SEOUL NATIONAL UNIVERSITY
Graduate School of Data Science

M3239.003100: Data Analysis and Visualization
Lecture 4

Univariate Analysis

Hyunwoo Park
Graduate School of Data Science
Seoul National University

Agenda

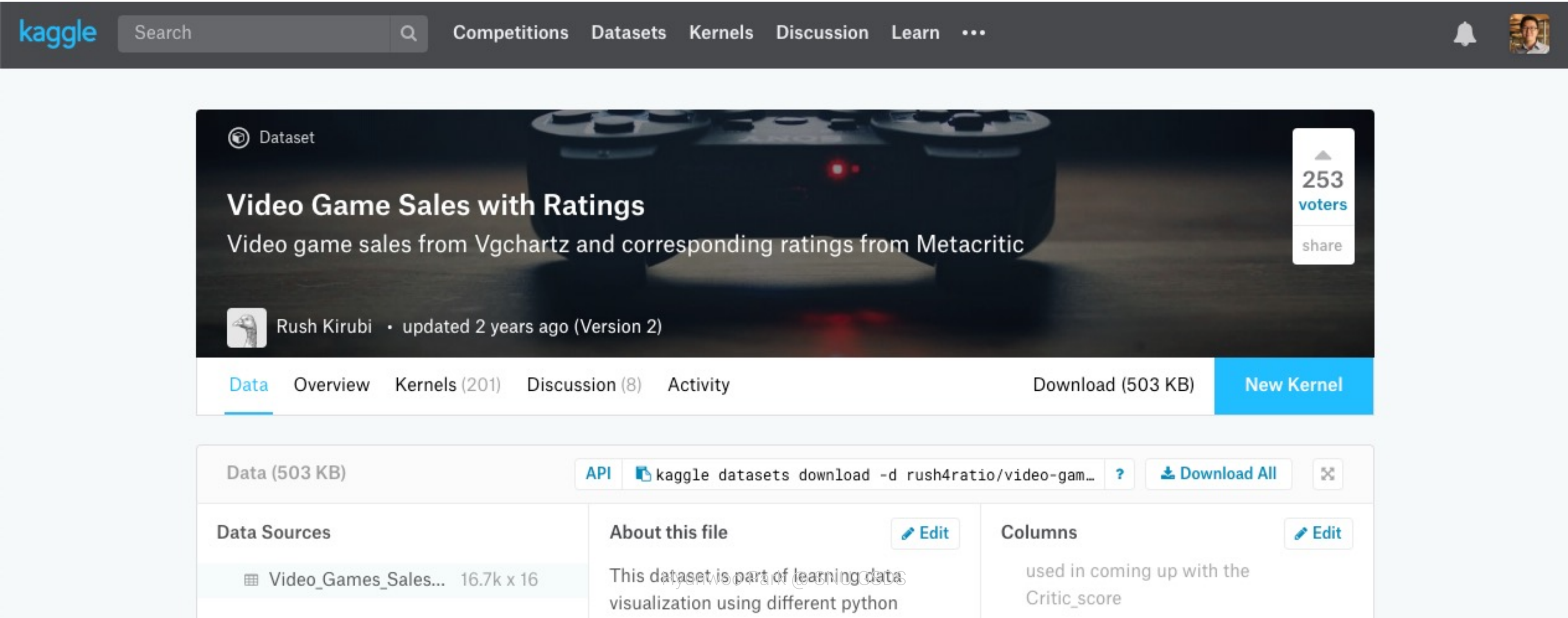
- Setup
  - Data Loading
  - Tidy Data
  - Data Transformation
- Descriptive Statistics
  - Describing One Variable
  - Types of Variables
  - Distributions
  - Summary Statistics

- Homework 1
  - Due 9/28 Tue Before Class 12:30pm
- Things to do
  - Review pandas / csv packages.
  - Compute summary statistics

Setup

Let's grab a dataset.

- Video Game Sales with Ratings
  https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings



Video Game Sales with Ratings

Video game sales from Vgchartz and corresponding ratings from Metacritic

Rush Kirubi • updated 2 years ago (Version 2)

Data (503 KB)

```
kaggle datasets download -d rush4ratio/video-gam…
```

Data Sources

Video_Games_Sales... 16.7k x 16

About this file

This dataset is part of learning data visualization using different python

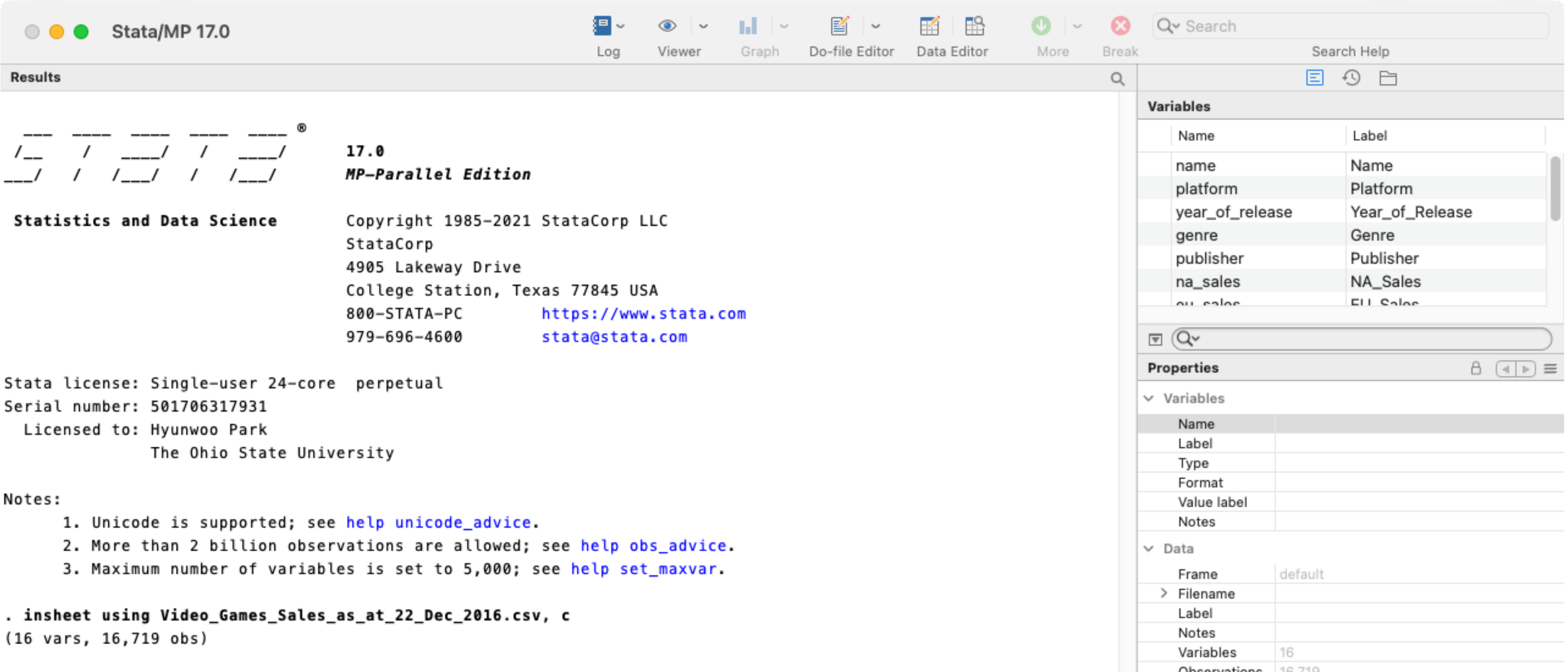
Columns

used in coming up with the Critic_score

Data Loading

- Using Stata

```
insheet using Video_Games_Sales_as_at_22_Dec_2016.csv, c
```



```
  ___  ____  ____  ____  ____ ®
 /__    /   ____/   /   ____/      17.0
___/   /   /___/   /   /___/       MP—Parallel Edition

 Statistics and Data Science       Copyright 1985-2021 StataCorp LLC
                                   StataCorp
                                   4905 Lakeway Drive
                                   College Station, Texas 77845 USA
                                   800-STATA-PC        https://www.stata.com
                                   979-696-4600        stata@stata.com

Stata license: Single-user 24-core  perpetual
Serial number: 501706317931
  Licensed to: Hyunwoo Park
               The Ohio State University

Notes:
      1. Unicode is supported; see help unicode_advice.
      2. More than 2 billion observations are allowed; see help obs_advice.
      3. Maximum number of variables is set to 5,000; see help set_maxvar.

. insheet using Video_Games_Sales_as_at_22_Dec_2016.csv, c
(16 vars, 16,719 obs)
```

Name	Label
name	Name
platform	Platform
year_of_release	Year_of_Release
genre	Genre
publisher	Publisher
na_sales	NA_Sales
eu_sales	EU_Sales

Variables

Name	
Label	
Type	
Format	
Value label	
Notes	

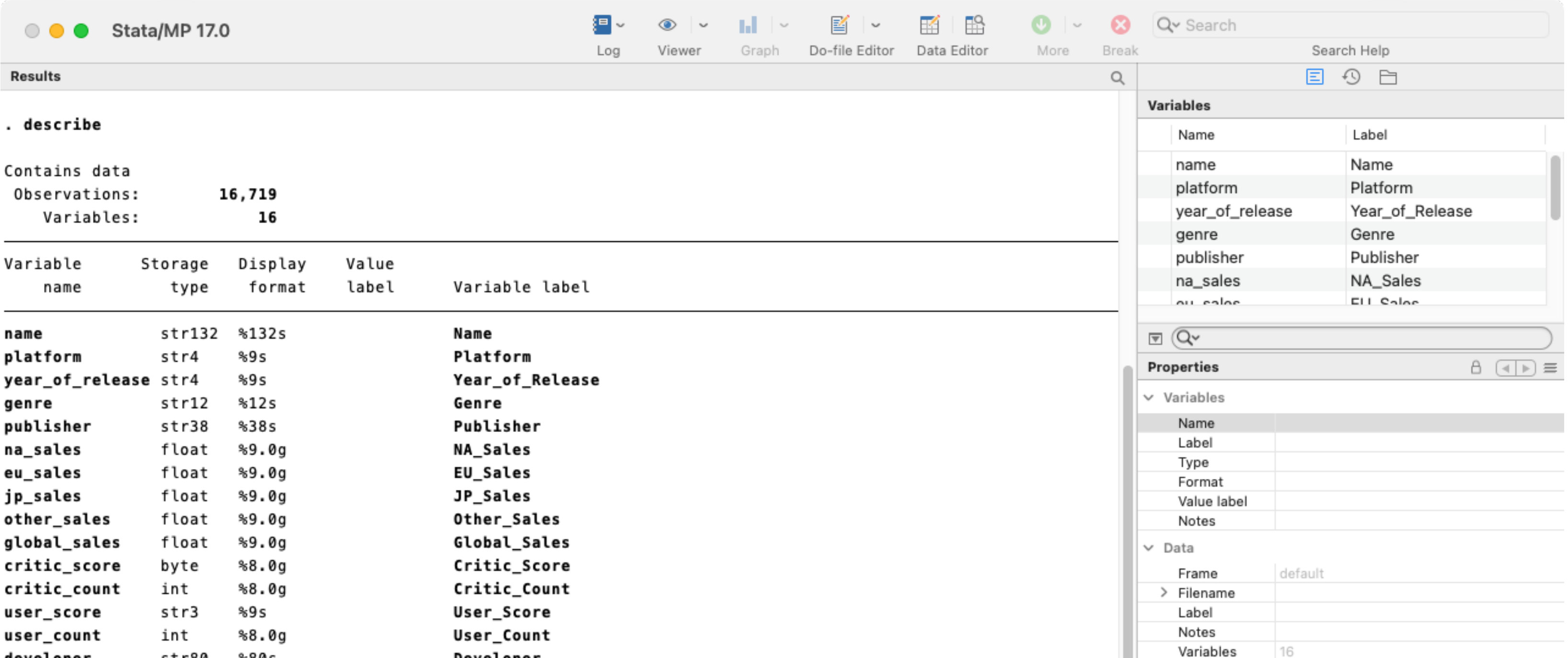
Data

Frame	default
Filename	
Label	
Notes	
Variables	16
Observations	16,719

Data loading

- Using Stata

```
describe
```



```
. describe

Contains data
 Observations:        16,719
    Variables:            16
--------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
--------------------------------------------------------------------------------
name            str132  %132s                 Name
platform        str4    %9s                   Platform
year_of_release str4    %9s                   Year_of_Release
genre           str12   %12s                  Genre
publisher       str38   %38s                  Publisher
na_sales        float   %9.0g                 NA_Sales
eu_sales        float   %9.0g                 EU_Sales
jp_sales        float   %9.0g                 JP_Sales
other_sales     float   %9.0g                 Other_Sales
global_sales    float   %9.0g                 Global_Sales
critic_score    byte    %8.0g                 Critic_Score
critic_count    int     %8.0g                 Critic_Count
user_score      str3    %9s                   User_Score
user_count      int     %8.0g                 User_Count
developer       str80   %80s                  Developer
```

Variables

Name	Label
name	Name
platform	Platform
year_of_release	Year_of_Release
genre	Genre
publisher	Publisher
na_sales	NA_Sales
eu_sales	EU_Sales

Properties

Variables

Name	
Label	
Type	
Format	
Value label	
Notes	

Data

Frame	default
Filename	
Label	
Notes	
Variables	16

What is tidy data?

- It depends on the data context.
- Variables are easier to be linked; observations are harder.
- Further explanation on tidy data:
  Wickham, H. (2014). Tidy data. Journal of Statistical Software, 59.
  http://vita.had.co.nz/papers/tidy-data.pdf

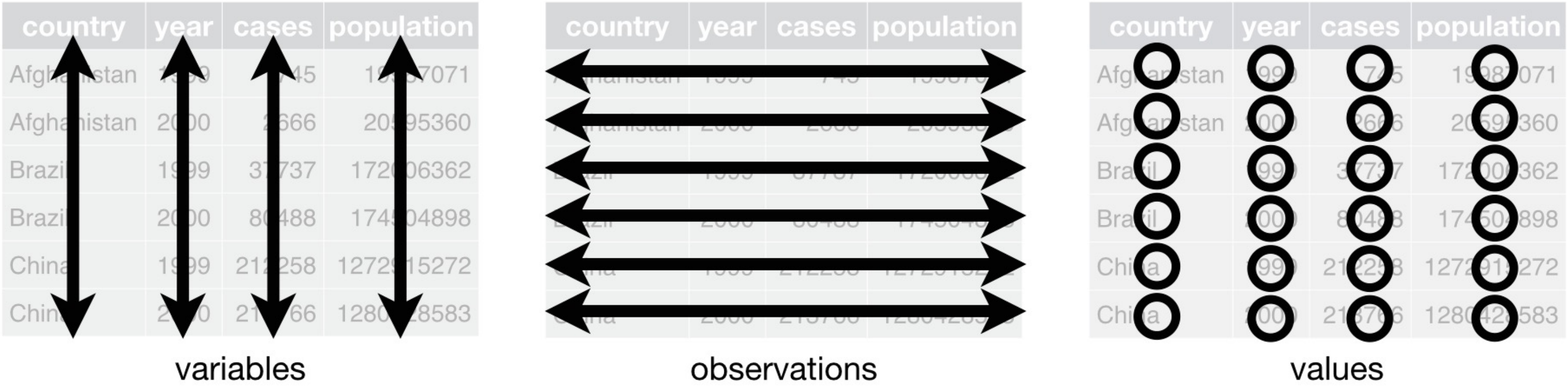
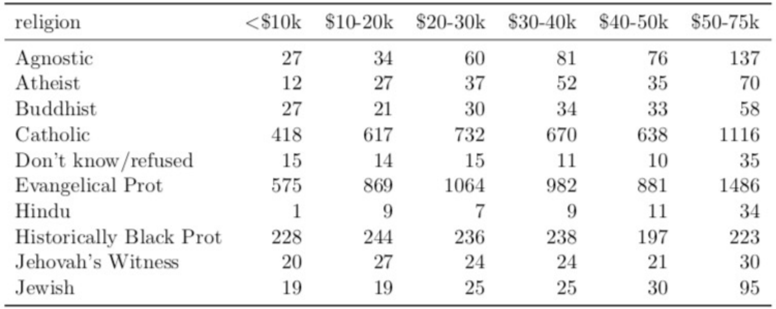


Figure from R4DS, p. 149 (https://r4ds.had.co.nz/tidy-data.html)

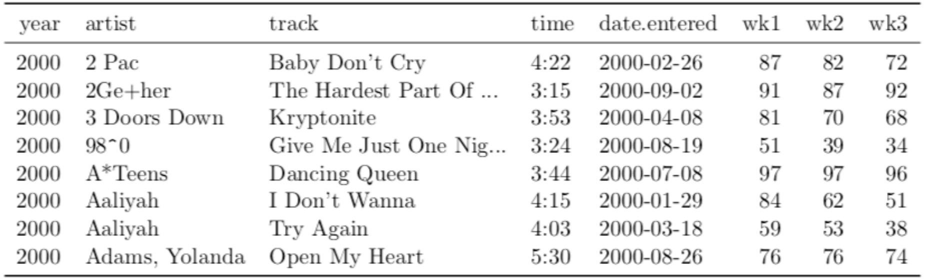
More on tidy data

- Why are these not tidy?



religion	<$10k	$10-20k	$20-30k	$30-40k	$40-50k	$50-75k
Agnostic	27	34	60	81	76	137
Atheist	12	27	37	52	35	70
Buddhist	27	21	30	34	33	58
Catholic	418	617	732	670	638	1116
Don't know/refused	15	14	15	11	10	35
Evangelical Prot	575	869	1064	982	881	1486
Hindu	1	9	7	9	11	34
Historically Black Prot	228	244	236	238	197	223
Jehovah's Witness	20	27	24	24	21	30
Jewish	19	19	25	25	30	95

Table 4: The first ten rows of data on income and religion from the Pew Forum. Three columns, `$75-100k`, `$100-150k` and `>150k`, have been omitted



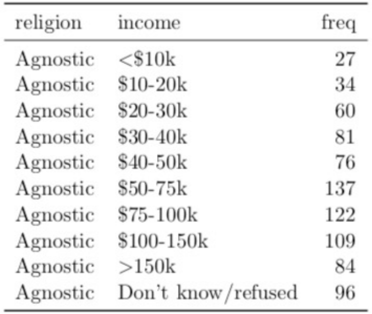
year	artist	track	time	date.entered	wk1	wk2	wk3
2000	2 Pac	Baby Don't Cry	4:22	2000-02-26	87	82	72
2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	91	87	92
2000	3 Doors Down	Kryptonite	3:53	2000-04-08	81	70	68
2000	98^0	Give Me Just One Nig...	3:24	2000-08-19	51	39	34
2000	A*Teens	Dancing Queen	3:44	2000-07-08	97	97	96
2000	Aaliyah	I Don't Wanna	4:15	2000-01-29	84	62	51
2000	Aaliyah	Try Again	4:03	2000-03-18	59	53	38
2000	Adams, Yolanda	Open My Heart	5:30	2000-08-26	76	76	74

Table 7: The first eight Billboard top hits for 2000. Other columns not shown are `wk4`, `wk5`, ..., `wk75`.

Tables from Wickham (2014) (http://vita.had.co.nz/papers/tidy-data.pdf)

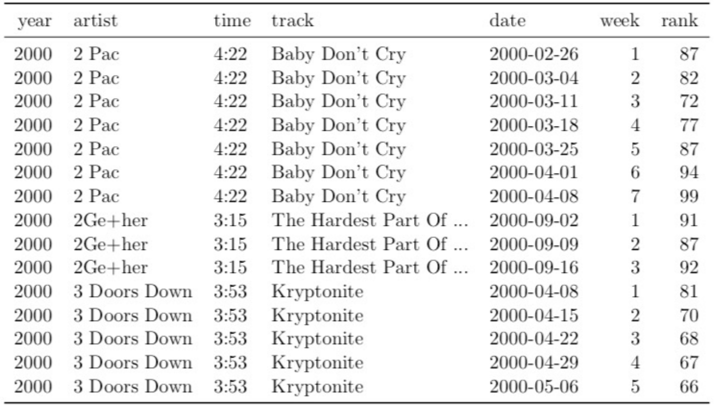
More on tidy data

- Are they now?



religion	income	freq
Agnostic	<$10k	27
Agnostic	$10-20k	34
Agnostic	$20-30k	60
Agnostic	$30-40k	81
Agnostic	$40-50k	76
Agnostic	$50-75k	137
Agnostic	$75-100k	122
Agnostic	$100-150k	109
Agnostic	>150k	84
Agnostic	Don't know/refused	96

Table 6: The first ten rows of the tidied Pew survey dataset on income and religion. The `column` has been renamed to `income`, and `value` to `freq`.



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`.

Tables from Wickham (2014) (http://vita.had.co.nz/papers/tidy-data.pdf)

Data Transformation

Inspect the data.

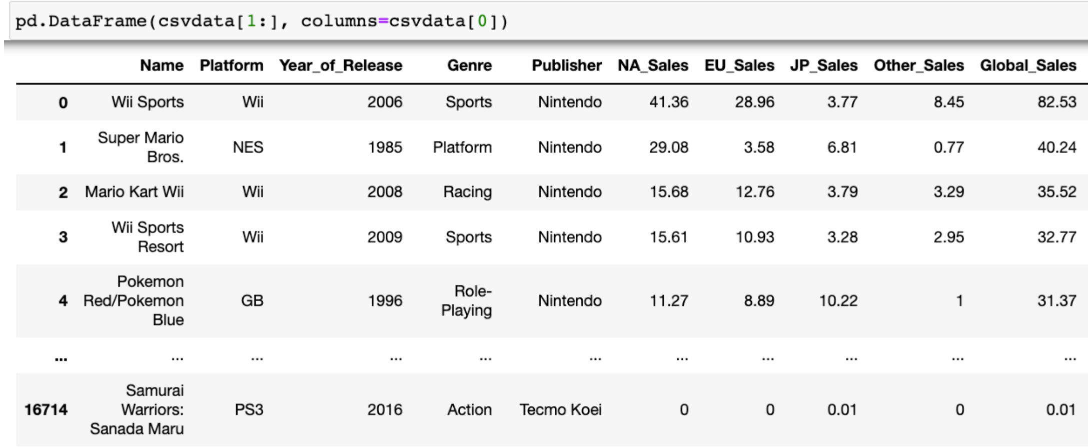
- Number of observations?
- Number of variables?
- What types of variables does it have?

```
count
count if platform=="PS4”
describe
edit
```

5 verbs of data transformation

- Column operations: `select (keep/drop), mutate (generate/replace)`
- Row operations: `filter (keep/drop), arrange (gsort)`
- Summarize: `summarize (summarize)`

- You select and mutate “variables”, filter and arrange “observations”.
- `mutate` = add or alter columns
  `arrange` = sort



```
pd.DataFrame(csvdata[1:], columns=csvdata[0])
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	Wii Sports	Wii	2006	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53
1	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52
3	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77
4	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	11.27	8.89	10.22	1	31.37
...	...	...	...	...	...	...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016	Action	Tecmo Koei	0	0	0.01	0	0.01

Select

- You select “columns” or “variables” NOT “rows” or “observations”.
- Let’s select these 5 columns:
  `Name, Platform, Year_of_Release, Genre, Global_Sales`.

```
preserve
keep name platform year_of_release genre global_sales
restore
```

Mutate

- You add a new column as a combination (or operation) of other columns.
  You can also alter a current column by creating a new column with the same name.
- Let's create a new column called:
  `Total_Sales = NA_Sales + EU_Sales + JP_Sales + Other_Sales`.

```
preserve
g total_sales = na_sales + eu_sales + jp_sales + other_sales
restore
```

Filter

- You can keep a subset of observations by filtering out others.
- Let’s collect videos games released on PS4 or Xbox One.

```
preserve
keep if platform=="PS4" | platform=="XOne"
count
restore
```

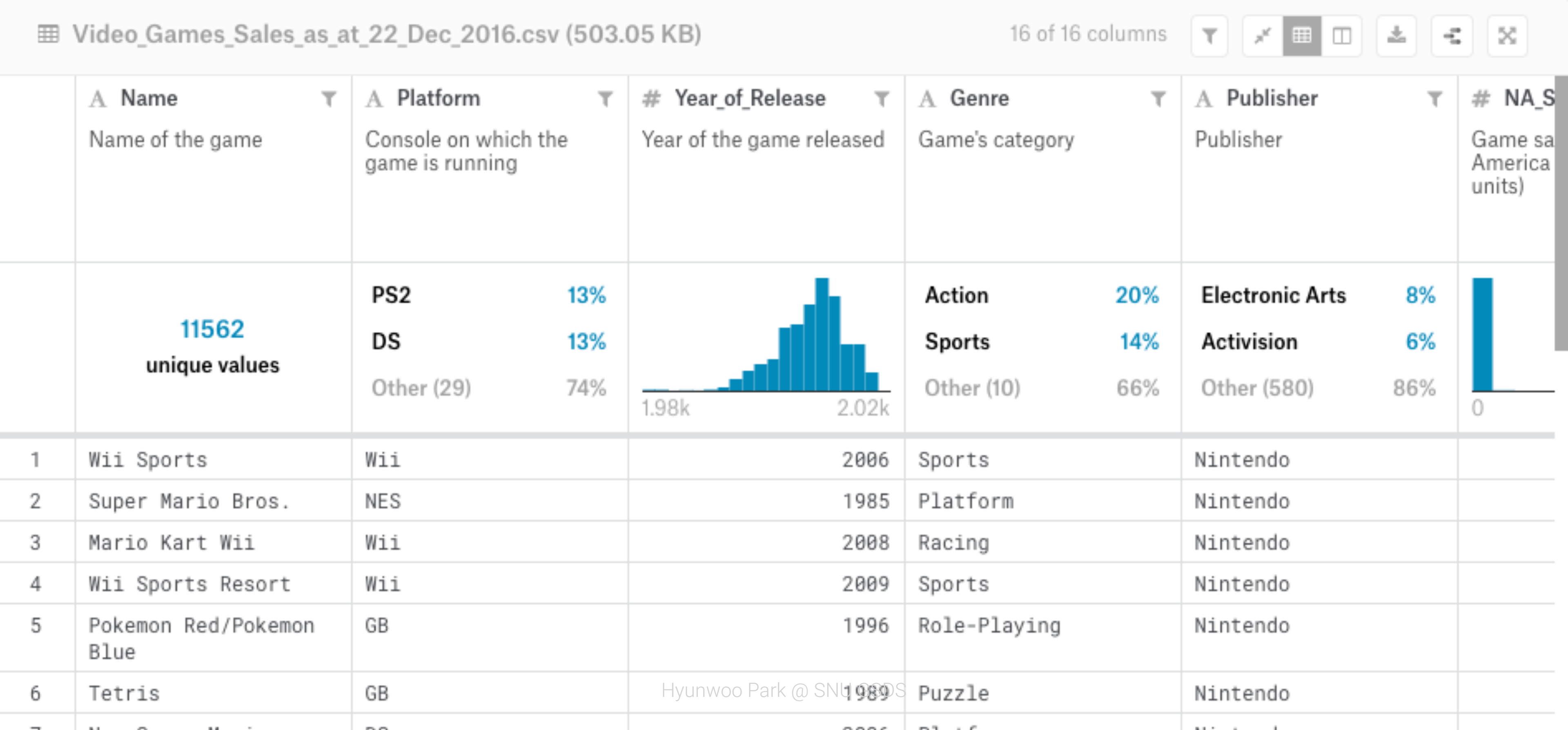
Arrange

- You sort rows (or observations) with one or more criteria.
- Let’s sort the data according to the following criteria in order:
    - (1) in descending order of `Year_of_Release`
    - (2) in ascending order of `Platform`
    - (3) in ascending order of `Name`

```
preserve
gsort -year_of_release platform name
edit if year_of_release !="N/A"
restore
```

Descriptive Statistics

A brief look at descriptive statistics and visualization



Video_Games_Sales_as_at_22_Dec_2016.csv (503.05 KB)

16 of 16 columns

	Name	Platform	Year_of_Release	Genre	Publisher	NA_S
	Name of the game	Console on which the game is running	Year of the game released	Game's category	Publisher	Game sa America units)
	11562 unique values	PS2 13% DS 13% Other (29) 74%	1.98k – 2.02k	Action 20% Sports 14% Other (10) 66%	Electronic Arts 8% Activision 6% Other (580) 86%	0
1	Wii Sports	Wii	2006	Sports	Nintendo	
2	Super Mario Bros.	NES	1985	Platform	Nintendo	
3	Mario Kart Wii	Wii	2008	Racing	Nintendo	
4	Wii Sports Resort	Wii	2009	Sports	Nintendo	
5	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	
6	Tetris	GB	1989	Puzzle	Nintendo	

Types of variables

- Categorical variables (or qualitative variable)
    - Platform: PS4, XOne, …
    - Genre: Action, Sports, …
- Numerical variables (or quantitative variable)
    - Discrete variables
        - Year_of_Release: 2006, 1985, 2008, 2009, …
    - Continuous variables
        - Global_Sales: 82.53, 40.24, 35.52, 32.77, …
- Understanding a single individual variable is to understand how it “varies” over different observations or measurements.
- Figuring out how observations are “distributed” along each of these variables is a good starting point.

Figuring out the distribution

- In essence, it is about counting observations that fall into a certain range.
- For categorical variable (and discrete variable sometimes), it's straightforward. And it's called tabulation.
- For continuous variable (and discrete variable sometimes), it requires another step before tabulation: binning.

Variable type	How to figure out the distribution?	Visualization
Categorical	Tabulation	Bar / Column Chart
Discrete	Both	Both
Continuous	Binning + Tabulation	Histogram, Density Plot, Boxplot

Tabulation for categorical variable

- You just need to count the number of observations for each category.
- Let’s count with `tab` function.
- Since categories don’t have intrinsic ordering,
  it’s usually useful to sort them by count in descending order.

```
tab platform
tab platform, sort
```

Binning + tabulation for continuous variable

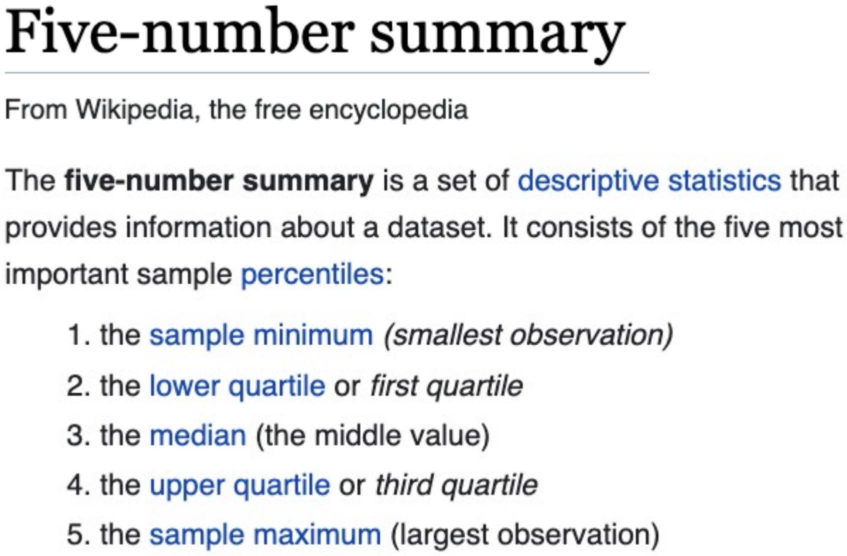
- For a continuous variable, you first need to discretize the variable into ranges.
- Three binning methods
  - makes groups of the given `width`
  - makes `n` groups with equal range
  - makes `n` groups with (approximately) equal numbers of observations

(Binning +) tabulation for discrete variable

- For a discrete variable, you can tabulate with or without binning.

Summary statistics for numerical variables

- Measure of location
  - Single representative numbers: mean, median
- Measure of spread
  - range, inter-quartile range, standard deviation
- Measure of rank
  - Five number summary
    - Minimum
    - First quartile
    - Median (= second quartile)
    - Third quartile
    - Maximum



Five-number summary

From Wikipedia, the free encyclopedia

The **five-number summary** is a set of descriptive statistics that provides information about a dataset. It consists of the five most important sample percentiles:

1. the sample minimum *(smallest observation)*
2. the lower quartile or *first quartile*
3. the median (the middle value)
4. the upper quartile or *third quartile*
5. the sample maximum (largest observation)

https://en.wikipedia.org/wiki/Five-number_summary

Summarize

- The fifth verb in data transformation is `summarize`.

```
replace year_of_release = "" if year_of_release=="N/A”
destring year_of_release, replace

su year_of_release
su year_of_release, d
```

Export data

```
outsheet using test.csv, c replace
```

Some other useful Stata commands & concepts

- egen (extensions to generate)
- collapse

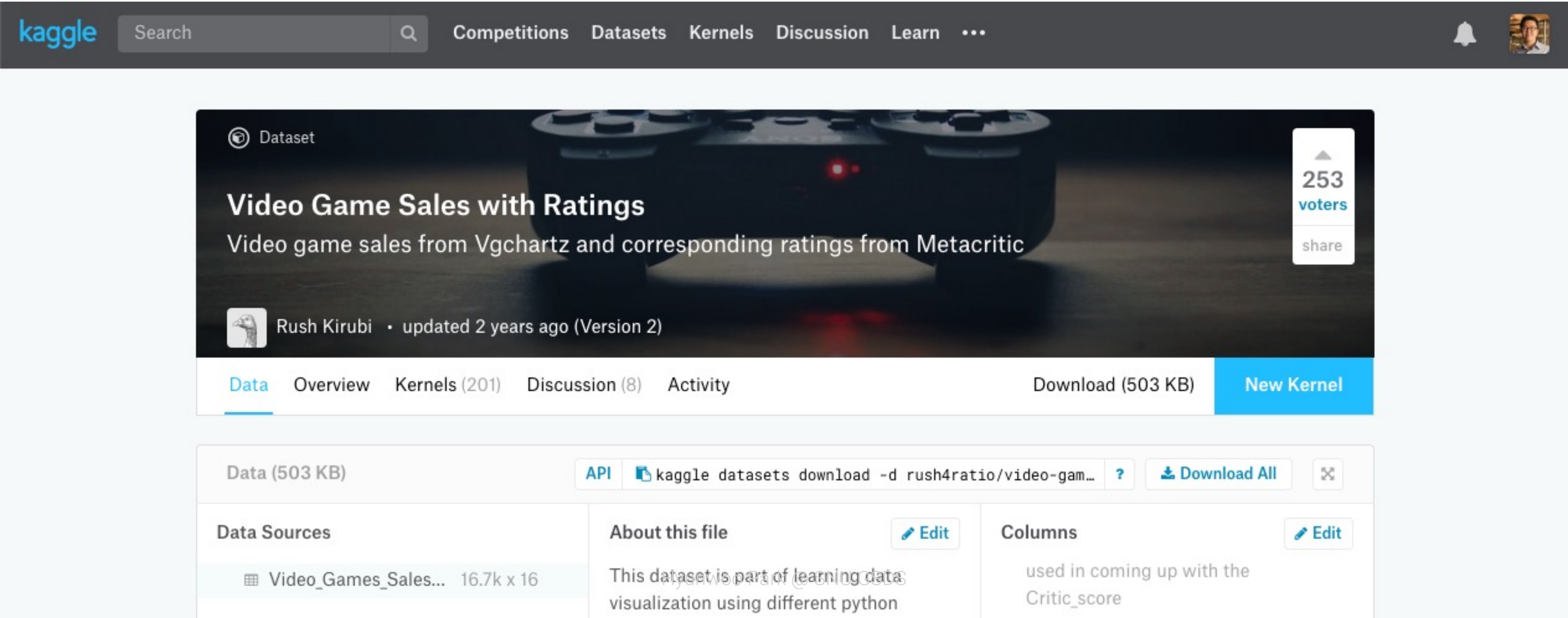
- Do files

- Long form vs wide form

Appendix:
Python-version

Let’s grab a dataset.

- Video Game Sales with Ratings
  https://www.kaggle.com/rush4ratio/video-game-sales-with-ratings



Video Game Sales with Ratings

Video game sales from Vgchartz and corresponding ratings from Metacritic

Rush Kirubi • updated 2 years ago (Version 2)

Data (503 KB)

```
kaggle datasets download -d rush4ratio/video-gam…
```

Data Sources

Video_Games_Sales... 16.7k x 16

About this file

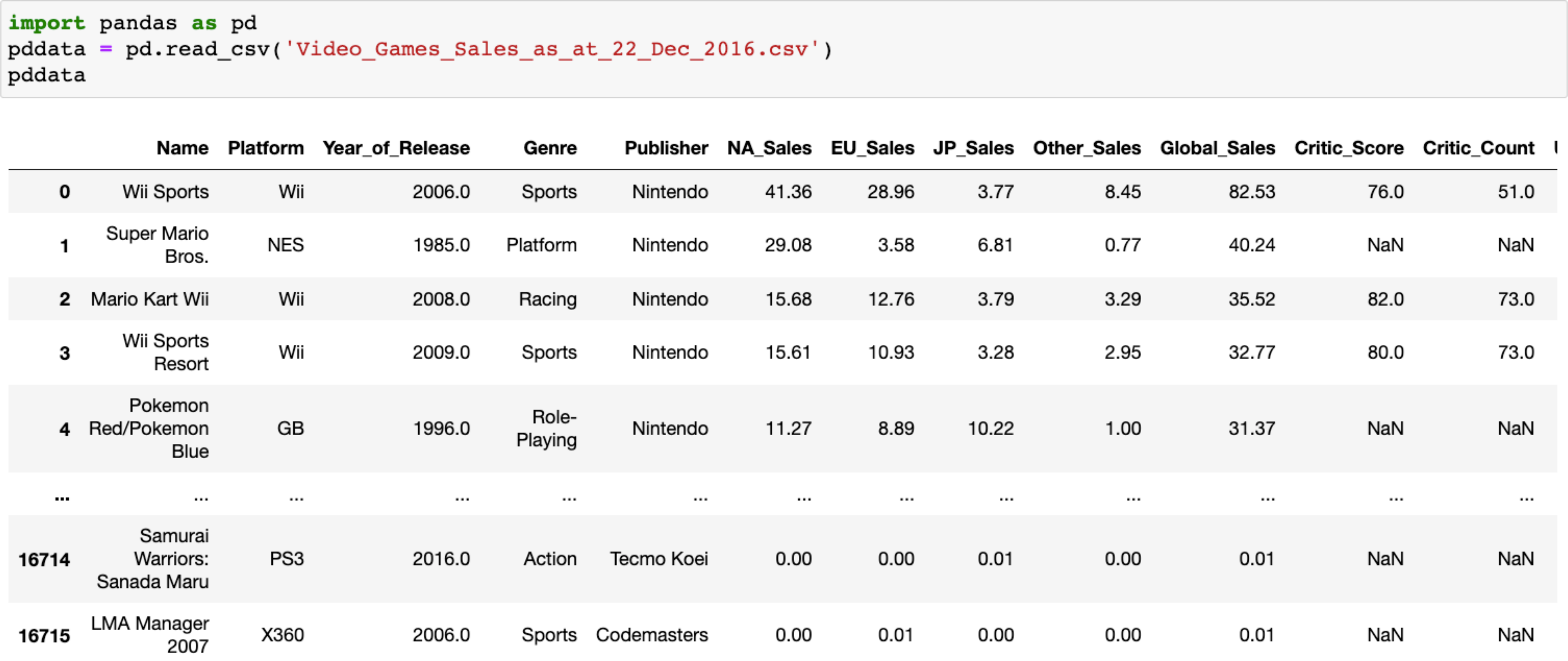
This dataset is part of learning data visualization using different python

Columns

used in coming up with the Critic_score

Data Loading

- Using pandas

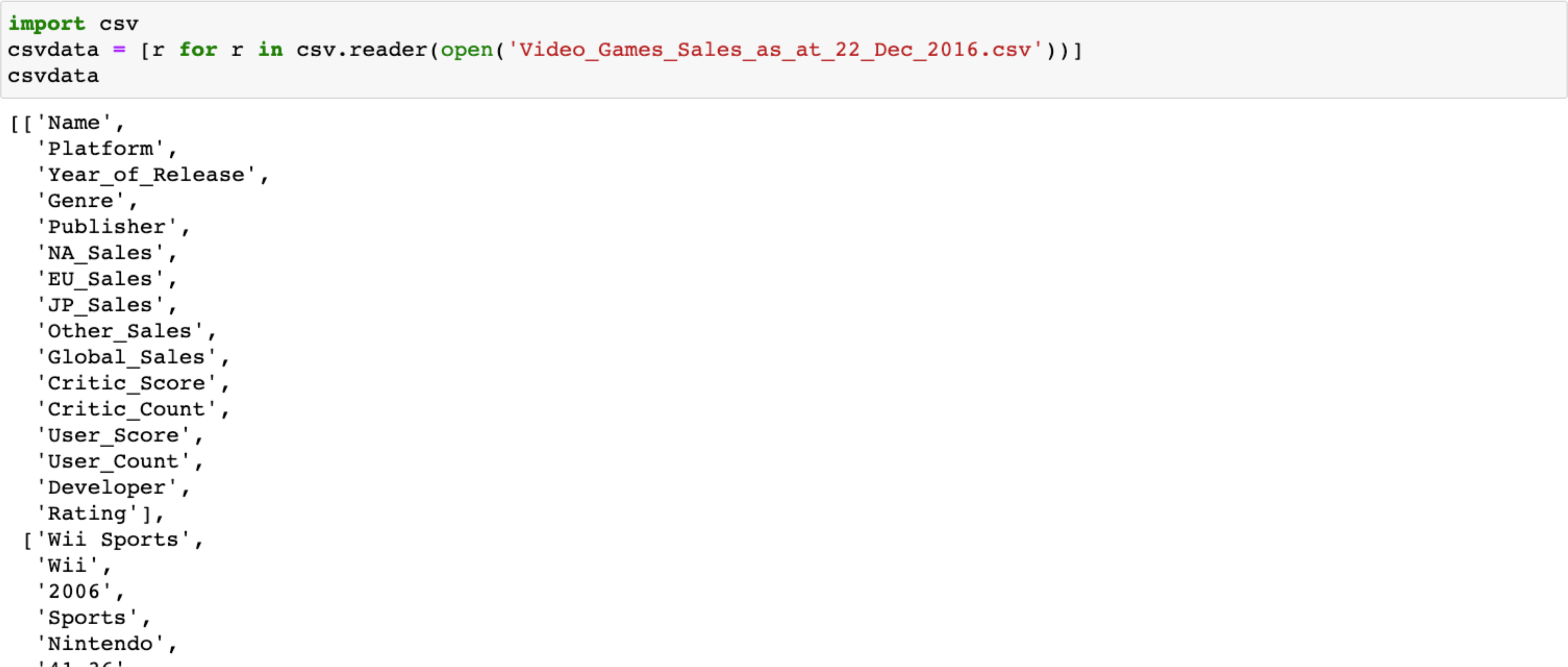


```python
import pandas as pd
pddata = pd.read_csv('Video_Games_Sales_as_at_22_Dec_2016.csv')
pddata
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016.0	Action	Tecmo Koei	0.00	0.00	0.01	0.00	0.01	NaN	NaN
16715	LMA Manager 2007	X360	2006.0	Sports	Codemasters	0.00	0.01	0.00	0.00	0.01	NaN	NaN

Data loading

- Using csv



```python
import csv
csvdata = [r for r in csv.reader(open('Video_Games_Sales_as_at_22_Dec_2016.csv'))]
csvdata
```

```
[['Name',
  'Platform',
  'Year_of_Release',
  'Genre',
  'Publisher',
  'NA_Sales',
  'EU_Sales',
  'JP_Sales',
  'Other_Sales',
  'Global_Sales',
  'Critic_Score',
  'Critic_Count',
  'User_Score',
  'User_Count',
  'Developer',
  'Rating'],
 ['Wii Sports',
  'Wii',
  '2006',
  'Sports',
  'Nintendo',
  '41.36',
```

What to use? pandas vs. csv

- pandas
  - Numeric values are automatically parsed out.
  - It returns a data frame object, which will work smoothly with numpy, jupyter notebook, and other numerical/scientific packages.
  - Column-oriented data structure.

  - [Pros] It comes with all the fancy helpers.
  - [Pros] You benefit from performance improvements of the package.
  - [Pros] You look like a data scientist.
  - [Pros] Your data science friends/collaborators are likely using it.

  - [Cons] It’s not designed for row-wise operations.
  - [Cons] Steeper learning curve

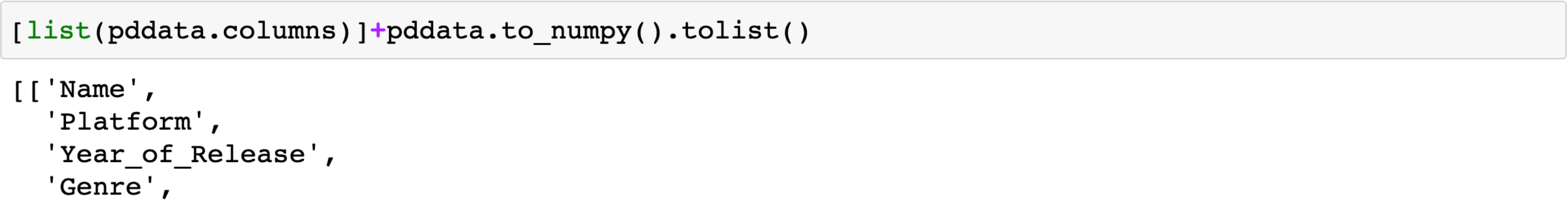
- csv
  - Everything is parsed as strings.
  - It returns a list of lists.
  - Row-oriented data structure.

  - [Pros] Transparent
  - [Pros] Easier to code for processing row by row

  - [Cons] You have to do the housekeeping work yourself.
  - [Cons] Easier to make mistake (You have to validate your data yourself.)
  - [Cons] You look like a primate in data science.

Going back and forth between DataFrame and list of lists

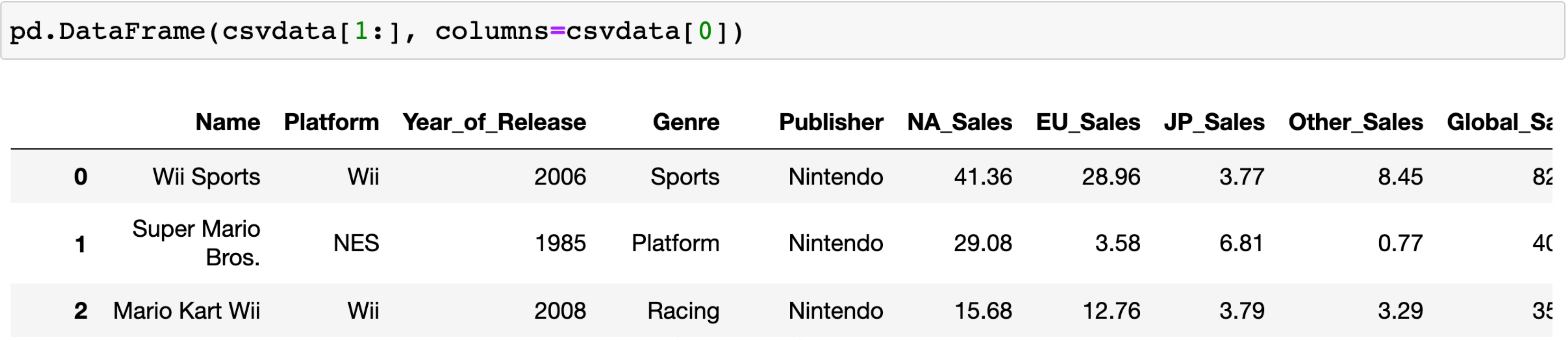
- DataFrame to LoL



```
[list(pddata.columns)]+pddata.to_numpy().tolist()
```

```
[['Name',
  'Platform',
  'Year_of_Release',
  'Genre',
```

- LoL to DataFrame



```
pd.DataFrame(csvdata[1:], columns=csvdata[0])
```

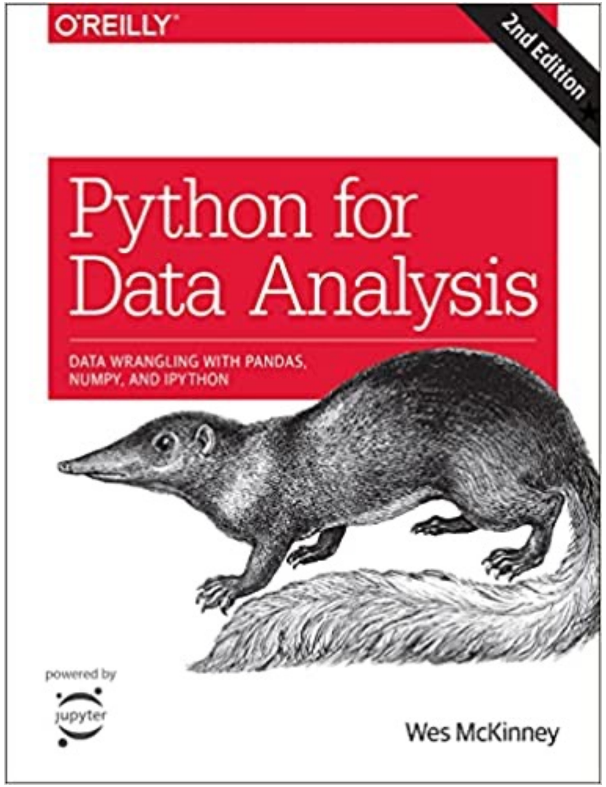
	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sa
0	Wii Sports	Wii	2006	Sports	Nintendo	41.36	28.96	3.77	8.45	82
1	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40
2	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.68	12.76	3.79	3.29	35

Your own library of helpers

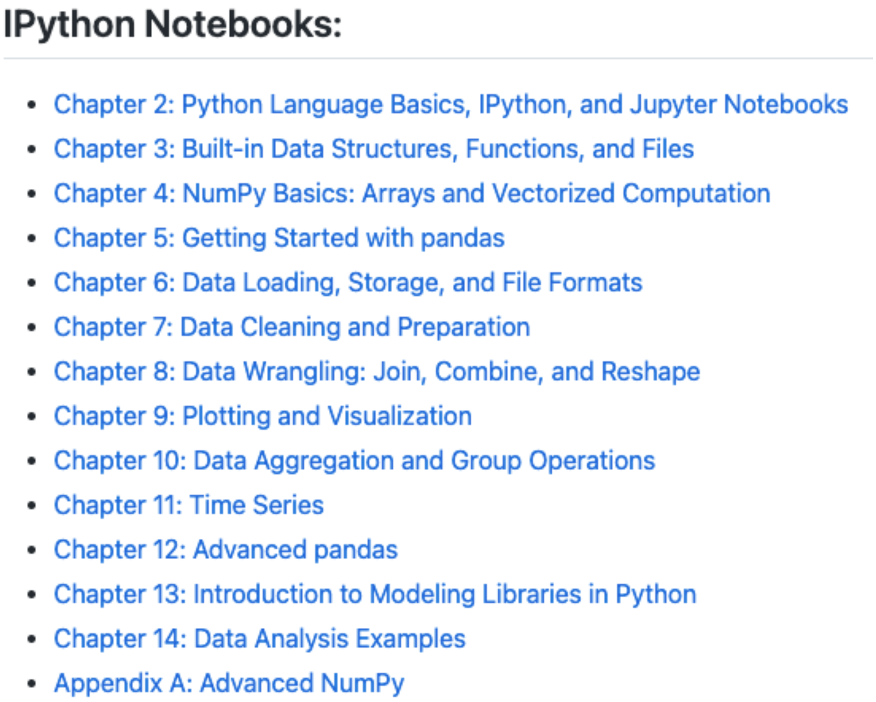
- Add the folder containing this file to PYTHON_PATH.
  (Or place it in the same folder of your working directory.)

```python
import pandas as pd
import csv

def pd2csv(df): return [list(df.columns)]+df.to_numpy().tolist()
def csv2pd(lol): return pd.DataFrame(lol[1:], columns=lol[0])

def read_csv(fname):
  print('Reading', fname)
  return list(csv.reader(open(fname, 'r')))

def write_csv(fname, data):
  print('Writing', fname)
  output = csv.writer(open(fname, 'w'))
  output.writerows(data)
```

Resources for pandas



- https://github.com/wesm/pydata-book



IPython Notebooks:

- Chapter 2: Python Language Basics, IPython, and Jupyter Notebooks
- Chapter 3: Built-in Data Structures, Functions, and Files
- Chapter 4: NumPy Basics: Arrays and Vectorized Computation
- Chapter 5: Getting Started with pandas
- Chapter 6: Data Loading, Storage, and File Formats
- Chapter 7: Data Cleaning and Preparation
- Chapter 8: Data Wrangling: Join, Combine, and Reshape
- Chapter 9: Plotting and Visualization
- Chapter 10: Data Aggregation and Group Operations
- Chapter 11: Time Series
- Chapter 12: Advanced pandas
- Chapter 13: Introduction to Modeling Libraries in Python
- Chapter 14: Data Analysis Examples
- Appendix A: Advanced NumPy

What is tidy data?

- It depends on the data context.
- Variables are easier to be linked; observations are harder.
- Further explanation on tidy data:
  Wickham, H. (2014). Tidy data. Journal of Statistical Software, 59.
  http://vita.had.co.nz/papers/tidy-data.pdf

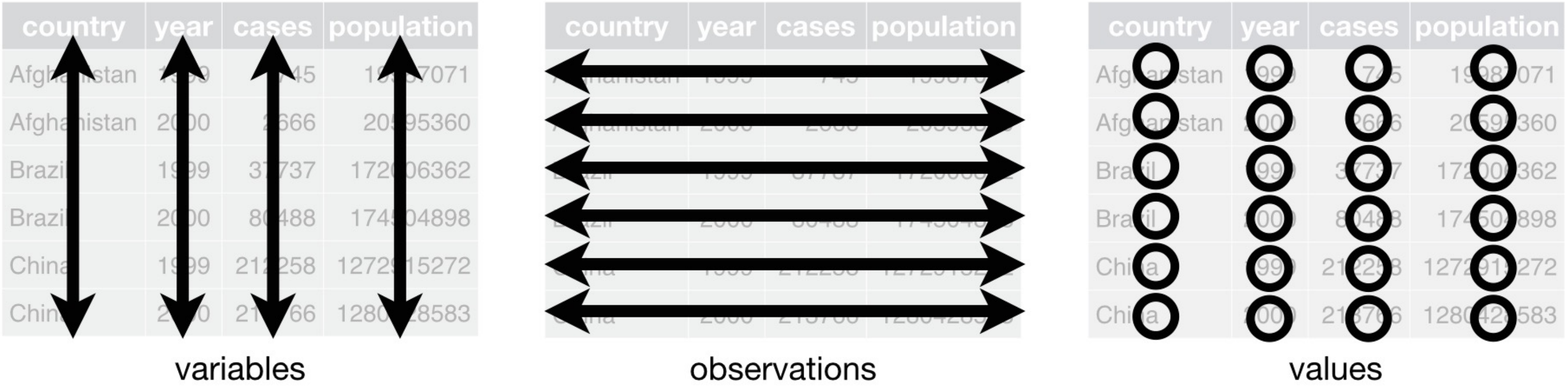
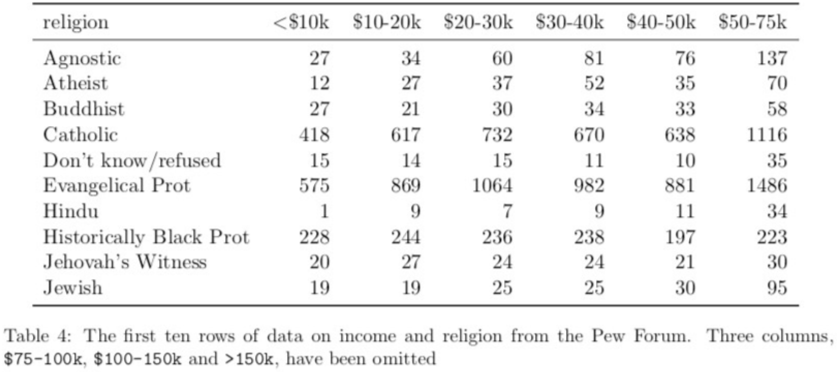


Figure from R4DS, p. 149 (https://r4ds.had.co.nz/tidy-data.html)

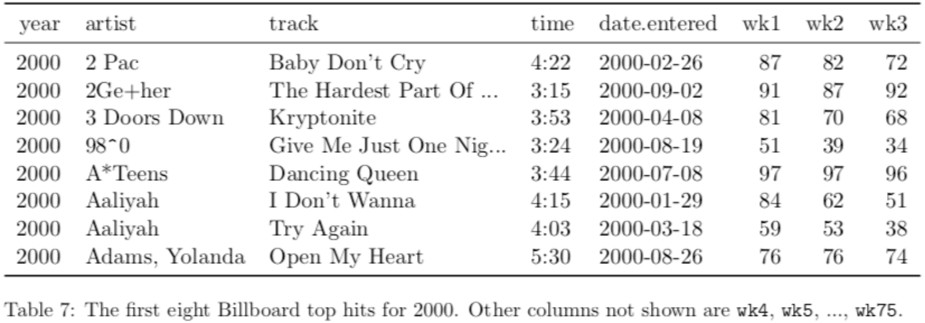
More on tidy data

- Why are these not tidy?



religion	<$10k	$10-20k	$20-30k	$30-40k	$40-50k	$50-75k
Agnostic	27	34	60	81	76	137
Atheist	12	27	37	52	35	70
Buddhist	27	21	30	34	33	58
Catholic	418	617	732	670	638	1116
Don't know/refused	15	14	15	11	10	35
Evangelical Prot	575	869	1064	982	881	1486
Hindu	1	9	7	9	11	34
Historically Black Prot	228	244	236	238	197	223
Jehovah's Witness	20	27	24	24	21	30
Jewish	19	19	25	25	30	95

Table 4: The first ten rows of data on income and religion from the Pew Forum. Three columns, `$75-100k`, `$100-150k` and `>150k`, have been omitted



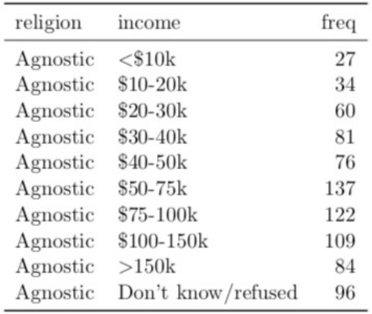
year	artist	track	time	date.entered	wk1	wk2	wk3
2000	2 Pac	Baby Don't Cry	4:22	2000-02-26	87	82	72
2000	2Ge+her	The Hardest Part Of ...	3:15	2000-09-02	91	87	92
2000	3 Doors Down	Kryptonite	3:53	2000-04-08	81	70	68
2000	98^0	Give Me Just One Nig...	3:24	2000-08-19	51	39	34
2000	A*Teens	Dancing Queen	3:44	2000-07-08	97	97	96
2000	Aaliyah	I Don't Wanna	4:15	2000-01-29	84	62	51
2000	Aaliyah	Try Again	4:03	2000-03-18	59	53	38
2000	Adams, Yolanda	Open My Heart	5:30	2000-08-26	76	76	74

Table 7: The first eight Billboard top hits for 2000. Other columns not shown are `wk4`, `wk5`, ..., `wk75`.

Tables from Wickham (2014) (http://vita.had.co.nz/papers/tidy-data.pdf)

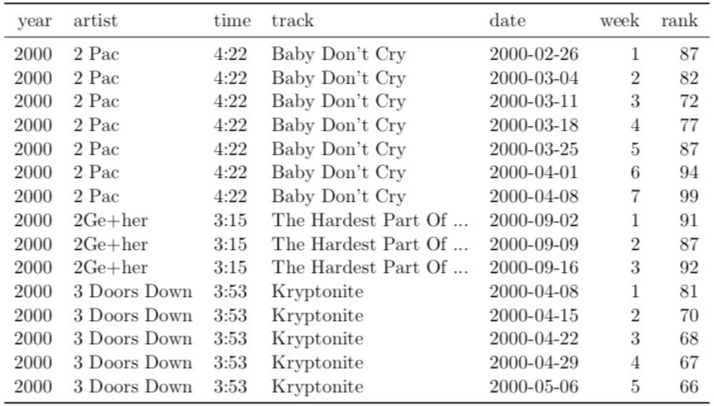
More on tidy data

- Are they now?



religion	income	freq
Agnostic	<$10k	27
Agnostic	$10-20k	34
Agnostic	$20-30k	60
Agnostic	$30-40k	81
Agnostic	$40-50k	76
Agnostic	$50-75k	137
Agnostic	$75-100k	122
Agnostic	$100-150k	109
Agnostic	>150k	84
Agnostic	Don't know/refused	96

Table 6: The first ten rows of the tidied Pew survey dataset on income and religion. The `column` has been renamed to `income`, and `value` to `freq`.



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

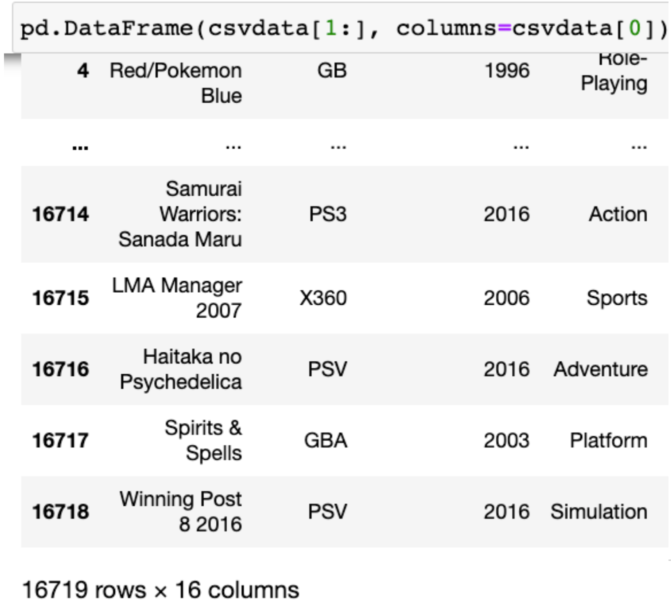
Table 8: First fifteen rows of the tidied billboard dataset. The `date` column does not appear in the original table, but can be computed from `date.entered` and `week`.

Tables from Wickham (2014) (http://vita.had.co.nz/papers/tidy-data.pdf)

Data Transformation

Inspect the data.

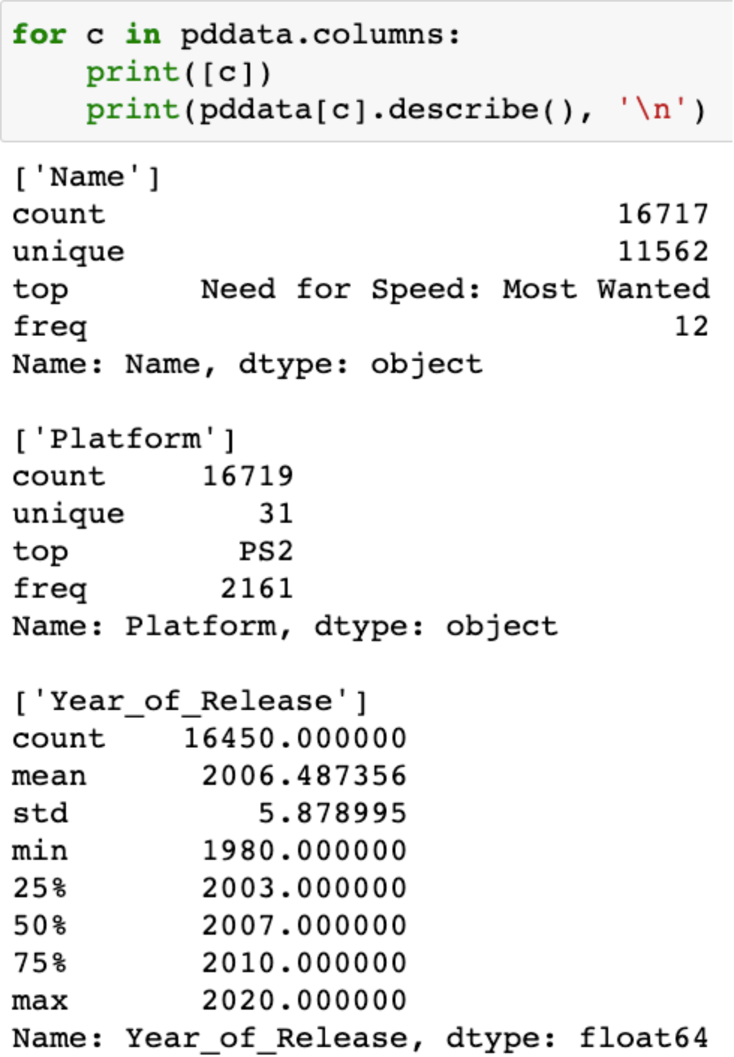
- Number of observations?
- Number of variables?
- What types of variables does it have?



```
pd.DataFrame(csvdata[1:], columns=csvdata[0])
```

4	Red/Pokemon Blue	GB	1996	Role-Playing
...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016	Action
16715	LMA Manager 2007	X360	2006	Sports
16716	Haitaka no Psychedelica	PSV	2016	Adventure
16717	Spirits & Spells	GBA	2003	Platform
16718	Winning Post 8 2016	PSV	2016	Simulation

16719 rows × 16 columns



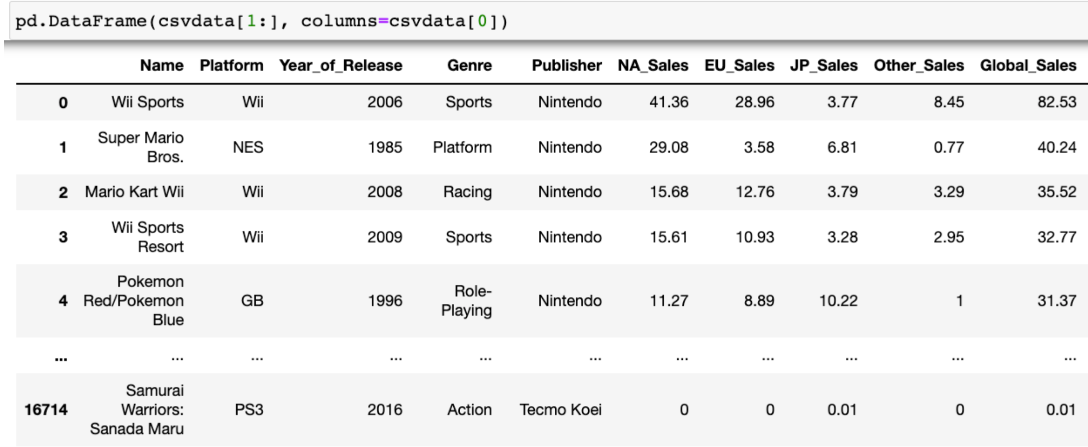
```
for c in pddata.columns:
    print([c])
    print(pddata[c].describe(), '\n')
```

```
['Name']
count                          16717
unique                         11562
top           Need for Speed: Most Wanted
freq                              12
Name: Name, dtype: object

['Platform']
count      16719
unique        31
top          PS2
freq        2161
Name: Platform, dtype: object

['Year_of_Release']
count     16450.000000
mean       2006.487356
std           5.878995
min        1980.000000
25%        2003.000000
50%        2007.000000
75%        2010.000000
max        2020.000000
Name: Year_of_Release, dtype: float64
```

5 verbs of data transformation in tidyverse

- Column operations: `select, mutate`
- Row operations: `filter, arrange`
- Summarize: `summarize`

- You select and mutate "variables", filter and arrange "observations".
- `mutate` = add or alter columns
  `arrange` = sort

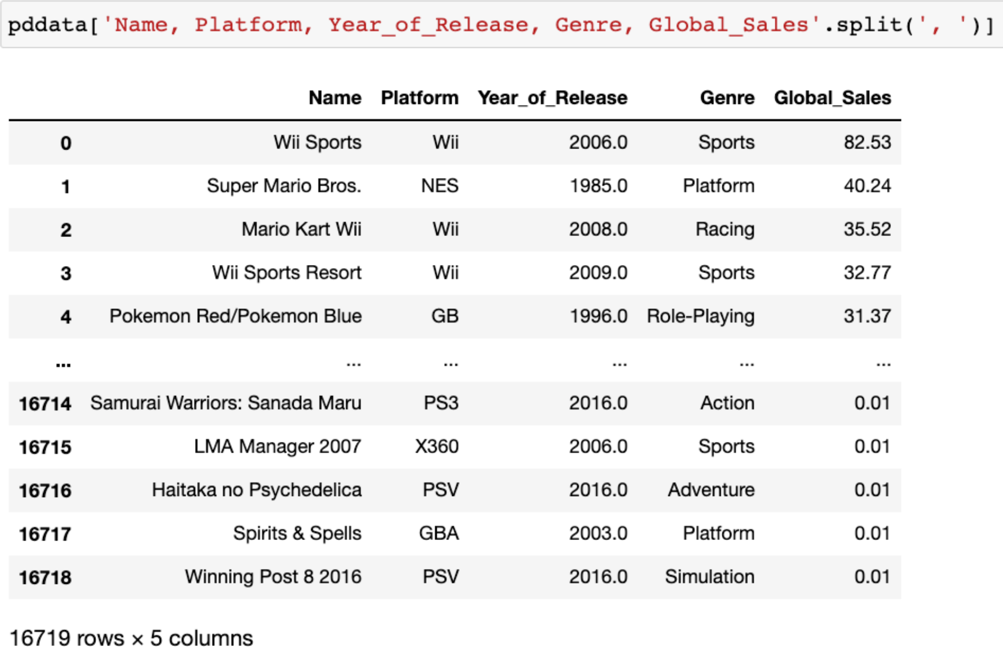


```
pd.DataFrame(csvdata[1:], columns=csvdata[0])
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales
0	Wii Sports	Wii	2006	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53
1	Super Mario Bros.	NES	1985	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24
2	Mario Kart Wii	Wii	2008	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52
3	Wii Sports Resort	Wii	2009	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77
4	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	11.27	8.89	10.22	1	31.37
...	...	...	...	...	...	...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016	Action	Tecmo Koei	0	0	0.01	0	0.01

Select

- You select “columns” or “variables” NOT “rows” or “observations”.
- Let’s select these 5 columns:
  `Name, Platform, Year_of_Release, Genre, Global_Sales`.



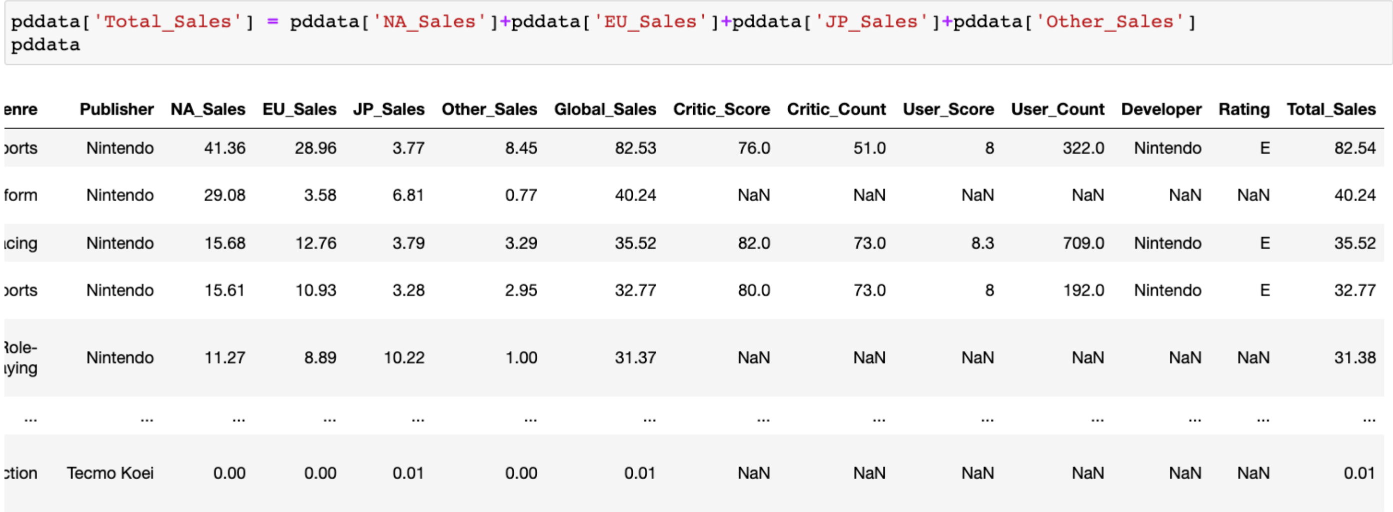
```
pddata['Name, Platform, Year_of_Release, Genre, Global_Sales'.split(', ')]
```

	Name	Platform	Year_of_Release	Genre	Global_Sales
0	Wii Sports	Wii	2006.0	Sports	82.53
1	Super Mario Bros.	NES	1985.0	Platform	40.24
2	Mario Kart Wii	Wii	2008.0	Racing	35.52
3	Wii Sports Resort	Wii	2009.0	Sports	32.77
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	31.37
...	...	...	...	...	...
16714	Samurai Warriors: Sanada Maru	PS3	2016.0	Action	0.01
16715	LMA Manager 2007	X360	2006.0	Sports	0.01
16716	Haitaka no Psychedelica	PSV	2016.0	Adventure	0.01
16717	Spirits & Spells	GBA	2003.0	Platform	0.01
16718	Winning Post 8 2016	PSV	2016.0	Simulation	0.01

16719 rows × 5 columns

Mutate

- You add a new column as a combination (or operation) of other columns.
  You can also alter a current column by creating a new column with the same name.
- Let's create a new column called:
  `Total_Sales = NA_Sales + EU_Sales + JP_Sales + Other_Sales`.

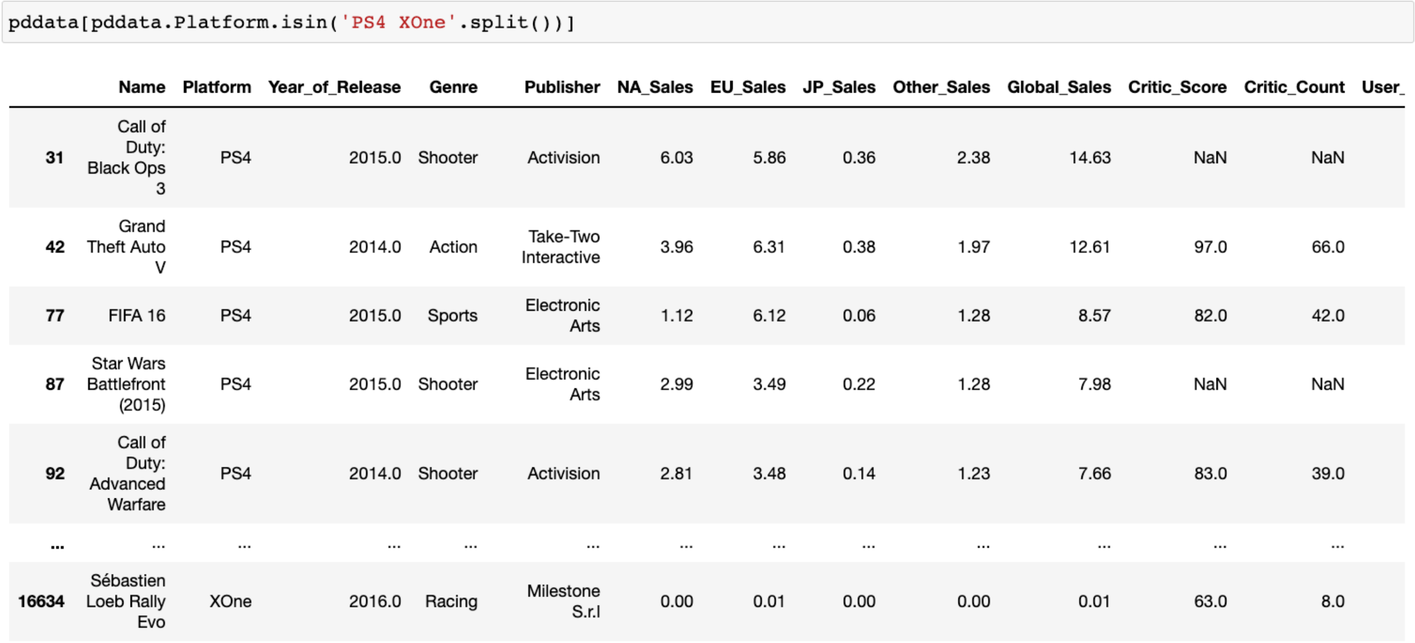


```
pddata['Total_Sales'] = pddata['NA_Sales']+pddata['EU_Sales']+pddata['JP_Sales']+pddata['Other_Sales']
pddata
```

enre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating	Total_Sales
orts	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E	82.54
form	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN	NaN	NaN	NaN	NaN	40.24
cing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E	35.52
orts	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E	32.77
ole- ying	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	NaN	NaN	NaN	NaN	31.38
...	...	...	...	...	...	...	...	...	...	...	...	...	...
ction	Tecmo Koei	0.00	0.00	0.01	0.00	0.01	NaN	NaN	NaN	NaN	NaN	NaN	0.01

Filter

- You can keep a subset of observations by filtering out others.
- Let’s collect videos games released on PS4 or Xbox One.

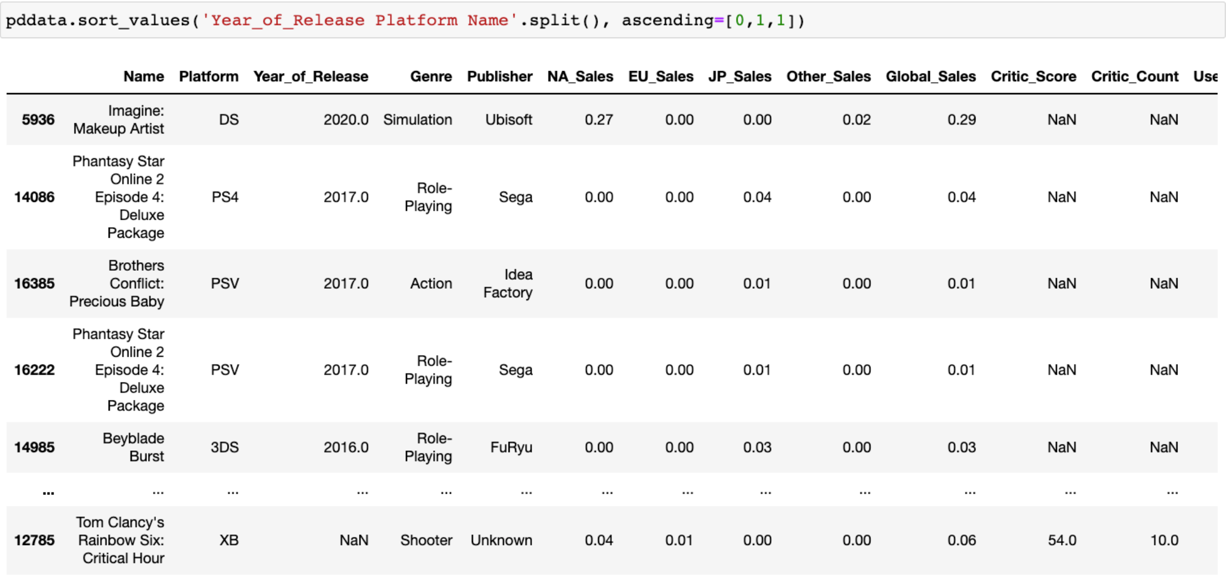


```
pddata[pddata.Platform.isin('PS4 XOne'.split())]
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_
31	Call of Duty: Black Ops 3	PS4	2015.0	Shooter	Activision	6.03	5.86	0.36	2.38	14.63	NaN	NaN	
42	Grand Theft Auto V	PS4	2014.0	Action	Take-Two Interactive	3.96	6.31	0.38	1.97	12.61	97.0	66.0	
77	FIFA 16	PS4	2015.0	Sports	Electronic Arts	1.12	6.12	0.06	1.28	8.57	82.0	42.0	
87	Star Wars Battlefront (2015)	PS4	2015.0	Shooter	Electronic Arts	2.99	3.49	0.22	1.28	7.98	NaN	NaN	
92	Call of Duty: Advanced Warfare	PS4	2014.0	Shooter	Activision	2.81	3.48	0.14	1.23	7.66	83.0	39.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
16634	Sébastien Loeb Rally Evo	XOne	2016.0	Racing	Milestone S.r.l	0.00	0.01	0.00	0.00	0.01	63.0	8.0	

Arrange

- You sort rows (or observations) with one or more criteria.
- Let’s sort the data according to the following criteria in order:
  - (1) in descending order of `Year_of_Release`
  - (2) in ascending order of `Platform`
  - (3) in ascending order of `Name`



```
pddata.sort_values('Year_of_Release Platform Name'.split(), ascending=[0,1,1])
```

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	Use
5936	Imagine: Makeup Artist	DS	2020.0	Simulation	Ubisoft	0.27	0.00	0.00	0.02	0.29	NaN	NaN	
14086	Phantasy Star Online 2 Episode 4: Deluxe Package	PS4	2017.0	Role-Playing	Sega	0.00	0.00	0.04	0.00	0.04	NaN	NaN	
16385	Brothers Conflict: Precious Baby	PSV	2017.0	Action	Idea Factory	0.00	0.00	0.01	0.00	0.01	NaN	NaN	
16222	Phantasy Star Online 2 Episode 4: Deluxe Package	PSV	2017.0	Role-Playing	Sega	0.00	0.00	0.01	0.00	0.01	NaN	NaN	
14985	Beyblade Burst	3DS	2016.0	Role-Playing	FuRyu	0.00	0.00	0.03	0.00	0.03	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	
12785	Tom Clancy's Rainbow Six: Critical Hour	XB	NaN	Shooter	Unknown	0.04	0.01	0.00	0.00	0.06	54.0	10.0	

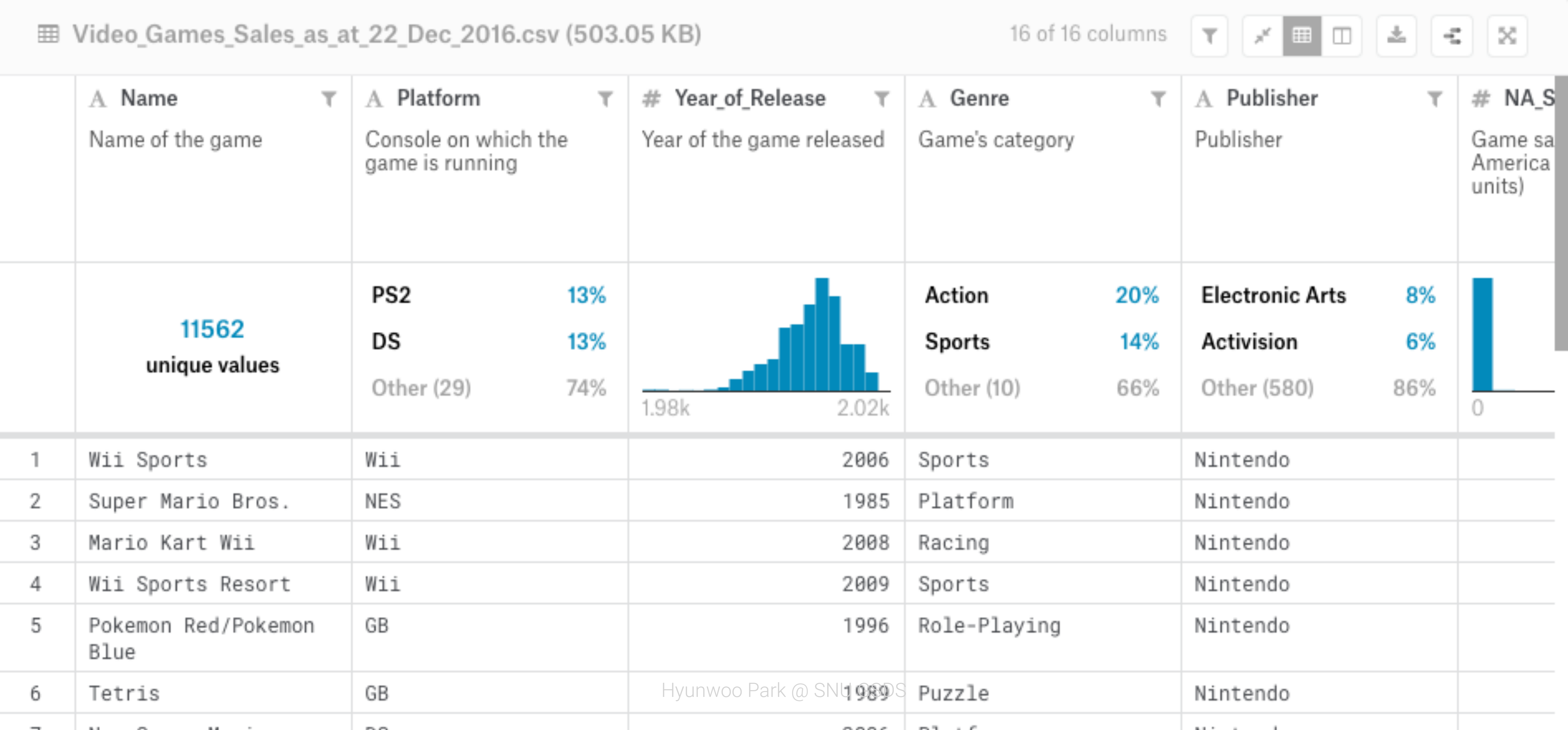
Same transformations with list of lists from csv

- Use just whatever way more intuitive and convenient for you.

```python
# select
[[r[c] for c in [0,1,2,3,9]] for r in csvdata]

# mutate
for i, r in enumerate(csvdata):
  if i==0: r.append('Total_Sales'); continue
  r.append(sum([float(r[c]) for c in range(5,9)]))

# filter
[r for i, r in enumerate(csvdata) if i==0 or r[1] in 'PS4 XOne'.split()]

# arrange
keys = [(2,1), (1,0), (0,0)]
temp = csvdata
for k in reversed(keys):
  temp = [temp[0]]+sorted(temp[1:], key=lambda r: r[k[0]], reverse=k[1])
```

Descriptive Statistics

A brief look at descriptive statistics and visualization



Video_Games_Sales_as_at_22_Dec_2016.csv (503.05 KB)

16 of 16 columns

	A Name	A Platform	# Year_of_Release	A Genre	A Publisher	# NA_S
	Name of the game	Console on which the game is running	Year of the game released	Game's category	Publisher	Game sa America units)
	11562 unique values	PS2 13% DS 13% Other (29) 74%	1.98k 2.02k	Action 20% Sports 14% Other (10) 66%	Electronic Arts 8% Activision 6% Other (580) 86%	0
1	Wii Sports	Wii	2006	Sports	Nintendo	
2	Super Mario Bros.	NES	1985	Platform	Nintendo	
3	Mario Kart Wii	Wii	2008	Racing	Nintendo	
4	Wii Sports Resort	Wii	2009	Sports	Nintendo	
5	Pokemon Red/Pokemon Blue	GB	1996	Role-Playing	Nintendo	
6	Tetris	GB	1989	Puzzle	Nintendo	

Types of variables

- Categorical variables (or qualitative variable)
    - Platform: PS4, XOne, …
    - Genre: Action, Sports, …
- Numerical variables (or quantitative variable)
    - Discrete variables
        - Year_of_Release: 2006, 1985, 2008, 2009, …
    - Continuous variables
        - Global_Sales: 82.53, 40.24, 35.52, 32.77, …
- Understanding a single individual variable is to understand how it “varies” over different observations or measurements.
- Figuring out how observations are “distributed” along each of these variables is a good starting point.

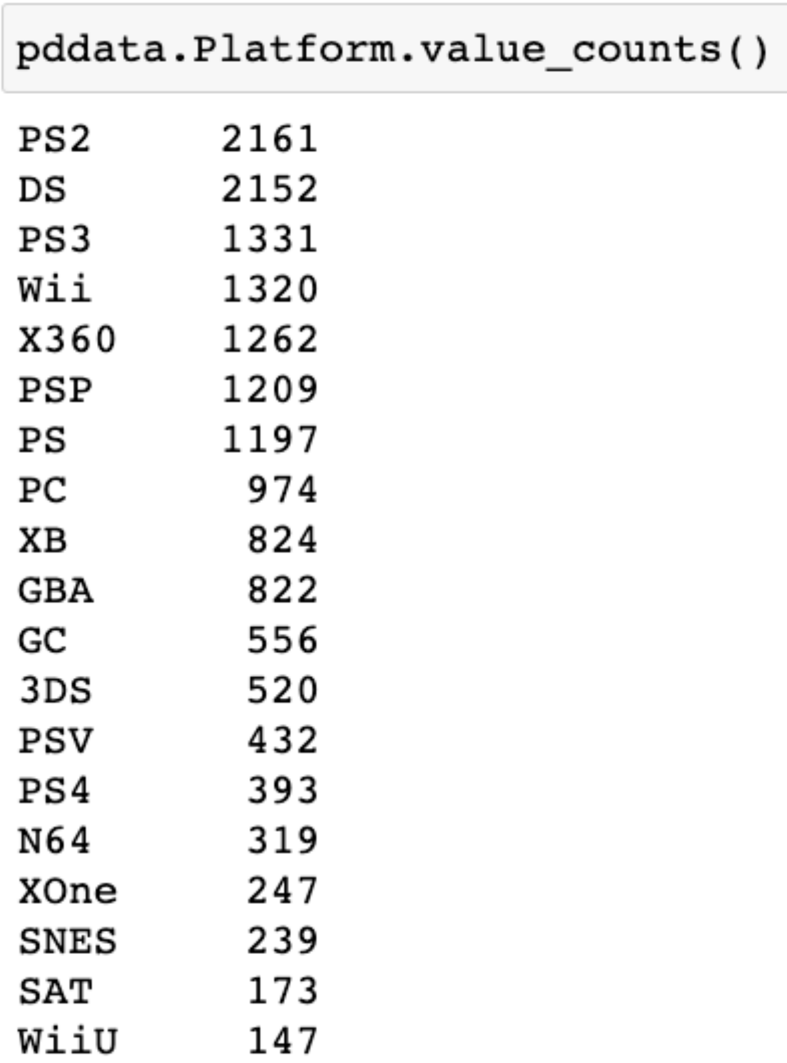
Figuring out the distribution

- In essence, it is about counting observations that fall into a certain range.
- For categorical variable (and discrete variable sometimes), it's straightforward. And it's called tabulation.
- For continuous variable (and discrete variable sometimes), it requires another step before tabulation: binning.

Variable type	How to figure out the distribution?	Visualization
Categorical	Tabulation	Bar / Column Chart
Discrete	Both	Both
Continuous	Binning + Tabulation	Histogram, Density Plot, Boxplot

Tabulation for categorical variable

- You just need to count the number of observations for each category.
- Let’s count with `value_counts` function.
- Since categories don’t have intrinsic ordering,
  it’s usually useful to sort them by count in descending order.



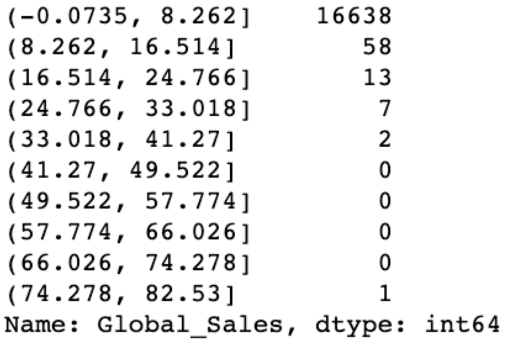
```
pddata.Platform.value_counts()
```

```
PS2     2161
DS      2152
PS3     1331
Wii     1320
X360    1262
PSP     1209
PS      1197
PC       974
XB       824
GBA      822
GC       556
3DS      520
PSV      432
PS4      393
N64      319
XOne     247
SNES     239
SAT      173
WiiU     147
```

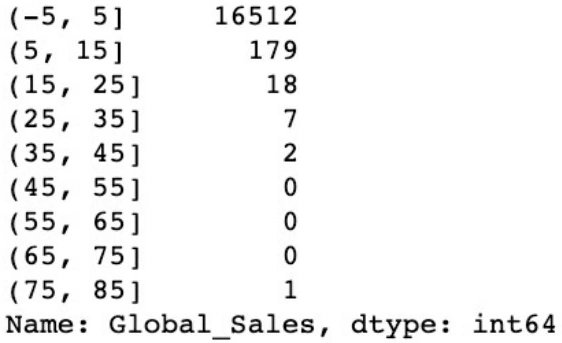
Binning + tabulation for continuous variable

- For a continuous variable, you first need to discretize the variable into ranges.
- Three binning methods
  - makes groups of the given `width`
  - makes **n** groups with equal range
  - makes **n** groups with (approximately) equal numbers of observations

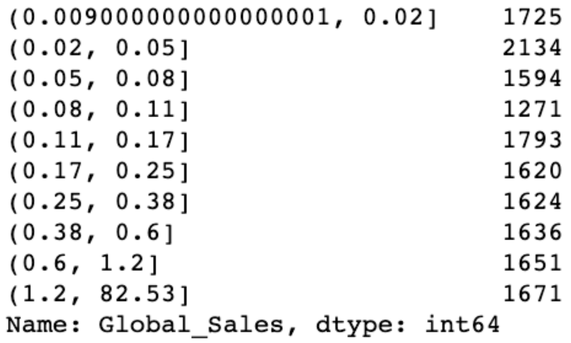
```
pddata['Global_Sales'].value_counts(bins=10, sort=False)
pddata['Global_Sales'].value_counts(bins=pd.interval_range(start=-5, end=85, freq=10), sort=False)
pd.qcut(pddata['Global_Sales'], q=10).value_counts(sort=False)
```



```
(-0.0735, 8.262]    16638
(8.262, 16.514]        58
(16.514, 24.766]       13
(24.766, 33.018]        7
(33.018, 41.27]         2
(41.27, 49.522]         0
(49.522, 57.774]        0
(57.774, 66.026]        0
(66.026, 74.278]        0
(74.278, 82.53]         1
Name: Global_Sales, dtype: int64
```



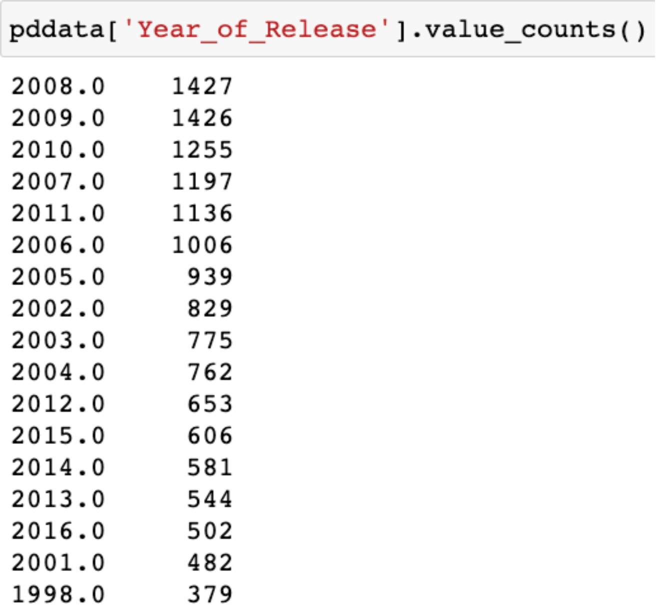
```
(-5, 5]     16512
(5, 15]       179
(15, 25]       18
(25, 35]        7
(35, 45]        2
(45, 55]        0
(55, 65]        0
(65, 75]        0
(75, 85]        1
Name: Global_Sales, dtype: int64
```



```
(0.009000000000000001, 0.02]    1725
(0.02, 0.05]                    2134
(0.05, 0.08]                    1594
(0.08, 0.11]                    1271
(0.11, 0.17]                    1793
(0.17, 0.25]                    1620
(0.25, 0.38]                    1624
(0.38, 0.6]                     1636
(0.6, 1.2]                      1651
(1.2, 82.53]                    1671
Name: Global_Sales, dtype: int64
```

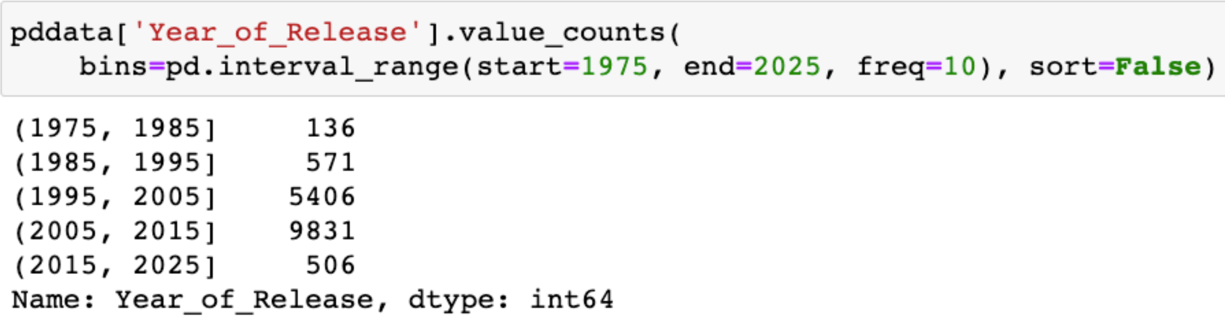
(Binning +) tabulation for discrete variable

- For a discrete variable, you can tabulate with or without binning.



```
pddata['Year_of_Release'].value_counts()
```

```
2008.0    1427
2009.0    1426
2010.0    1255
2007.0    1197
2011.0    1136
2006.0    1006
2005.0     939
2002.0     829
2003.0     775
2004.0     762
2012.0     653
2015.0     606
2014.0     581
2013.0     544
2016.0     502
2001.0     482
1998.0     379
```

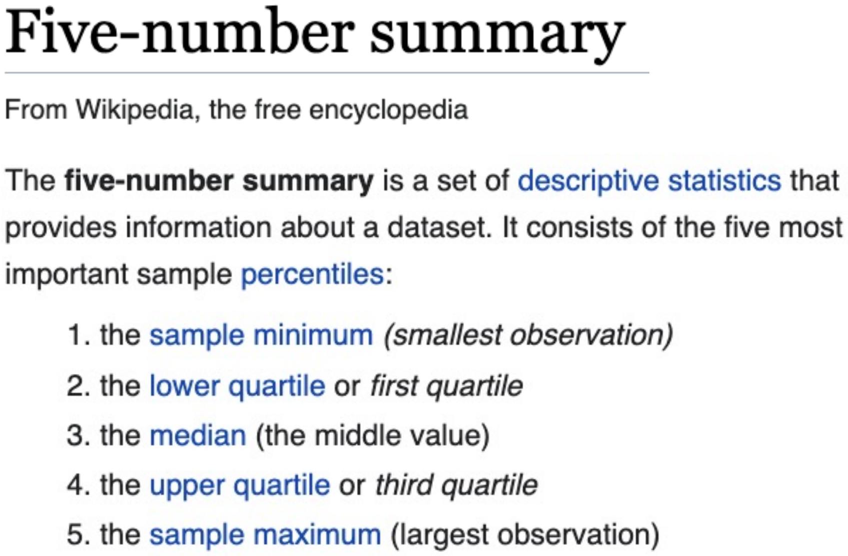


```
pddata['Year_of_Release'].value_counts(
    bins=pd.interval_range(start=1975, end=2025, freq=10), sort=False)
```

```
(1975, 1985]     136
(1985, 1995]     571
(1995, 2005]    5406
(2005, 2015]    9831
(2015, 2025]     506
Name: Year_of_Release, dtype: int64
```

Summary statistics for numerical variables

- Measure of location
  - Single representative numbers: mean, median
- Measure of spread
  - range, inter-quartile range, standard deviation
- Measure of rank
  - Five number summary
    - Minimum
    - First quartile
    - Median (= second quartile)
    - Third quartile
    - Maximum



Five-number summary

From Wikipedia, the free encyclopedia

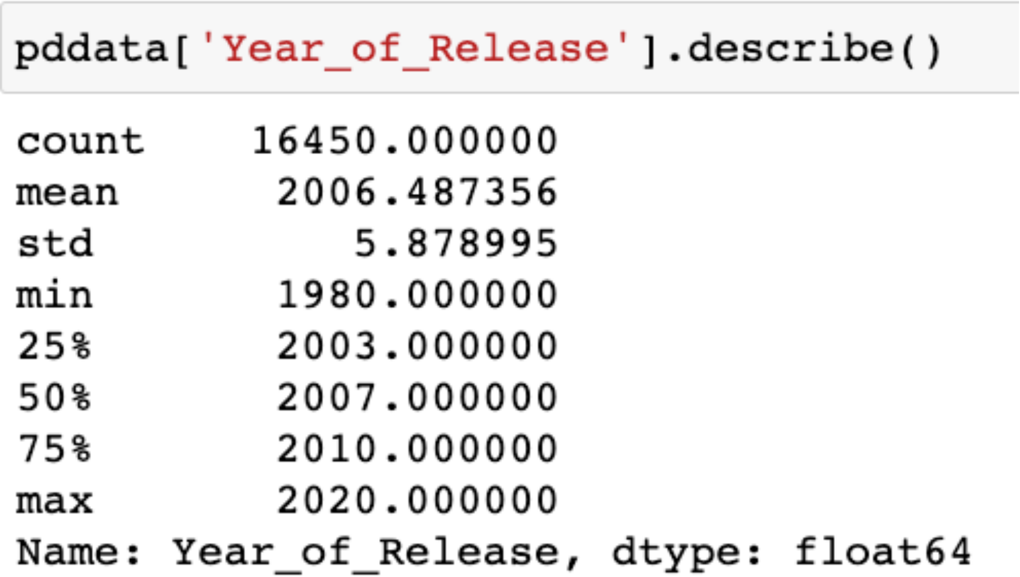
The **five-number summary** is a set of descriptive statistics that provides information about a dataset. It consists of the five most important sample percentiles:

1. the sample minimum *(smallest observation)*
2. the lower quartile or *first quartile*
3. the median (the middle value)
4. the upper quartile or *third quartile*
5. the sample maximum (largest observation)

https://en.wikipedia.org/wiki/Five-number_summary

Summarize

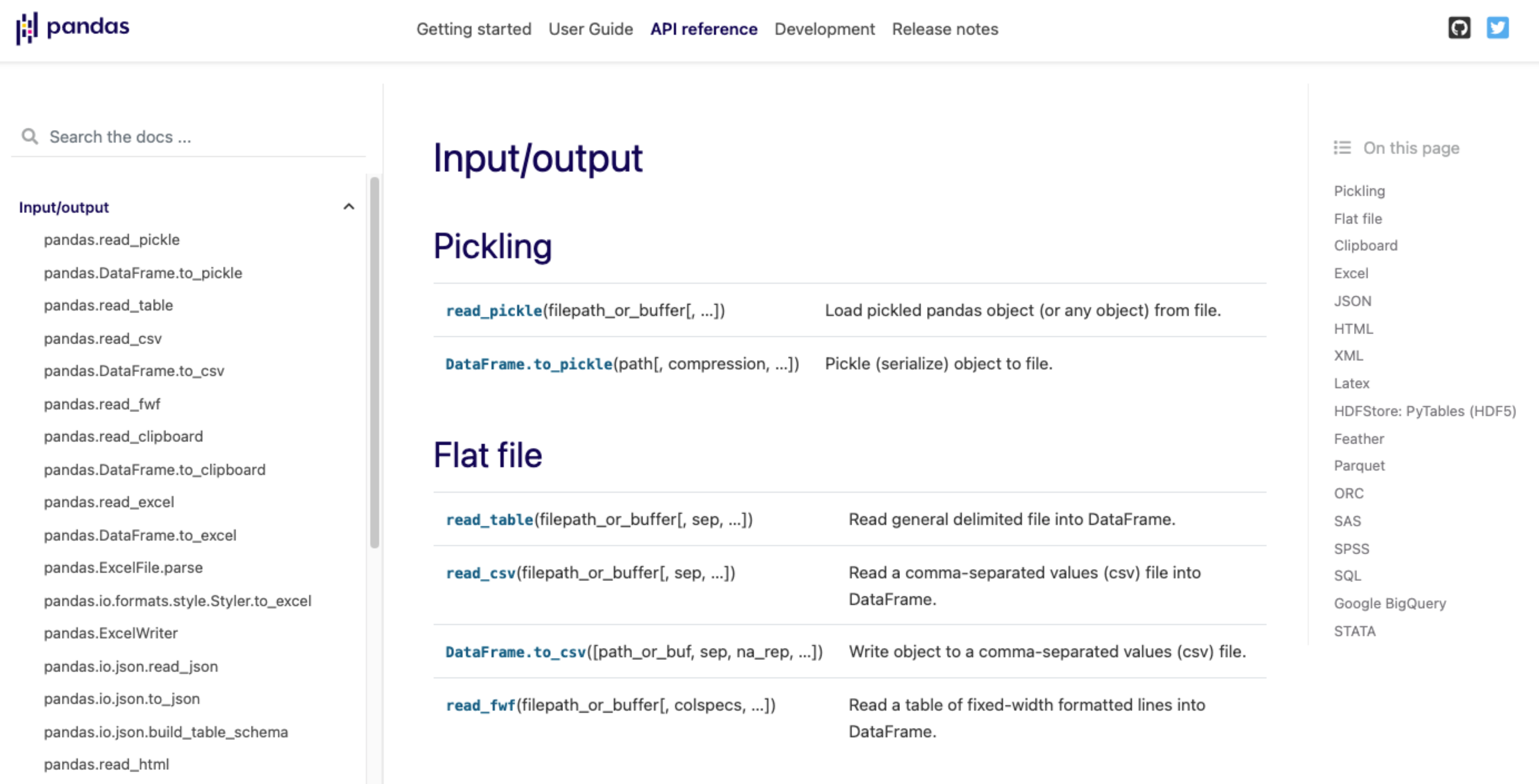
- The fifth verb in data transformation is `summarize`, In pandas, you use `describe`.



```
pddata['Year_of_Release'].describe()
```

```
count    16450.000000
mean      2006.487356
std          5.878995
min       1980.000000
25%       2003.000000
50%       2007.000000
75%       2010.000000
max       2020.000000
Name: Year_of_Release, dtype: float64
```

Export data



Input/output

Pickling

`read_pickle`(filepath_or_buffer[, ...])	Load pickled pandas object (or any object) from file.
`DataFrame.to_pickle`(path[, compression, ...])	Pickle (serialize) object to file.

Flat file

`read_table`(filepath_or_buffer[, sep, ...])	Read general delimited file into DataFrame.
`read_csv`(filepath_or_buffer[, sep, ...])	Read a comma-separated values (csv) file into DataFrame.
`DataFrame.to_csv`([path_or_buf, sep, na_rep, ...])	Write object to a comma-separated values (csv) file.
`read_fwf`(filepath_or_buffer[, colspecs, ...])	Read a table of fixed-width formatted lines into DataFrame.

Common discrete distributions

Distribution	R function	Example
Bernoulli	`rbernoulli`	Head or tail in a coin toss
Rectangular (or uniform)		Number of 1's in n dice rolls
Binomial	`rbinom`	Number of heads in n coin tosses
Geometric	`rgeom`	Number of failures until the first success
Negative binomial	`rnbinom`	Number of failures until nth success
Poisson	`rpois`	Number of events if occurrences are independent from each other
Zipf		Number of occurrences of words in texts

Common continuous distributions

Distribution	R function	Example
Uniform	`runif`	Random number within a range
Normal	`rnorm`	Height of people in a population
Exponential	`rexp`	Length of time between independent events
Lognormal	`rlnorm`	Length of comments, system repair times, income distribution
Pareto		Wealth distribution, city size, size of meteorites, 80/20 rule

Create a new DataFrame with random variables

```python
from numpy.random import default_rng
rng = default_rng()
N = 1000
data = [
  rng.binomial(1,.1,size=N),
  rng.integers(1,7,size=N),
  rng.binomial(10,.1,size=N),
  rng.geometric(.1,size=N),
  rng.negative_binomial(10,.1,size=N),
  rng.poisson(10,size=N),
  rng.uniform(10,20,size=N),
  rng.normal(10,2,size=N),
  rng.exponential(.1,size=N),
  rng.lognormal(2,2,size=N)
]
cnames = 'Bernoulli RandInt Binomial Geometric NBinom Poisson Uniform Normal Exp LogNormal'.split()
randdf = pd.DataFrame({cnames[i]: data[i] for i in range(len(data))})
randdf
```

- Try to compute the summary statistics table about this randomly generated dataset.