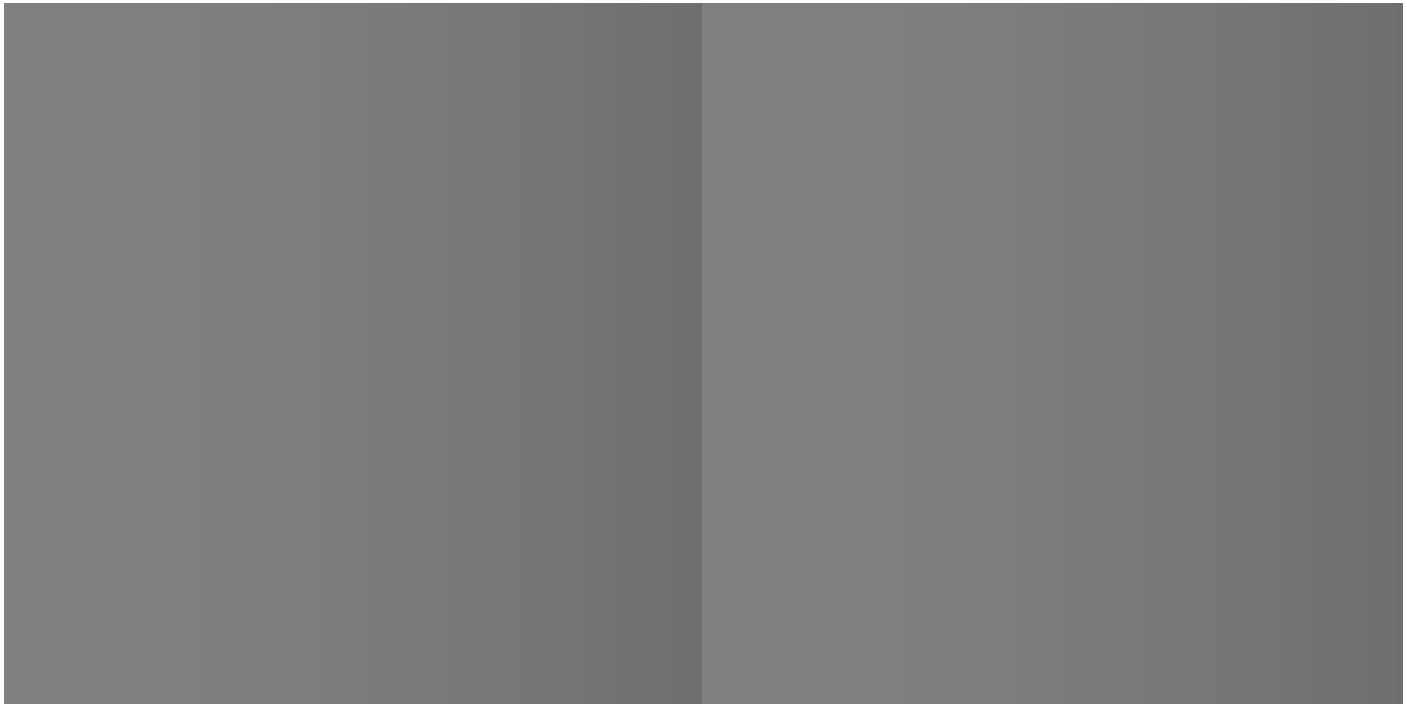
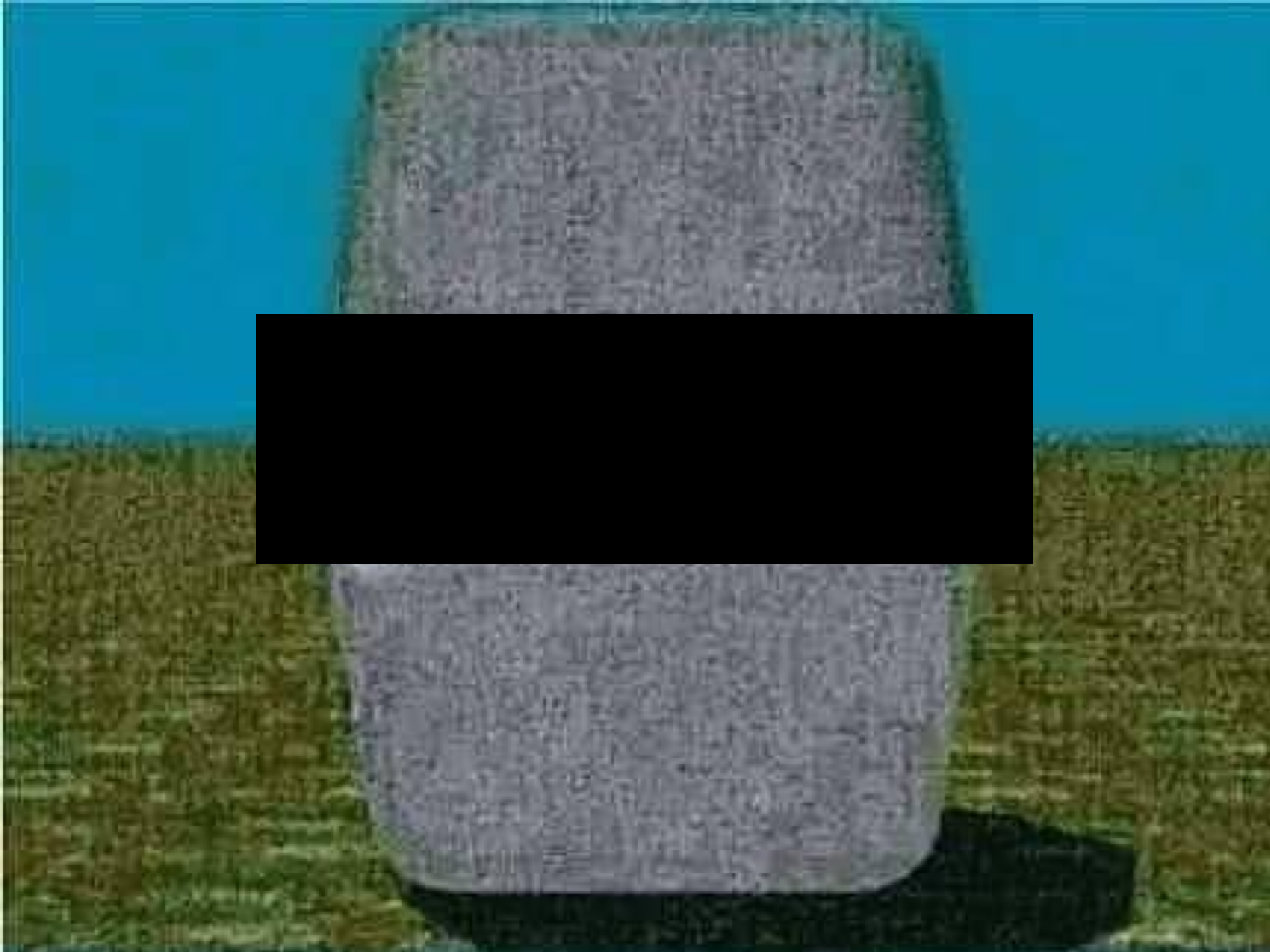
# Information Visualization: Data and Tasks

Class 5

# Describe What You See…



**The Gradient Illusion**



# Class Outline

- Questions?
  - Assignment 2 due **today (now past due!)**
  - Team formation
    - Submit problem preferences by tomorrow (Wed @ 11am)
    - Form teams In class on Thursday
- Today
  - Finish "Hands On" from last class
  - Information Visualization Part 2: Data and Tasks
- Coming Soon…
  - In one week, next “Hands On” will cover D3

# From Last Class…

# Introduction to SVG

- Scalable Vector Graphics
  - Web standard for vector graphics
  - Most common way to use D3 for visualization

- Lines
- Circles
- Squares
- Polygons
- Transforms

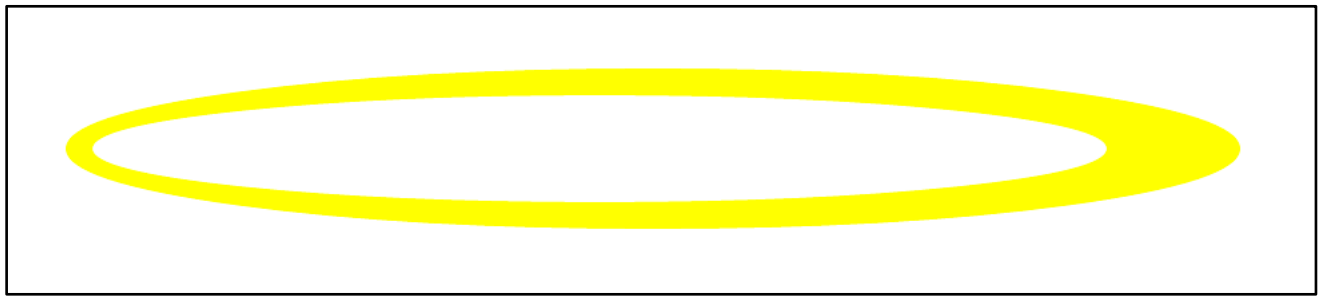
# Simple SVG Scene

- Looks a lot like other HTML elements:

```
<html>
<body>

<svg height="100" width="500">
  <ellipse cx="240" cy="50" rx="220" ry="30" style="fill:yellow" />
  <ellipse cx="220" cy="50" rx="190" ry="20" style="fill:white" />
  Sorry, your browser does not support inline SVG.
</svg>

</body>
</html>
```



- Let's take a look…
  - https://www.w3schools.com/graphics/svg_examples.asp

# SVG Elements and Shapes

- <svg> … </svg>
  - The element on which the shapes will be drawn.

- **Shapes**
  - <rect>
  - <circle>
  - <ellipse>
  - <line>
  - <polygon>
  - <polyline>
  - <path>
  - <text>

- **More advanced:**
  - Groups (<g>) are containers.  A bit like “divs” for regular HTML.
  - Transforms can shift/rotate SVG elements (e.g., turn a <rect> into a diamond)
    - Not an element itself, but an ***attribute*** of an element
    - <g transform=“rotate(45)”>
    - For more info: http://www.w3.org/TR/SVG/coords.html#TransformAttribute

# SVG and the DOM

- **SVG adds new element types to the DOM**
  - Still elements with attributes
  - CSS can be used to “style” svg elements
    - fill
    - stroke
    - stroke-width
- **Drawing is done by composing an SVG “scene”**
  - Represented as a DOM subtree on a web page
  - Shapes are drawn in the order they appear in DOM

# Hands On: SVG

- Scene Definition
- Interaction

# D3 and the DOM

- D3 is largely a DOM manipulation language
  - Add nodes
  - Remove nodes
  - Update nodes

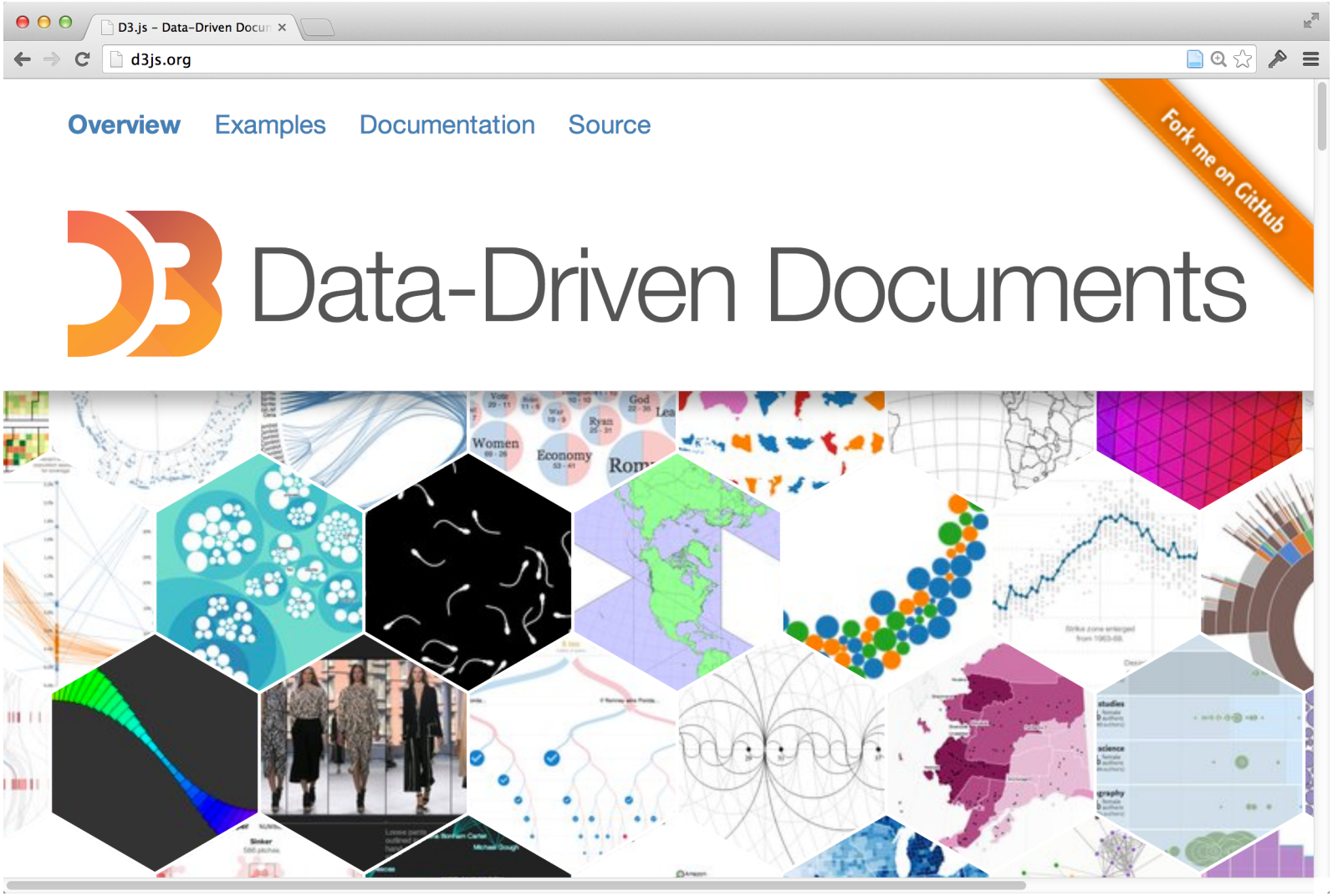
- D3 == **D**ata **D**riven **D**ocuments

# D3 and the DOM

- Manipulation is driven by data
  - **Mappings** (defined as JavaScript functions) define correspondence between **Data** and **DOM Elements**

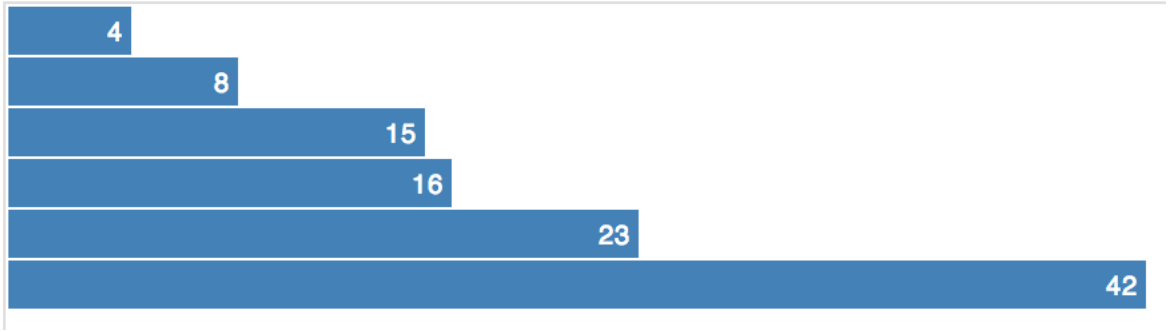
- When the DOM elements are SVG elements
  - We have a graphical **visualization**
  - Need not be SVG!
    - Canvas, “plain HTML” including color-coded DIVs.

# D3js.org


D3.js – Data-Driven Docu
d3js.org
Overview
Examples
Documentation
Source
Fork me on GitHub
Data-Driven Documents

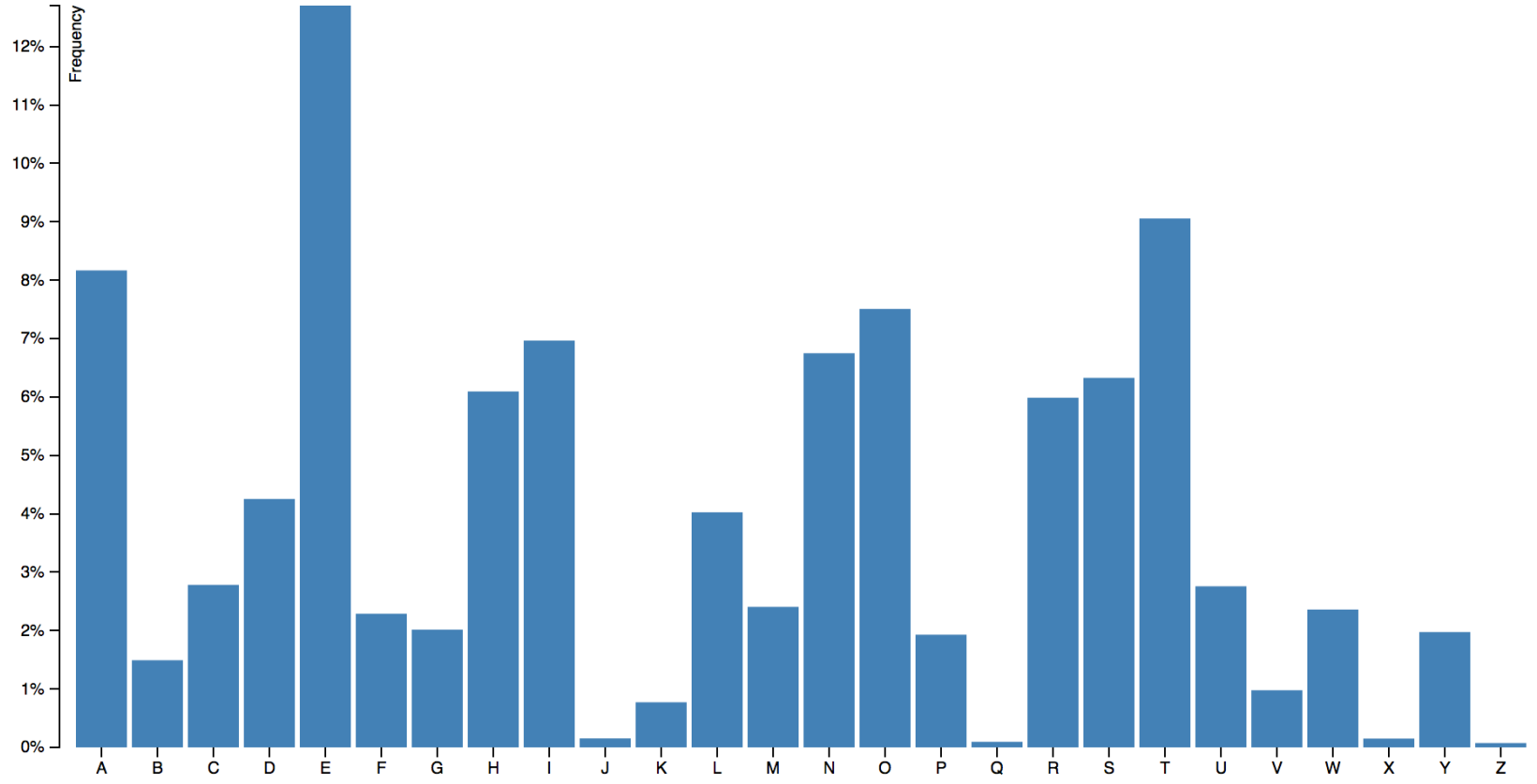
# An HTML-based Example

- https://bl.ocks.org/davegotz/8c0baa7a34a137af33fd28498e269a75



# An SVG-based Example

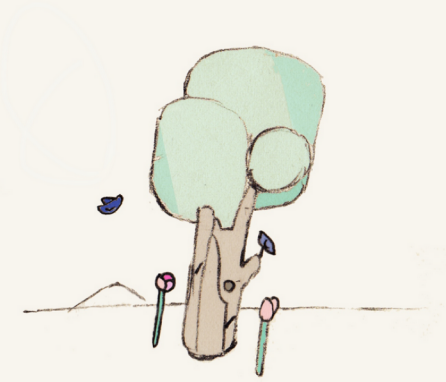
- https://bl.ocks.org/davegotz/93eae24cb3b0cf8f209334d0626bd62f



# Simple Statistics

## http://simplestatistics.org/

SIMPLE STATISTICS



**Statistical methods in readable JavaScript for browsers, servers, and people.**

GITHUB NPM DOCS

Star 862 Tweet 7

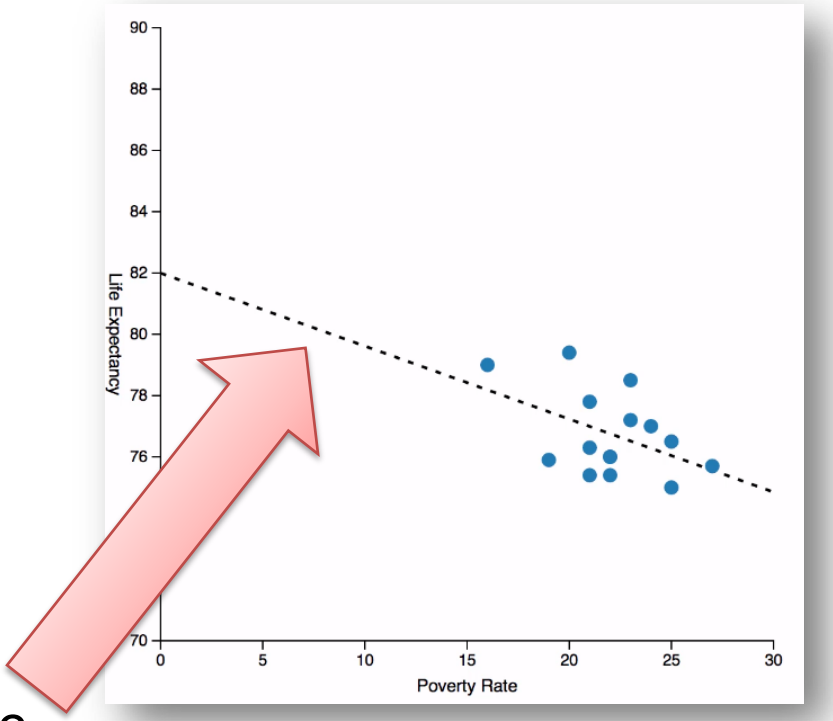
FEATURES

# Simple Statistics

- Does NOT manipulate the DOM
- Basic “library” design
  - Functions
  - You call them with parameters
  - They return results
- Examples
  - Correlation
  - Regression
  - Classification

# Simple Statistics

- **We'll use Simple Statistics to generate new data**
  - Used to guide data transformation
    - e.g., filter to a given class
  - Used to feed D3-based visualization
    - e.g., draw a regression line



# Hands On: D3 and Simple Statistics

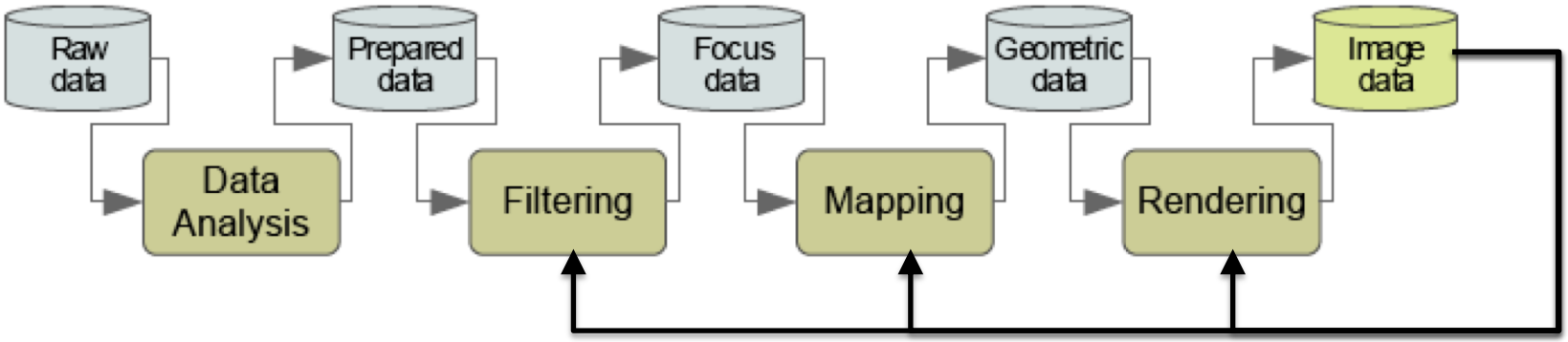
- Include in your web page
- Make a function call

- We'll do more in future classes…

# Data and Tasks

# From Data to Graphic

- The Visualization Pipeline



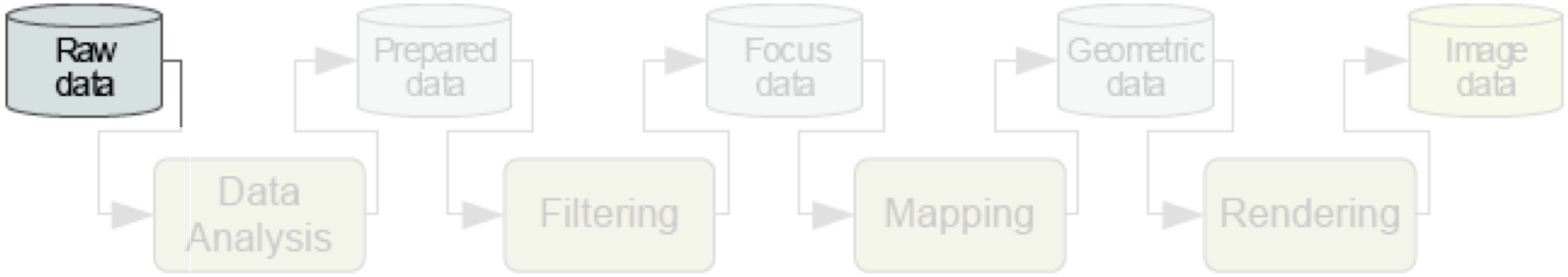

- **Creating a visualization includes designing for <u>analysis</u>, <u>filtering</u>, <u>mapping</u>, and <u>interaction</u>**
  - Rendering is a “solved problem” with several libraries available for use (e.g., SVG)

Image from http://www.infovis-wiki.net/index.php/Visualization_Pipeline, with modifications.

# Where to begin?

- How do we begin designing a visualization?
- What must we know?

**1. The raw data**

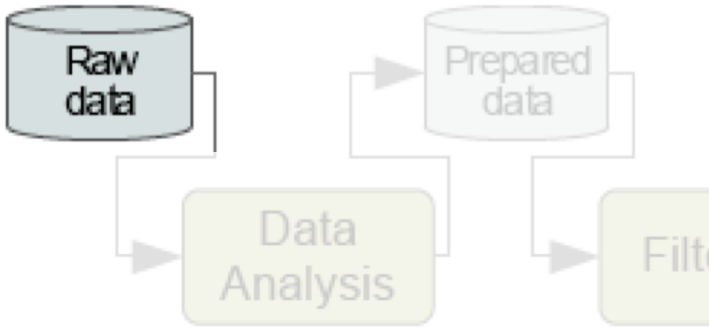


# Data

- Data is the input to a visualization
- For example:
  - Tableau, Excel and other visualization packages all allow users to load data to visualize


# Is Raw Data Enough?

| | | | |
|---|---|---|---|
| Alabama | 75.4 | 22 | 1 |
| Alaska | 78.3 | 21 | 6 |
| Arizona | 79.6 | 23 | 3 |
| Arkansas | 76 | 22 | 1 |
| California | 80.8 | 24 | 5 |
| Colorado | 80 | 16 | 3 |
| Connecticut | 80.8 | 14 | 2 |
| Delaware | 78.4 | 17 | 2 |
| Dist of Columbia | 76.5 | 25 | 2 |
| Florida | 79.4 | 20 | 1 |
| Georgia | 77.2 | 23 | 1 |
| Iowa | 79.7 | 14 | 3 |
| Kansas | 78.7 | 17 | 3 |
| Kentucky | 76 | 22 | 1 |
| … | | | |



# Understanding Data

- Just “having data” is not enough

- You must understand
  - How it is defined
  - How it is organized
  - What it means

- More formally, you must understand a dataset’s:
  - **Types**
  - **Structures**
  - **Semantics**

# Data Types

- **"Unfortunately, classification of data is a big issue. It is closely related to the classification of knowledge, and it is with great trepidation that we approach the subject."**

  - Colin Ware, Information Visualization

- Ware and Munzner: today's readings
  - **Both proposed classification systems**
    - Different, but similar
  - **We don't need a single universally accepted system**

- Our motivation is to formalizing our way of thinking
  - Allows us to create visualizations for classes of data problems
  - "Templates" for data that meets certain requirements

# Data Types

- **Four basic categories for individual units of data**
  - **Categorical**
  - **Ordinal**
  - **Interval**
  - **Ratio**

# Categorical

- **Categorical** data is defined using an unordered set of labels
  - Sometimes called "**nominal**" data

- **For example:**
  - **A list of car brands:**
    Honda, Toyota, Ford, GM, Nissan, Volkswagen, Tesla
  - **A list of baseball teams:**
    Yankees, Mets, Orioles, Dodgers, Tigers
  - **A list of department/school codes:**
    INLS, COMP, BIOS

# Categorical Data

- **Allows comparison**
  - Is INLS = COMP?
- **Ordering is undefined**
  - Unless we have more information… how are we comparing?
  - Is INLS > COMP?

# Ordinal

- Ordinal data has a defined order
  - It makes sense to ask of A>B, A<B, or A=B

- For example:
  - Saffir-Simpson Scale for Hurricanes
    - Category One to Category Five
  - US Government's Homeland Security Advisory System (the "Threat Level")

# Ordinal Data

- Some ordinal data is **cyclical**
  - For example, “Morning, Afternoon, Evening, Night”
    - Afternoon is before Evening
    - However the sequence repeats (morning comes after night)

# Interval Data

- Ordered data that has a defined scale, allowing measurement between values

- For example, **Response Scales**
  - "On a scale of 1 to 10, rate your level of pain?"
    - A pain level increasing by 3 is more than a pain level increasing by 1.
  - Amazon Star ratings 


- Contrast these to US Threat Level ratings
  - Is difference between "Orange" and "Yellow" the same as between "Yellow" and "Blue"?

# Ratio Data

- Real numbers on a number line that includes a defined “zero”

- For example:
  - **Money: $40 is 2x $20**
  - **Age: 60 is 3x 20**

- This is our core quantitative representation, what we think of as a “number”

# Why Do Data Types Matter **for Visualization**?

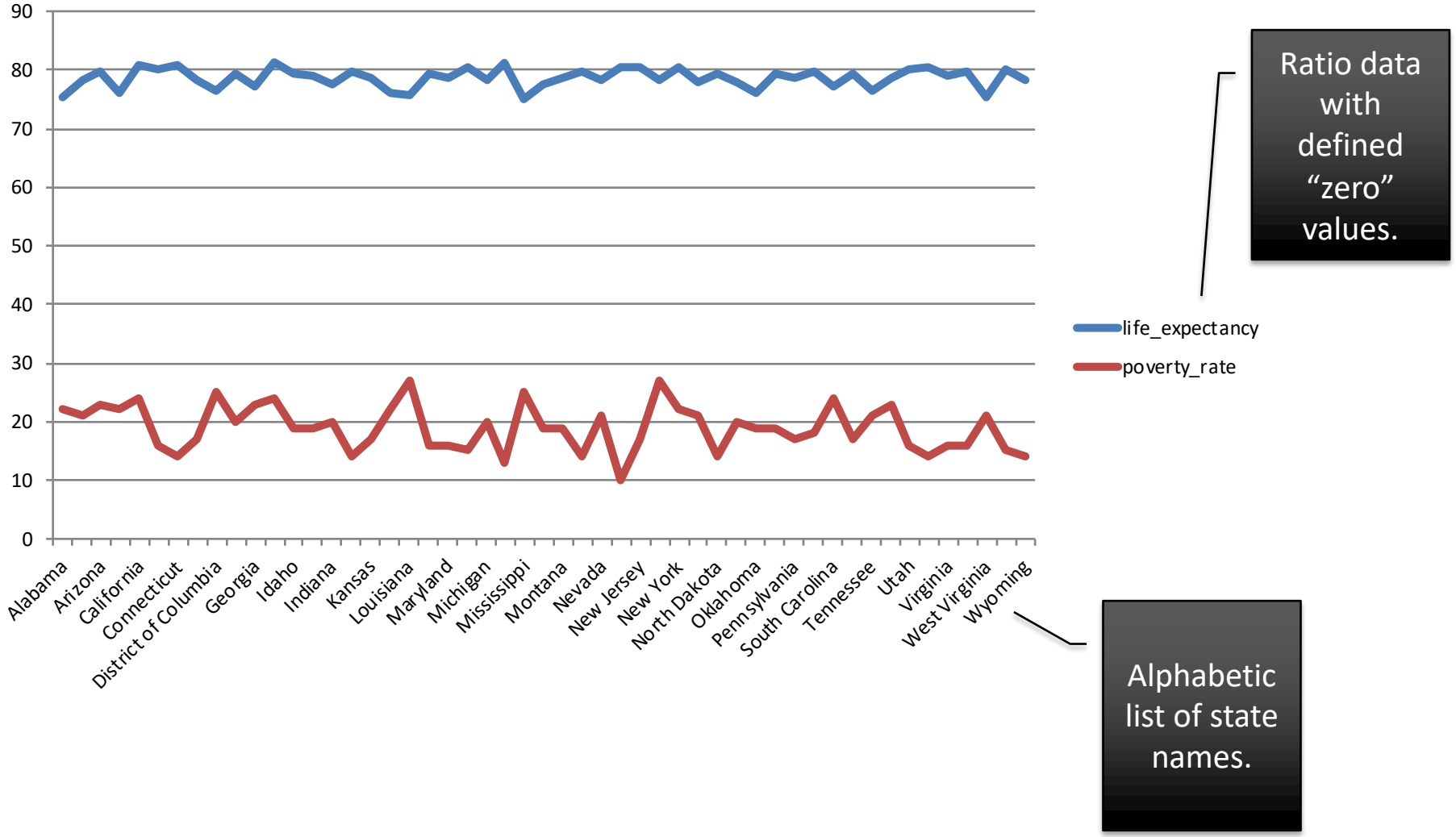
Ratio data with defined "zero" values.

Ratio data with defined "zero" values.

Treat state name as ordinal: Alphabetical order.

| | | | |
|---|---|---|---|
| Alabama | 75.4 | 22 | 1 |
| Alaska | 78.3 | 21 | 6 |
| Arizona | 79.6 | 23 | 3 |
| Arkansas | 76 | 22 | 1 |
| California | 80.8 | 24 | 5 |
| Colorado | 80 | 16 | 3 |
| Connecticut | 80.8 | 14 | 2 |
| Delaware | 78.4 | 17 | 2 |
| Dist of Columbia | 76.5 | 25 | 2 |
| Florida | 79.4 | 20 | 1 |
| Georgia | 77.2 | 23 | 1 |
| Iowa | 79.7 | 14 | 3 |
| Kansas | 78.7 | 17 | 3 |
| Kentucky | 76 | 22 | 1 |
| … | | | |

# Why Do Data Types Matter **for Visualization**?



# Data Structures

- **Data Types** provide a classification of atomic units of data

- **Data Structures** describe how that data organized with respect to other data

# Entities and Attributes

- Data is typically organized as **entities**
  - aka items, objects, records
  - e.g., a US State
- Entities have **attributes**
  - Attributes have types
  - e.g. a US State might have:
    - **Name** (categorical)
    - **Poverty rate** (ratio)
    - **Life expectancy** (ratio)
    - **Region** (categorical)
    - **State bird** (categorical)
    - **"Political rating"** from Conservative to Liberal (ordinal)

# Relationships

- Entities can have **relationships**
  - North Carolina **borders** South Carolina
  - North Carolina **borders** Virginia

- Many types of relationships
  - Peer relationships (like **borders**, between same types of things)
  - Part-of relationships (SILS is **part-of** UNC)
  - **Causal** relationships; **temporal** relationships; **spatial** relationships; and many, many more

# Relationships

- Like entities, **relationships** can have **attributes**
  - Distance between a pair of cities
  - Number of interactions between contacts in a social network

- Relationships can be **directed (or not)**
  - SILS **is part of** UNC; UNC **is not part of** SILS
  - UNC **plays sports against** Duke, and Duke **plays sports against** UNC

# Dimensionality

- Attributes can have one or more **dimensions**
  - A person's weight and age are 1-dimensional attributes
  - A person's location on a map is an attribute with two dimensions: latitude and longitude

# Data Structure Representations

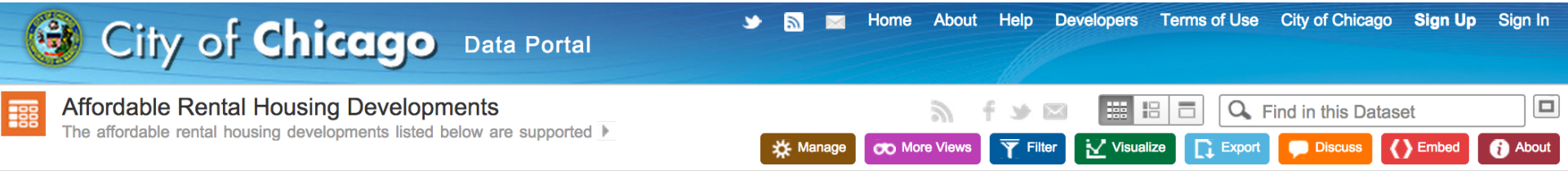
- **Tables** are perhaps the most common representation
  - Relational databases
  - Spreadsheets
  - CSV files

- Within a table:
  - Each **row** is an **entity**
  - Each **column** is an **attribute**
    - Multi-dimensional attributes often “flattened” into multiple columns

- Relationships (and their attributes) also flattened into columns or other tables

# Data Structure Representations

- For example, a data table about Chicago:




Affordable Rental Housing Developments

The affordable rental housing developments listed below are supported

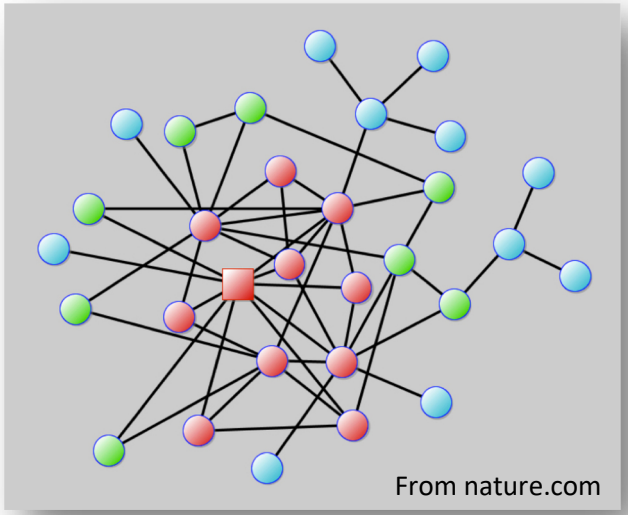
Find in this Dataset

Manage | More Views | Filter | Visualize | Export | Discuss | Embed | About

| | Community Area Name | Community Area Number | Property Type | Property Name | Address | Zip Code |
|---|---|---|---|---|---|---|
| 1 | Albany Park | 14 | Senior | Mayfair Commons | 4444 W. Lawrence Ave. | 60630 |
| 2 | Albany Park | 14 | Senior | Senior Suites of Ravenswood Manor | 2800 W. Montrose Ave. | 60618 |
| 3 | Ashburn | 70 | Senior | Wrightwood Senior Apartments | 2815 W. 79th St. | 60652 |
| 4 | Auburn Gresham | 73 | Senior | Brainerd Senior Center | 8915 S. Loomis St. | 60620 |
| 5 | Auburn Gresham | 71 | Multifamily | Stone Terrace | 8440 S. Parnell Ave. | 60620 |
| 6 | Auburn Gresham | 71 | Senior HUD 202 | St. Sabina Elder Village Apartments | 1222 W. 79th St. | 60620 |
| 7 | Auburn Gresham | 71 | Senior | Senior Suites of Auburn-Gresham | 1050 W. 79th St. | 60620 |
| 8 | Auburn Gresham | 71 | Senior HUD 202 | Naomi & Sylvester Smith Senior Living Ctr. | 8019-55 S. Halsted St. | 60620 |
| 9 | Auburn Gresham | 71 | Senior | Auburn Commons | 1626 W. 87th St. | 60620 |
| 10 | Austin | 25 | Multifamily | Pine Central | 557 N. Pine Ave. | 60644 |
| 11 | Austin | 25 | Senior | Senior Suites of Austin | 335 N. Menard Ave. | 60644 |
| 12 | Austin | 22 | Multifamily | Madison Renaissance Apts. | 1816 N. St. Louis Ave. | 60647 |
| 13 | Austin | 25 | Supportive Housing | Menard Apartments | 334 N. Menard Ave. | 60644 |
| 14 | Austin | 25 | Multifamily | Rebecca Walker Complex | 126 S. Central Ave. | 60644 |
| 15 | Austin | 25 | Multifamily | Rebecca Walker Complex | 223-9 S. Central Ave. | 60644 |
| 16 | Austin | 25 | Multifamily | Rebecca Walker Complex | 5565-71 W. Quincy Ave. | 60644 |

# Other Representations: Network

- A Network (or graph)
  - **Nodes** represent entities
  - **Edges** represent relationships

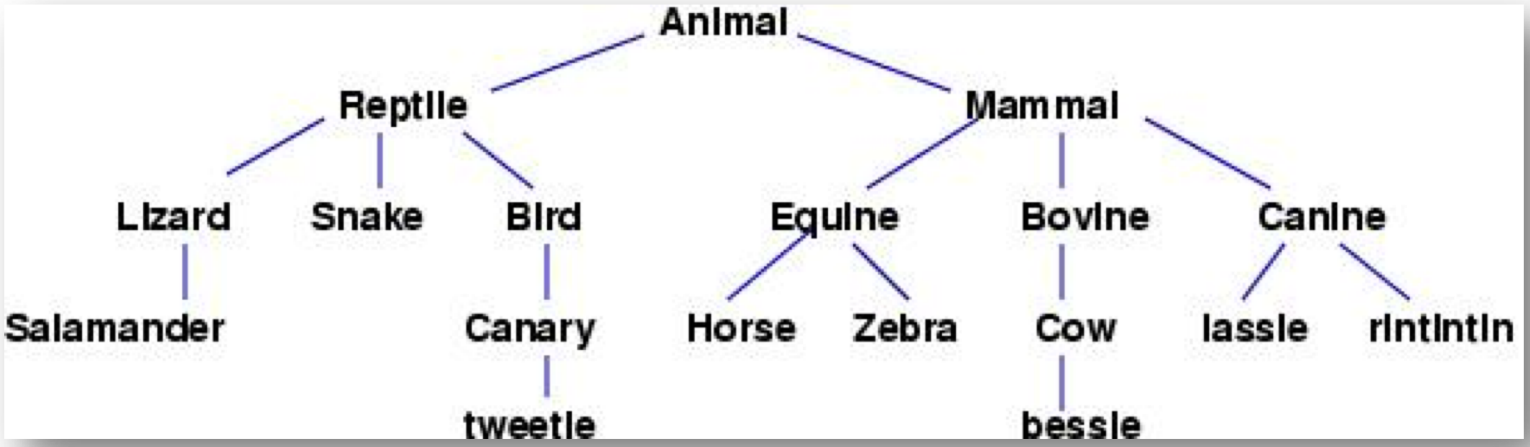



- Can generally be mapped to tabular representation
  - Table(s) of nodes
  - Table(s) of relationships
  - Foreign keys to link them together

- Some calculations/operations are more efficient when data is represented as a network
  - Avoids table “joins” which can be expensive
  - e.g., algorithms that “traverse” the graph

# Other Representations: Trees

- Trees capture hierarchical relationships
  - A single “root” node
    - If not, you have a “forest” of disjoint trees
  - No cycles
- **Trees are just a special form of network**



# Other Representations: Fields

- **Fields are functions with a continuous domain**

- Fields vs. tables of discrete data points:
  - Fields can be **sampled** to generate a table
  - More precision needed? You can re-sample.

- Consider Newton's Law of Motion: **F = ma**
  - Models the relationship between **F** (force) and **a** (acceleration) for an object with mass **m**.
  - **To get data for mass=10kg, we can sample the field for Force values at several different acceleration levels.**
  - **Need more resolution? Re-sample at smaller intervals.**

# Statistical Models and Fields

- Fields are often generated during data analysis
- Consider a linear regression model:

$$Y = mX + b$$

  - Regression solves for **m** and **b** given a set of (x,y) samples from a table

  - You can visualize the regression line by sampling the field

# Semantics

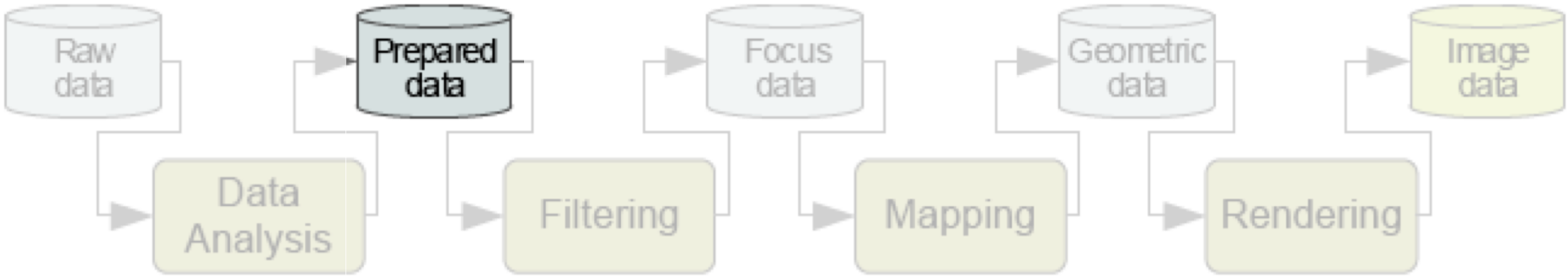
- Data **type** and **structure** alone are insufficient for proper interpretation.
  - We need semantics. What does the data represent?

- What can we learn from the following data without knowing the semantics of each column?

| *Categorical* | *Ratio* | *Ratio* | *Categorical* |
|---|---|---|---|
| Alabama | 75.4 | 22 | 1 |
| Alaska | 78.3 | 21 | 6 |
| Arizona | 79.6 | 23 | 3 |
| Arkansas | 76 | 22 | 1 |
| California | 80.8 | 24 | 5 |

# Practicalities: Tables and D3

- In practice, data analysis routines that creates **Prepared Data** in **Table** form are convenient

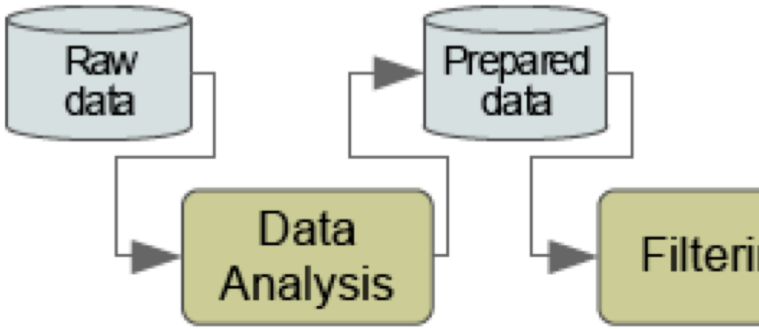



- In JavaScript, we can use **Arrays of Objects** to represent a table
  - **Filtering** can use methods like Array.filter()
  - **Mapping** via D3 is analogous to JavaScript's Array.map()

# Is Typed, Structured, Semantically Meaningful Data Enough to Design a Vis?

| STATE | LifeExp | PovRate | Region |
|---|---|---|---|
| *Categorical* | *Ratio* | *Ratio* | *Categorical* |
| Alabama | 75.4 | 22 | 1 |
| Alaska | 78.3 | 21 | 6 |
| Arizona | 79.6 | 23 | 3 |
| Arkansas | 76 | 22 | 1 |
| California | 80.8 | 24 | 5 |
| Colorado | 80 | 16 | 3 |
| Connecticut | 80.8 | 14 | 2 |
| Delaware | 78.4 | 17 | 2 |
| Dist of Columbia | 76.5 | 25 | 2 |
| Florida | 79.4 | 20 | 1 |
| Georgia | 77.2 | 23 | 1 |
| Iowa | 79.7 | 14 | 3 |
| Kansas | 78.7 | 17 | 3 |
| … | | | |

| Region | Name |
|---|---|
| *Categorical* | *Categorical* |
| 1 | South |
| 2 | Northeast |
| … | |



# Both Data and Task

- Good designs must reflect both **data** and **task**
  - **What** you are visualizing?
  - **Why** you are visualizing it?

# Theory vs. Practice

- **Theory**
  - Task characterization
  - Enables classification of visualization methods by identifying which tasks they support

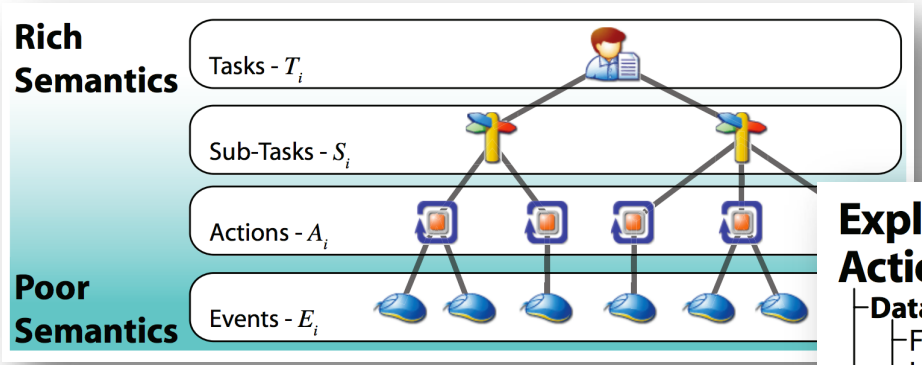
- **Practice**
  - For a given **dataset**, what **user tasks** should your visualization design support?
    - Only then can we answer determine **how best to design a visualization.**

# Task Classification Systems

- There are no universally accepted classification systems for of visualization tasks
- Similar to data type classification
  - But even more variation between systems!
- In this class, we'll focus primarily on practical suggestions

# An Example Classification System

A hierarchical classification representing high level analysis tasks as a composition of low-level user **actions**:



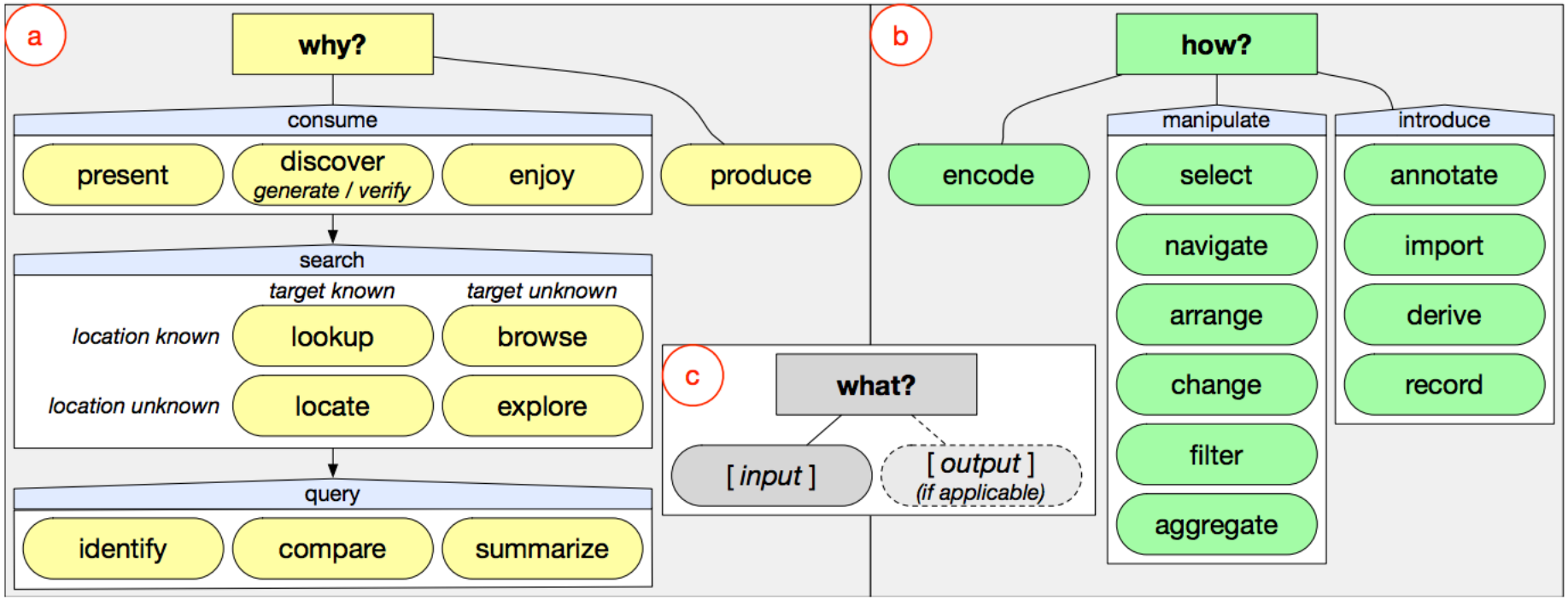

**Exploration Actions**
- **Data Exploration Actions**
  - Filter
  - Inspect
  - Query
  - Restore
- **Visual Exploration Actions**
  - Brush
  - Change-Metaphor
  - Change-Range
    - Zoom
    - Pan
  - Merge
  - Sort
  - Split

**Insight Actions**
- **Visual Insight Actions**
  - Annotate
  - Bookmark
- **Knowledge Insight Actions**
  - Create
  - Modify
  - Remove

**Meta Actions**
- Delete
- Edit
- Redo
- Revisit
- Undo

From Gotz and Zhou "Characterizing Users' Visual Analytic Activity for Insight Provenance" 2008.

# A “Top Down” Look at Task Classification




Brehmer and Munzner. **A Multi-Level Typology of Abstract Visualization Tasks.** 2013.

# Seven Abstract Tasks

from Ben Shneiderman's 1996 "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualization"

- **Overview:** Gain an overview of the entire collection
- **Zoom:** Zoom in on items of interest
- **Filter:** filter out uninteresting items
- **Details-on-demand:** Select an item or group and get details when ended.
- **Relate:** View relationships among items.
- **History:** Keep a history of actions to support undo, replay, and progressive refinement
- **Extract:** Allow extraction of sub-collections of the query parameters

# Many Many More. For example…

- Amar, Eagan, and Stasko. **Low-level components of analytic activity in information visualization.** 2005.
- Amar and Stasko. **A knowledge task-based framework for design and evaluation of information visualizations.** 2004.
- Chi and Riedl. **An operator interaction framework for visualization systems.** 1998.
- Chuah and Roth. **On the semantics of interactive visualizations.** 1996.
- Wehrend and Lewis. **A problem-oriented classification of visualization techniques.** 1990.
- Zhou and Feiner. **Visual task characterization for automated visual discourse synthesis.** 1998.

# Practical Considerations

- **In practice, the specific vocabulary you use is not so important.**
- **However, before you design an interface:**
  - **Identify your users?**
    - Why do they want to visualize the data?
    - What questions are they interested in answering?
    - What would be valuable for them to to confirm?
    - What might be valuable for them to discover?
  - **What tasks do you need to support to answer these questions?**
    - Consider the “task types” we’ve discussed
    - Others:
      - Outlier identification
      - Trend discovery
      - Subsetting and comparison
      - Sorting and ranking
      - “Story telling” or advocacy
  - **What data will those tasks require? Do you have it?**

# Consider an Example

- Suppose we plan to build a new visual analysis platform for the real estate industry. We have access to all home listings, plus historical sales data for the last 20 years.

- Without focusing on visual or interaction design choices, let's discuss the following.
  - **Who are the potential users and what questions might they want to answer?**
  - **What tasks should be supported?**
  - **What data is required? Do we have it?**

# Next Visualization Lecture…

- **Visual Encoding:** how properties of graphical marks can be used to represent the data and tasks discussed today.

## Additional Readings about Tasks…

- Chapter 3, "Why: Task Abstraction" in **Visual Analysis and Design,** by Munzner.
- Ben Shneiderman's 1996 paper "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualization"

Next Class…

# Project Team Formation