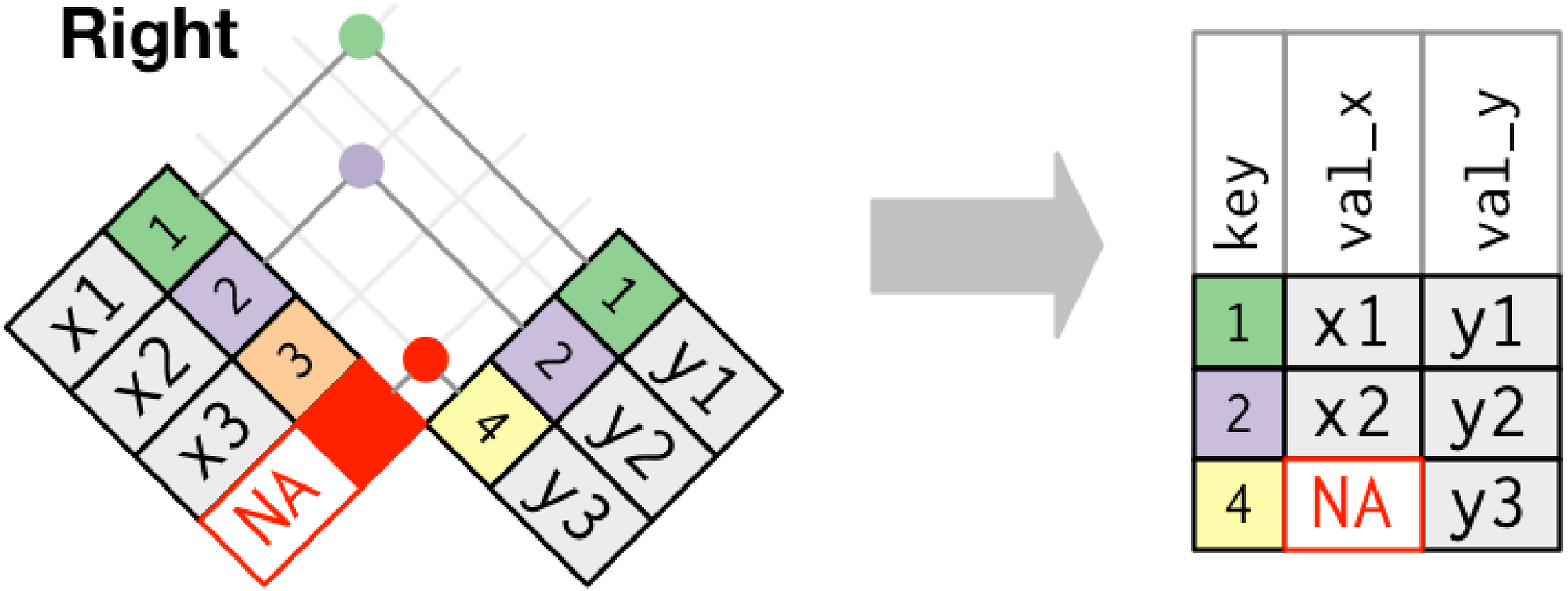
Inner join. Source: R4DS

Merging (joining) datasets using a left join



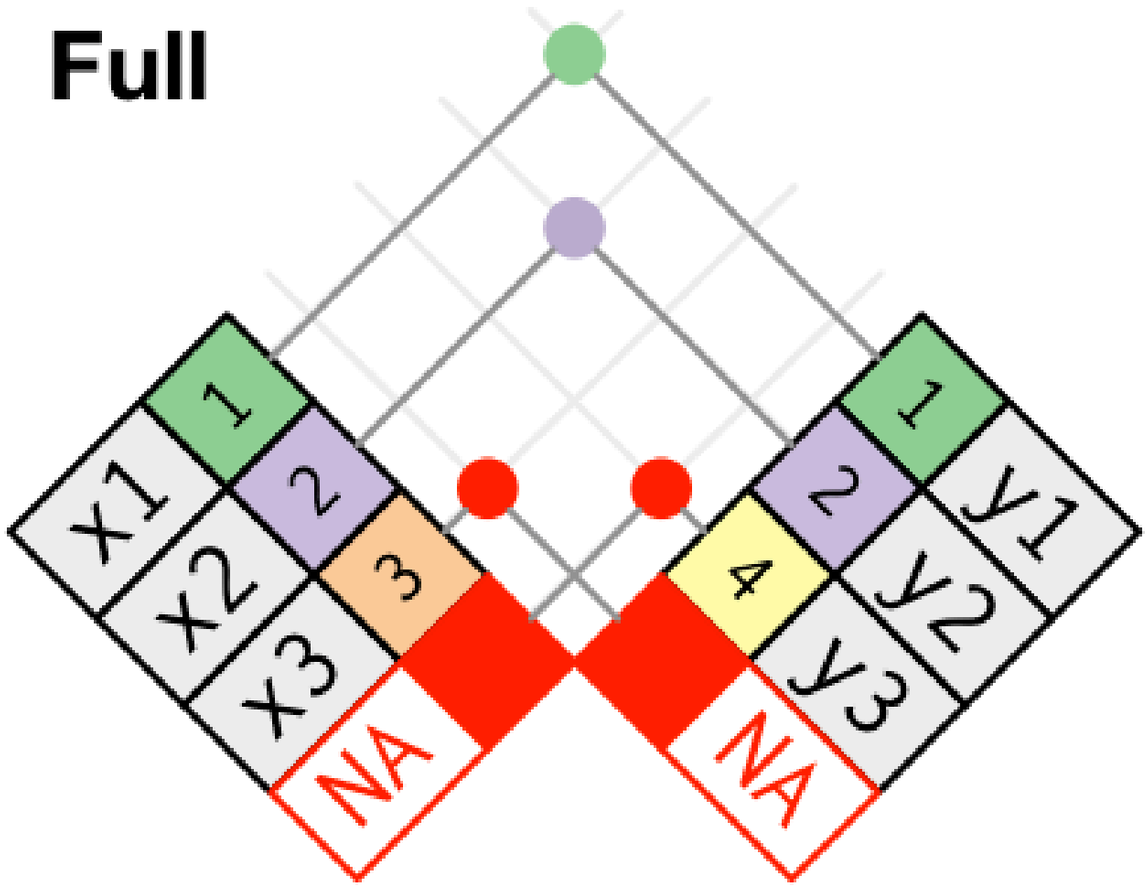
Left join. Source: R4DS.

Merging (joining) datasets using a right join



Right join. Source: R4DS.

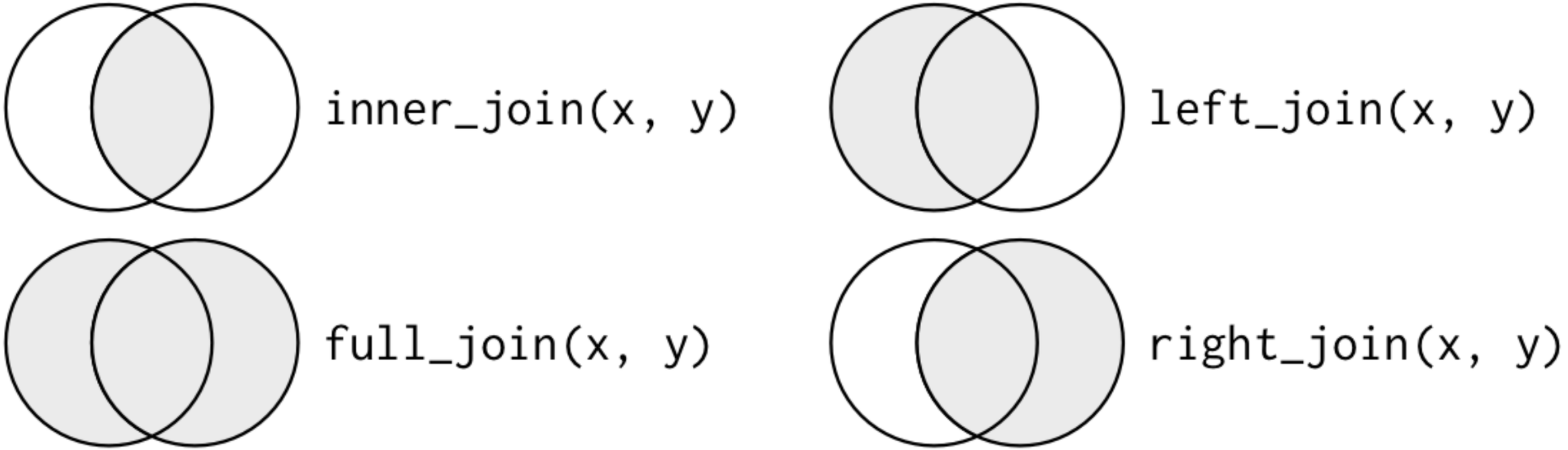
Merging all x and all y using a full join



key	val_x	val_y
1	x1	y1
2	x2	y2
3	x3	NA
4	NA	y3

Full join. Source: R4DS.

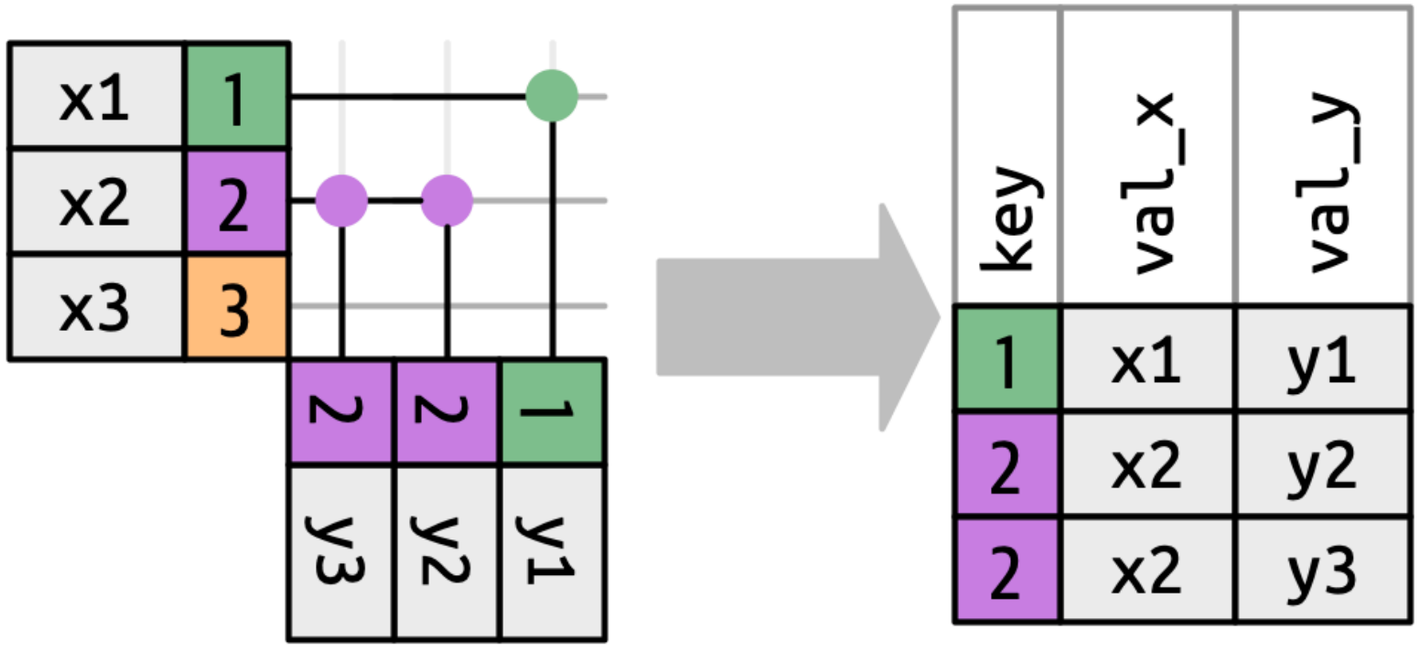
The four fundamental joins are :



Join Venn Diagramm. Source: R4DS.

Row-matching behaviors in joins

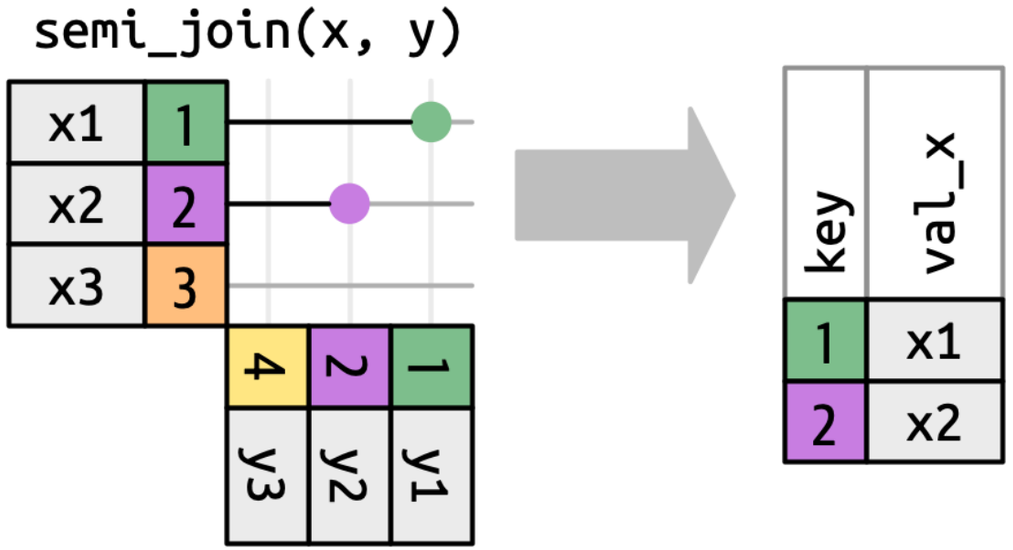
1. “One-to-one”
2. “Many-to-many”
3. “One-to-many”
4. “Many-to-one”



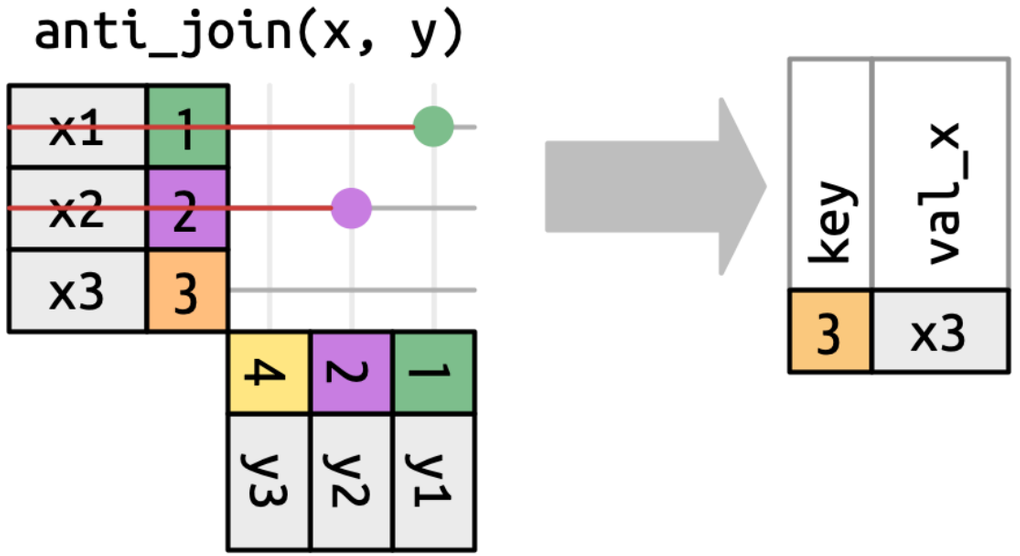
One-to-many joins. Source: R4DS

Always check how many rows are returned after your merge! In `tidyverse`, warnings appear in case of “many-to-many”. As of `dplyr 1.1.1`, no warning for one-to-many relationships.

Filtering joins with the semi join and the anti join



The semi-join keeps rows in x that have one or more matches in y. Source: R4DS.



The anti-join keeps rows in x that match zero rows in y. Source: R4DS.

Merging (joining) datasets: example

```
# load packages
library(tidyverse)

# initiate data frame on persons personal spending
df_c <- data.frame(id = c(1:3,1:3),
                   money_spent= c(1000, 2000, 6000, 1500, 3000, 5500),
                   currency = c("CHF", "CHF", "USD", "EUR", "CHF", "USD"),
                   year=c(2017,2017,2017,2018,2018,2018))
df_c
```

```
  id money_spent currency year
1  1        1000      CHF 2017
2  2        2000      CHF 2017
3  3        6000      USD 2017
4  1        1500      EUR 2018
5  2        3000      CHF 2018
6  3        5500      USD 2018
```

Merging (joining) datasets: example

```
# initiate data frame on persons' characteristics
df_p <- data.frame(id = 1:4,
                   first_name = c("Anna", "Betty", "Claire", "Diane"),
                   profession = c("Economist", "Data Scientist",
                                  "Data Scientist", "Economist"))
df_p
```

```
  id first_name     profession
1  1       Anna      Economist
2  2      Betty Data Scientist
3  3     Claire Data Scientist
4  4      Diane      Economist
```

Merging (joining) datasets: example

```
df_merged <- left_join(df_p, df_c, by="id")
df_merged
```

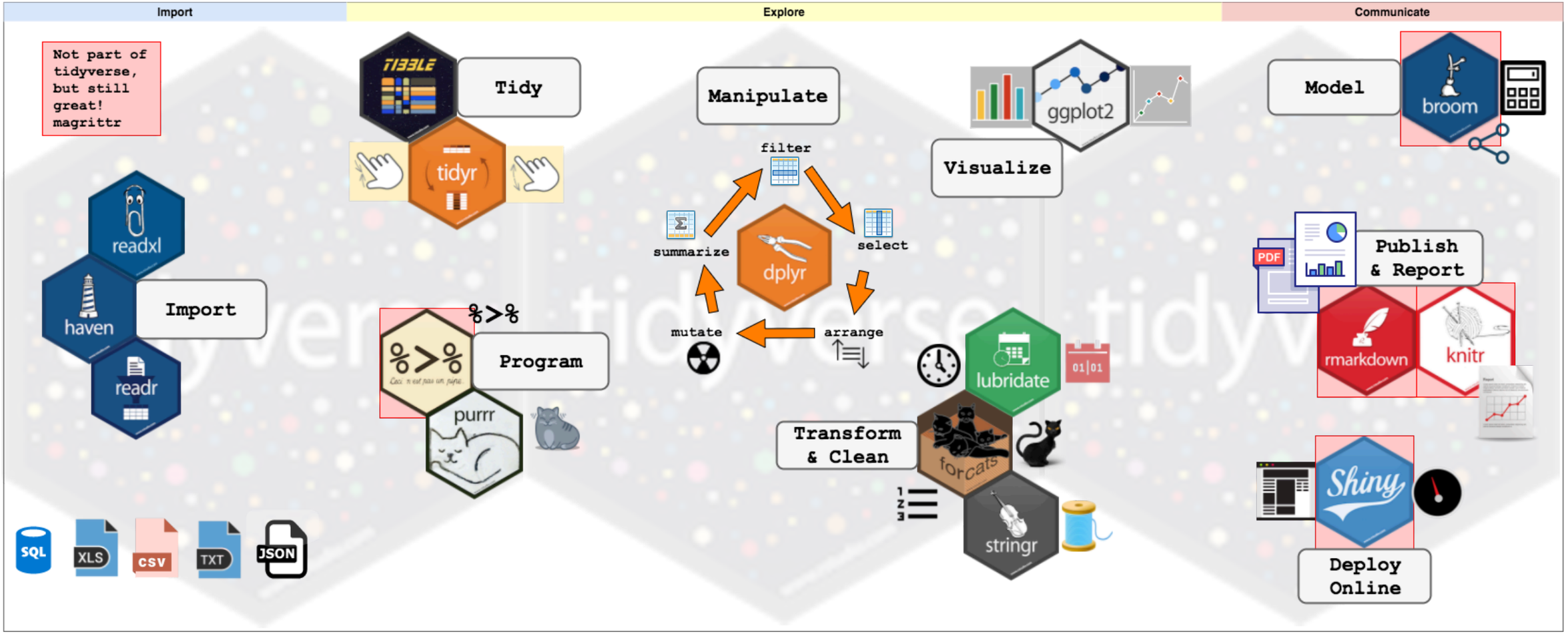
```
  id first_name     profession money_spent currency year
1  1       Anna      Economist        1000      CHF 2017
2  1       Anna      Economist        1500      EUR 2018
3  2      Betty Data Scientist        2000      CHF 2017
4  2      Betty Data Scientist        3000      CHF 2018
5  3     Claire Data Scientist        6000      USD 2017
6  3     Claire Data Scientist        5500      USD 2018
7  4      Diane      Economist          NA     <NA>   NA
```

Merging (joining) datasets: R

Overview by R4DS:

dplyr (tidyverse)	base::merge
`inner_join(x, y)`	`merge(x, y)`
`left_join(x, y)`	`merge(x, y, all.x = TRUE)`
`right_join(x, y)`	`merge(x, y, all.y = TRUE),`
`full_join(x, y)`	`merge(x, y, all = TRUE)`

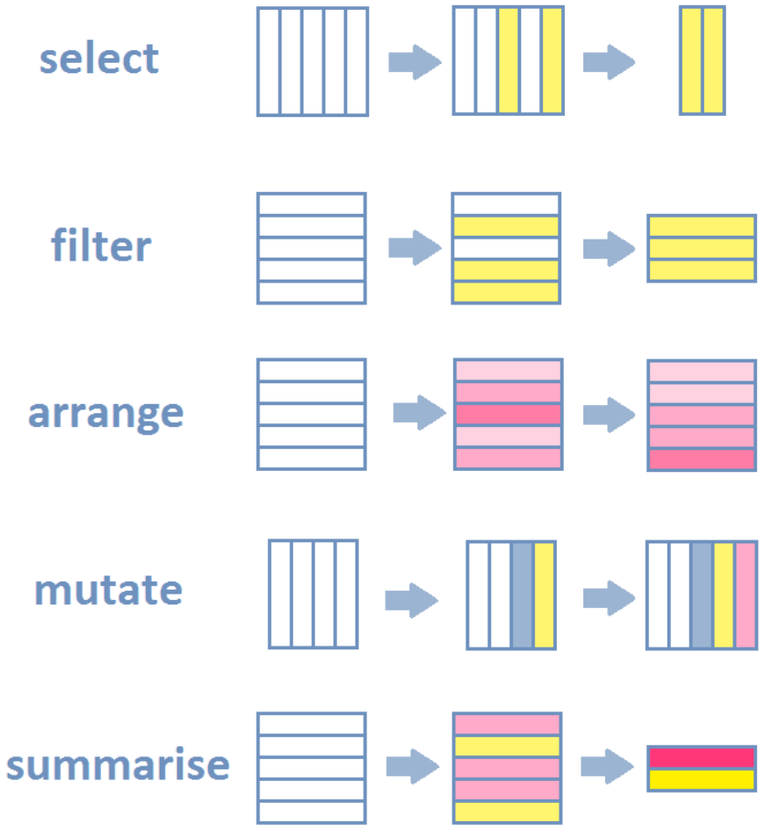
Transforming and cleaning data



Source: https://www.storybench.org/wp-content/uploads/2017/05/tidyverse.png

`select`, `filter`, `arrange`, `mutate` are the building blocks of `dplyr`

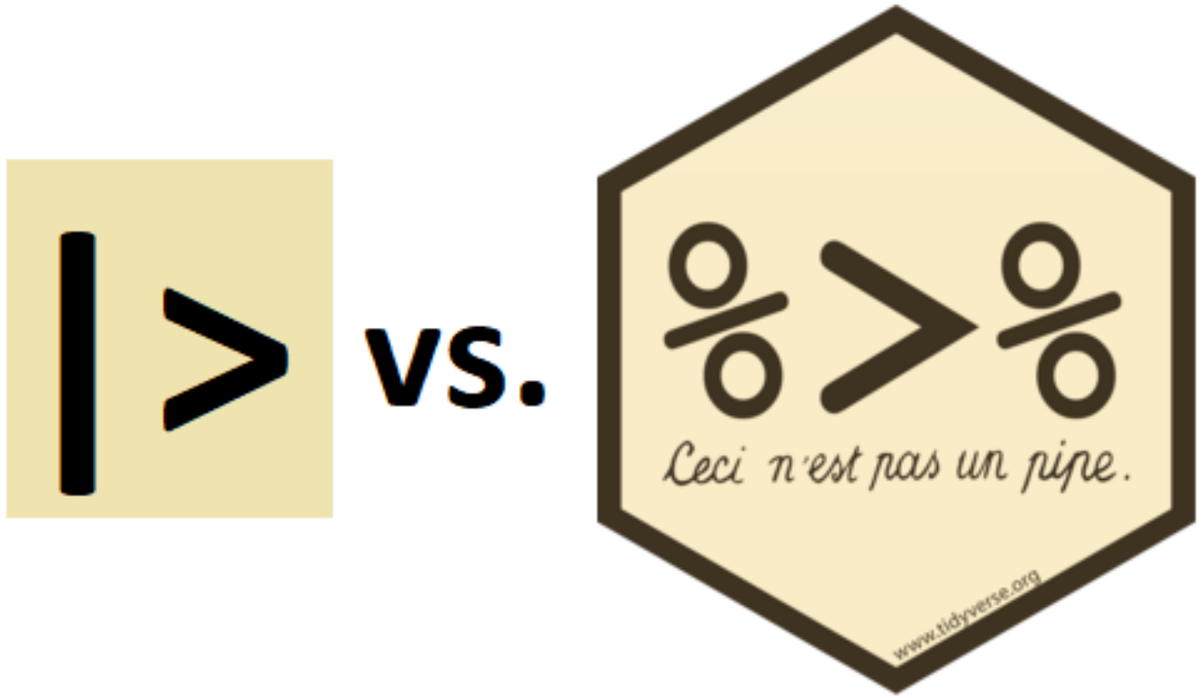
- **Select** the subset of variables you need (e.g., for comparisons).
- **Filter** the dataset by restricting your dataset to observations needed in *this* analysis.
- **Arrange** the dataset by reordering the rows.
- **Mutate** the dataset by adding the values you need for your analysis.
- **Group** and **summarize** the dataset by a variable to apply functions to groups of observations.



Source: Intro to R for Social Scientists

Prepare your data in a `pipe`line

- Using the `piping` `%>%` operator is to chain one function after another without the need to assign intermediate variables.
- The operator has been now replaced with `|>`.



Prepare your data in a pipeline with dplyr

```r
# Traditional way
mydf <- data(swiss)
mydf <- arrange(mydf, -Catholic)
mydf <- filter(mydf, Education > 8 & Catholic > 90)
mydf <- mutate(mydf, Country = "Switzerland")
mydf <- select(mydf, Examination)

# The pipe way
mydf <- data(swiss)  |>
  arrange(-Catholic) |>
  filter(Education > 8 & Catholic > 90) |>
  mutate(Country = "Switzerland") |>
  select(Examination)
```

```r
# Base-R equivalent
mydf <- data(swiss)
mydf <- mydf[order(-mydf$Catholic), ]
mydf <- mydf[mydf$Education > 8 & mydf$Catholic > 90, ]
mydf$Country <- "Switzerland"
mydf <- mydf["Examination"]
```

Further tools for data transformation and cleaning in `dplyr`

- `forecats` to deal with factors;
- `lubridate` to deal with dates;
- `stringr` to deal with strings and regular expressions.

Exploratory Data Analysis and Descriptive Statistics


TERRA INCOGNITA

Exploratory Data Analysis is the first step of data analysis

- 🔎 Get a first understanding of your dataset.
- Show key aspects of data by modelling, transforming, and visualizing your data.
  - Investigate the quality and reliability of your data.
  - Inform your own statistical analysis.
  - Inform audience (helps understand advanced analytics parts).
- … which in turns generates new questions about your dataset (creative process 🎨).

One piece of advice and two questions will guide you in exploring your data

Know your data

- *“To not mislead others and not embarrass yourself, know your data”.*
- Understand the **number of observations**, the **units**, the **quality** of data, the **definitions** of variables, what to do with **missing values**.

What type of variation occurs within my variables?

- **Typical values**: mean, mode, range, standard deviation.
- **Surprising values**: outliers, rare values, unusual patterns.

What type of covariation occurs between my variables?

- **Covariation** and **Patterns**.

Source: R4DS, “Statistics for Public Policy” by Jeremy G. Weber, 2024.

Exploratory Data Analysis: first steps in R

- *Quick overview*: `summary()`
- *Cross-tabulation*: `table()`
- *Summarizing tools*: `skimr`, `summarytools`, `janitor` (also cleaning)

Descriptive/aggregate statistics

- Overview of key characteristics of main variables used in analysis.
- Key characteristics:
  - Mean
  - Standard deviation
  - No. of observations
  - etc.

Aggregate statistics in R

1. Functions to compute statistics (e.g., `mean()`).
2. Functions to *apply* the statistics function to one or several columns in a tidy dataset.

- Including all values in a column.
- By group (observation categories, e.g. by location, year, etc.)

`summary()` in `base`; `summarise()` in `tidyverse`; `group_by()` in `tidyverse`; `sapply()`, `apply()`, `lapply()`, etc. in `base`; `skimr` package; etc.

Exploratory Data Analysis: an example with the `swiss` data

▶ Show code

```
swiss
```

```
   municipality Fertility Agriculture Examination Education Catholic Infant.Mortality
1     Courtelary      80.2        17.0          15        12     9.96             22.2
2       Delemont      83.1        45.1           6         9    84.84             22.2
3   Franches-Mnt      92.5        39.7           5         5    93.40             20.2
4        Moutier      85.8        36.5          12         7    33.77             20.3
5     Neuveville      76.9        43.5          17        15     5.16             20.6
6     Porrentruy      76.1        35.3           9         7    90.57             26.6
7          Broye      83.8        70.2          16         7    92.85             23.6
8          Glane      92.4        67.8          14         8    97.16             24.9
9        Gruyere      82.4          NA          12         7    97.67             21.0
10        Sarine      82.9        45.2          16        13    91.38             24.4
11       Veveyse      87.1        64.5          14         6    98.61             24.5
12         Aigle      64.1        62.0          21        12     8.52             16.5
13       Aubonne      66.9        67.5          14         7     2.27             19.1
14     Avenches       68.9        60.7          19        12     4.43             22.7
15     Cossonay       61.7        69.3          22         5     2.82             18.7
16    Echallens       68.3        72.6          18         2    24.20             21.2
17     Grandson       71.7        34.0          17         8     3.30             20.0
18     Lausanne       55.7        19.4          26        28    12.11            100.0
19    La Vallee       54.3        15.2          31        20     2.15             10.8
20        Lavaux      65.1        73.0          19         9     2.84             20.0
21       Morges       65.5        59.8          22        10     5.23             18.0
22        Moudon      65.0        55.1          14         3     4.52             22.4
```

Exploratory Data Analysis: an example with the `swiss` data

```
summary(swiss)
```

```
 municipality          Fertility      Agriculture     Examination      Education
 Length:47          Min.   :35.00   Min.   : 1.20   Min.   : 3.00   Min.   : 1.00
 Class :character   1st Qu.:64.70   1st Qu.:35.60   1st Qu.:12.00   1st Qu.: 6.00
 Mode  :character   Median :70.40   Median :54.60   Median :16.00   Median : 8.00
                    Mean   :70.14   Mean   :50.60   Mean   :16.49   Mean   :10.98
                    3rd Qu.:78.45   3rd Qu.:67.72   3rd Qu.:22.00   3rd Qu.:12.00
                    Max.   :92.50   Max.   :89.70   Max.   :37.00   Max.   :53.00
                                    NA's   :1
    Catholic       Infant.Mortality
 Min.   :  2.150   Min.   : 10.80
 1st Qu.:  5.195   1st Qu.: 18.15
 Median : 15.140   Median : 20.00
 Mean   : 41.144   Mean   : 21.64
 3rd Qu.: 93.125   3rd Qu.: 22.20
 Max.   :100.000   Max.   :100.00
```

Exploratory Data Analysis: an example with the `swiss` data

```
skimr::skim(swiss)
```

Data summary

Name	swiss
Number of rows	47
Number of columns	7
Column type frequency:	
character	1
numeric	6
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
municipality	0	1	4	12	0	47	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	his
Fertility	0	1.00	70.14	12.49	35.00	64.70	70.40	78.45	92.5	
Agriculture	1	0.98	50.60	22.96	1.20	35.60	54.60	67.72	89.7	
Examination	0	1.00	16.49	7.98	3.00	12.00	16.00	22.00	37.0	
Education	0	1.00	10.98	9.62	1.00	6.00	8.00	12.00	53.0	
Catholic	0	1.00	41.14	41.70	2.15	5.20	15.14	93.12	100.0	
Infant.Mortality	0	1.00	21.64	12.04	10.80	18.15	20.00	22.20	100.0	

Exploratory Data Analysis: an example with the `swiss` data

```
swiss |>
  group_by(Catholic > 50) |>
  summarize(mean(Fertility))
```

```
# A tibble: 2 × 2
  `Catholic > 50` `mean(Fertility)`
  <lgl>                       <dbl>
1 FALSE                        66.2
2 TRUE                         76.5
```

```
swiss |>
  group_by(Catholic > 50) |>
  summarize(across(.cols = c(Fertility, Education),
                   .fns = list("min" = min, "mean" = mean, "max" = max)))
```

```
# A tibble: 2 × 7
  `Catholic > 50` Fertility_min Fertility_mean Fertility_max Education_min Education_mean
  <lgl>                   <dbl>          <dbl>         <dbl>         <int>          <dbl>
1 FALSE                    35             66.2          85.8             1          12.1
2 TRUE                     42.8           76.5          92.5             2           9.11
# ℹ 1 more variable: Education_max <int>
```

Some practice

Summarizing categorical variables: challenge

Use what we just saw in the lecture to solve the following problem. You have the following dataset:

```
df_p <- data.frame(id = 1:5,
                   first_name = c("Anna", "John", "Claire", "Evan", "Brigitte"),
                   profession = c("Economist", "Data Scientist",
                                  "Data Scientist", "Economist", "Economist"),
                   salaryK = c("100 ", 120, 90, 110, 105),
                   experienceY = c(10, 10, 10, 10, 10))

df_p
```

1. Clean the data
2. Summarize the data.
3. Give summary statistics on the categorical variable “profession”. What can you show, and how can you code it?
4. You are interested in quantifying the gender pay gap. Prepare the data accordingly and give an estimate of the gender pay gap.