



**190F**  
Fall 2018

# Foundations of Data Science

## Lecture 5

---

Working with Tables

# Ways to create a table

---

- `Table.read_table(filename)` - reads a table from a spreadsheet
- `Table()` - an empty table
- and...

# Arrays → Tables

---

- `Table().with_column(label, data)` - creates a table with a single column; `data` is an array
  - `Table().with_columns(label1, data1, ...)` - creates a table, with an array of data for each column
-

# Table Methods

---

- Extending tables:
  - `t.with_columns` and `t.read_table`

# Examples

---

The table `students` has columns `Name`, `ID`, and `Score`.  
Write one line of code that evaluates to:

a) A table consisting of only the column labeled `Name`

```
students.select('Name')
```

b) The largest score

```
students.column('Score').max()  
max(students.column('Score'))
```

---

# Lists are Generic Sequences

---

A list is a sequence of values (just like an array), but the values can all have different types

```
[2+3, 'four', Table().with_column('K', [3, 4])]
```

If you create a table column from a list, it will be converted to an array automatically

---

# Minard's Map

# Charles Joseph Minard, 1781-1870

---



- French civil engineer who some say created one of the greatest graphs of all time.
  - Visualized Napoleon's 1812 invasion of Russia, including the number of soldiers, the direction of the march, names and locations of cities, temperatures, dates.
-



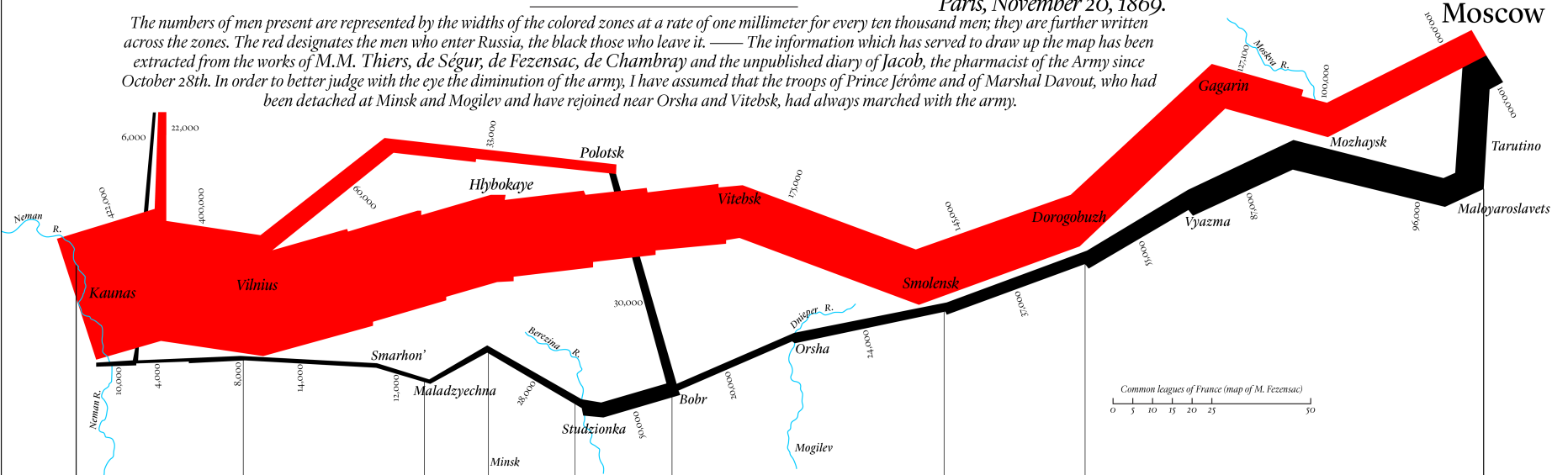
# Visualization of 1812 March

## Figurative Map of the successive losses in men of the French Army in the Russian campaign 1812 ~ 1813

Drawn by M. Minard, Inspector General of Bridges and Roads (retired).

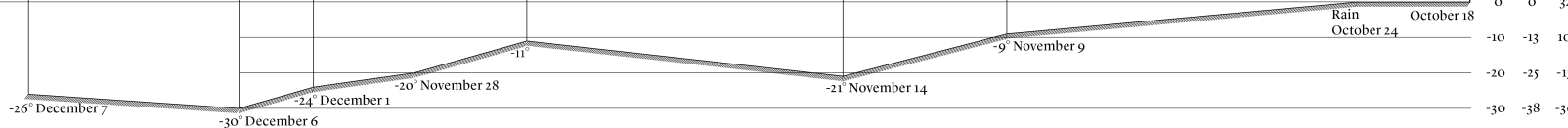
Paris, November 20, 1869.

The numbers of men present are represented by the widths of the colored zones at a rate of one millimeter for every ten thousand men; they are further written across the zones. The red designates the men who enter Russia, the black those who leave it. — The information which has served to draw up the map has been extracted from the works of M.M. Thiers, de Ségur, de Fezensac, de Chambray and the unpublished diary of Jacob, the pharmacist of the Army since October 28th. In order to better judge with the eye the diminution of the army, I have assumed that the troops of Prince Jérôme and of Marshal Davout, who had been detached at Minsk and Mogilev and have rejoined near Orsha and Vitebsk, had always marched with the army.



## GRAPHIC TABLE of the temperature in degrees below zero of the Réaumur thermometer.

The Cossacks pass the frozen Neman in a gallop.



# Different types of data

---

**float:**  
decimal number

Longitude	Latitude	City	Direction	Survivors
32	54.8	Smolensk	Advance	145000
33.2	54.9	Dorogobouge	Advance	140000
34.4	55.5	Chjat	Advance	127100
37.6	55.8	Moscou	Advance	100000
34.3	55.2	Wixma	Retreat	55000
32	54.6	Smolensk	Retreat	24000
30.4	54.4	Orscha	Retreat	20000
26.8	54.3	Moiodexno	Retreat	12000

**string:**  
text

**int:**  
integer

# More Tables

# Table Methods

---

- Creating new tables containing some of the original columns:
    - `select`, `drop`
  - Accessing data in a column
    - `column` takes a label or index and returns an array
  - Finding the size: `num_rows` and `num_columns`
  - Referring to columns: labels, relabeling, and indices
    - `labels` and `reabeled`; column indices start at 0
  - Using array methods to work with data in columns
    - `item`, `sum`, `min`, `max`, and so on
-

# Take Rows, Select Columns

---

The `select` method returns a table with only some columns

The `take` method returns a table with only some rows

- Rows are numbered, starting at 0
  - Taking a single number returns a one-row table
  - Taking a list of numbers returns a table as well
-

# The where method

---

- `t.where(label, condition)` - constructs a new table with just the rows that match the condition

# Manipulating Rows

---

- `t.sort(column)` sorts the rows in increasing order
  - `t.take(row_numbers)` keeps the numbered rows
    - Each `row` has an index, starting at 0
  - `t.where(column, are.condition)` keeps all rows for which a column's value satisfies a condition
  - `t.where(column, value)` keeps all rows containing a certain value in a column
-