Visualización de Información y Analítica Visual

Daniela Flores (diflores@uc.cl)
Hernán Valdivieso (hvaldivieso@ing.puc.cl)

Resumen clase 2

What

What?

Datasets

Data Types

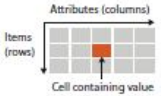
- Items
- Attributes
- Links
- Positions
- Grids

Data and Dataset Types

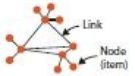
Tables	Networks & Trees	Fields	Geometry	Clusters, Sets, Lists
Items	Items (nodes)	Grids	Items	Items
Attributes	Links	Positions	Positions	
	Attributes	Attributes		

Dataset Types

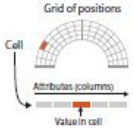
- Tables



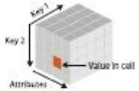
- Networks



- Fields (Continuous)



- *Multidimensional Table*



- *Trees*

- Geometry (Spatial)

Dataset Availability

- Static

- Dynamic

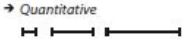
Attributes

Attribute Types

- Categorical

- Ordered
  - *Ordinal*

  - *Quantitative*



Ordering Direction

- Sequential

- Diverging

- Cyclic

Why

Why?

Actions

- **Analyze**
  - Consume
    - *Discover*
    - *Present*
    - *Enjoy*
  - Produce
    - *Annotate*
    - *Record*
    - *Derive*
- **Search**

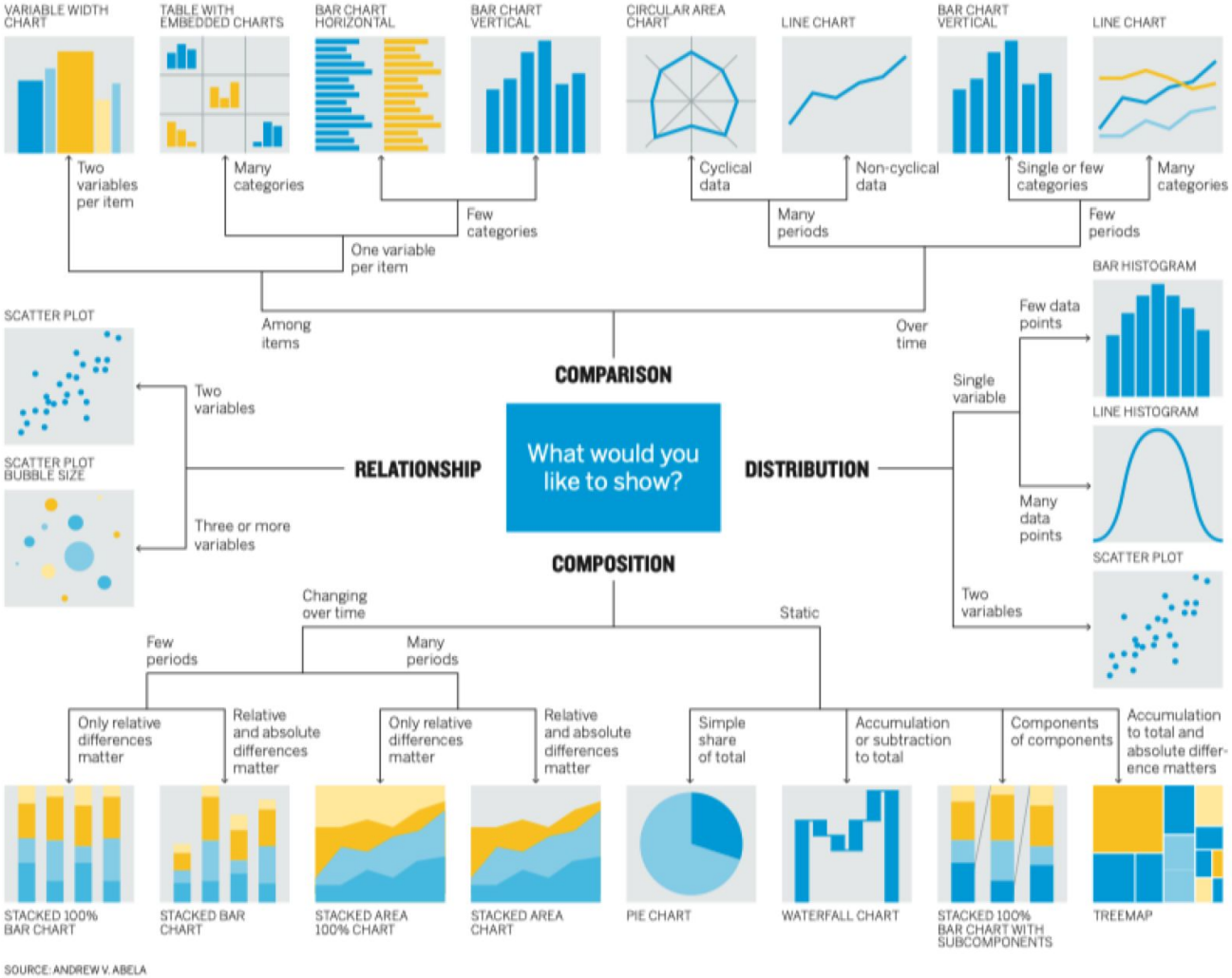
	Target known	Target unknown
Location known	*Lookup*	*Browse*
Location unknown	*Locate*	*Explore*

- **Query**
  - Identify
  - Compare
  - Summarize

Targets

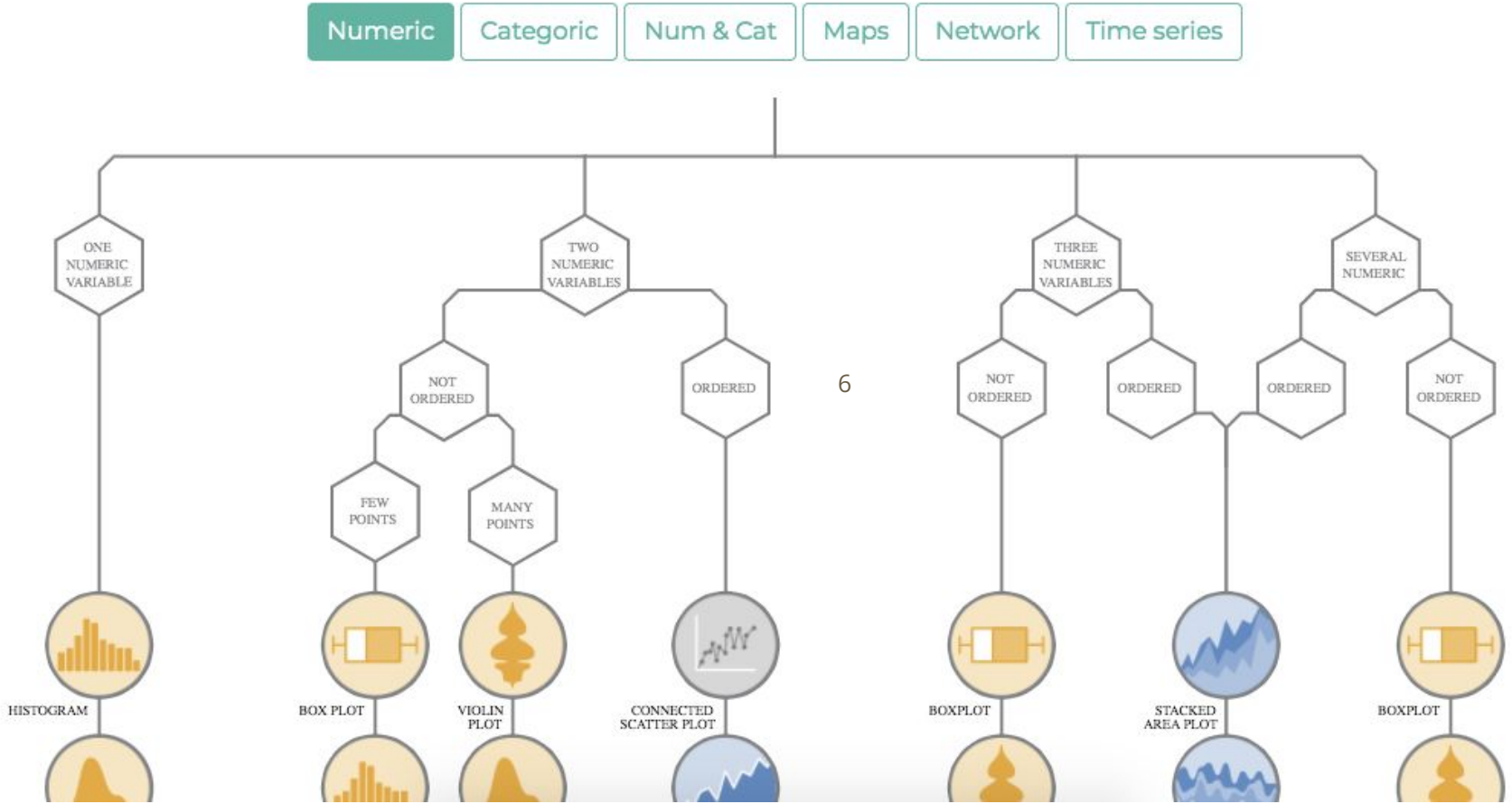
- **All Data**
  - Trends
  - Outliers
  - Features
- **Attributes**
  - One
    - *Distribution*
    - *Extremes*
  - Many
    - *Dependency*
    - *Correlation*
    - *Similarity*
- **Network Data**
  - Topology
  - *Paths*
- **Spatial Data**
  - Shape

What?
Why?
How?



SOURCE: ANDREW V. ABELA

https://www.data-to-viz.com/#explore



Clase 3: *Framework* anidado de Tamara Munzner

Contenidos

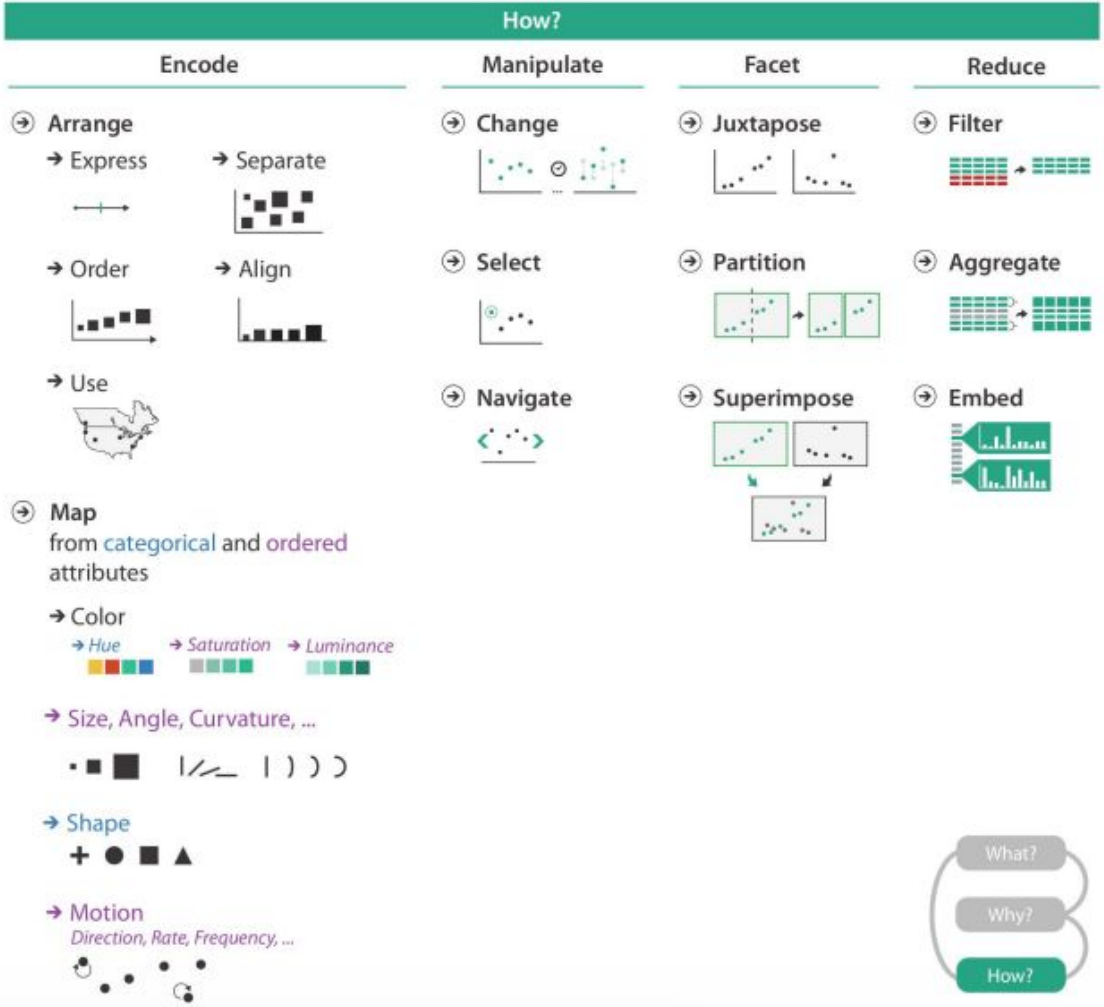
- *How*
- Altair más avanzado. Aplicando *How* en Python.

How

How

Última etapa del *framework.*

Qué **elecciones de diseño** disponemos para construir nuestra visualización.

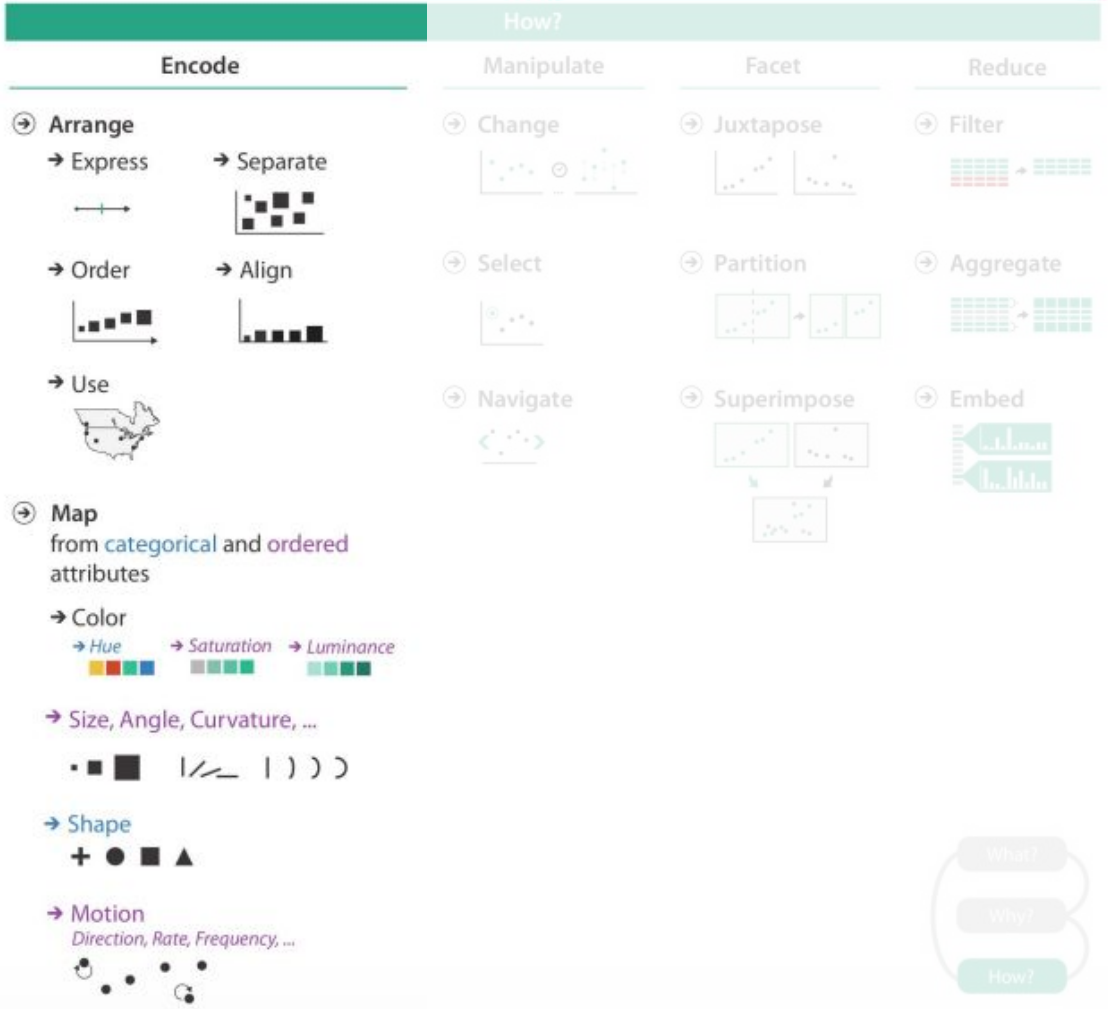


How

Las decisiones se pueden separar en dos áreas.

Visual encoding

Establece una traducción entre datos y elementos visuales

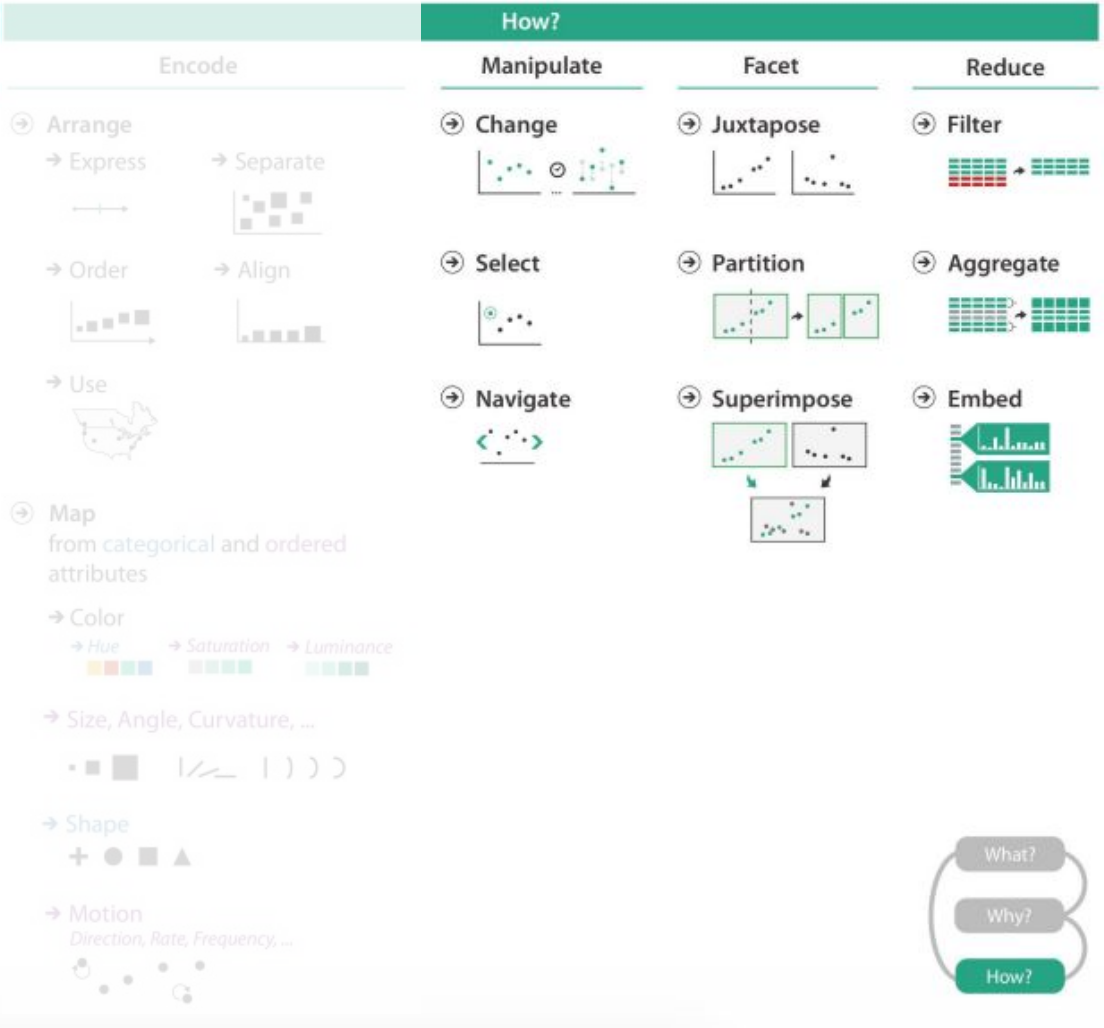


How

Las decisiones se pueden separar en dos áreas.

Interactions

Establece cómo el usuario va a interactuar con los elementos visuales



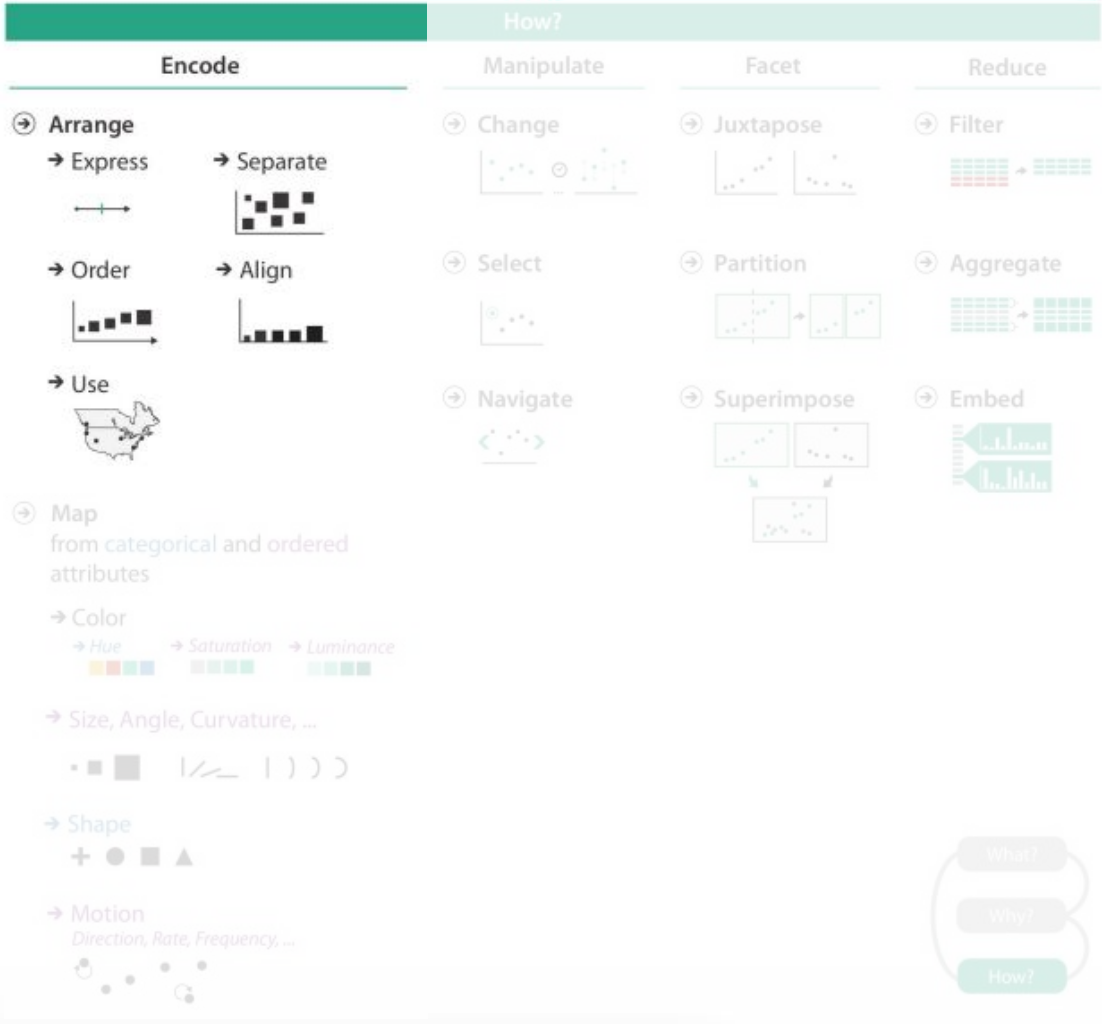
Visual Encoding

- Arrange
- Map

Arrange

Con *arrange*, buscamos saber cómo **organizar los datos** en el espacio.

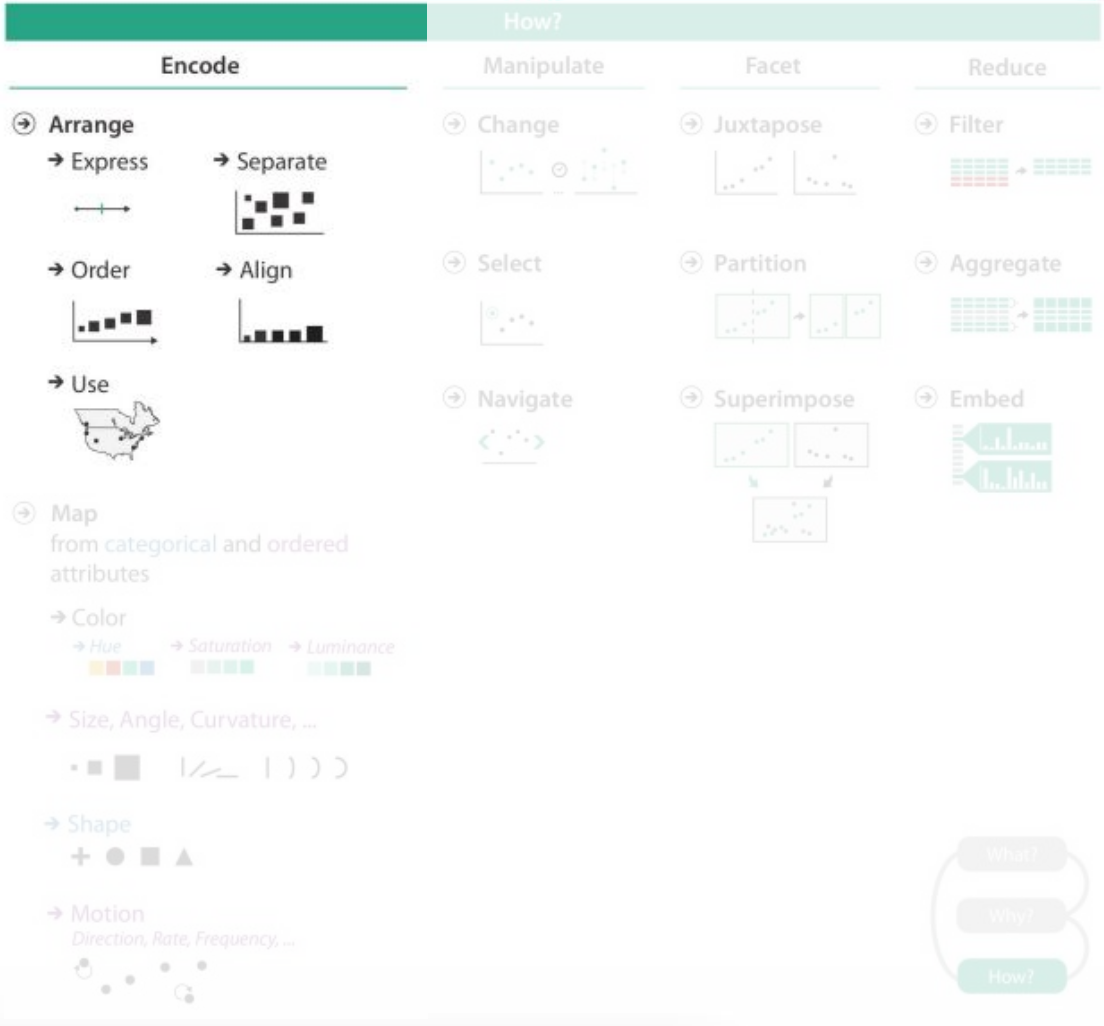
- De todos los *encodings*, es el más crucial porque el **uso del espacio** domina el modelo mental que tiene el usuario de los datos.



Arrange

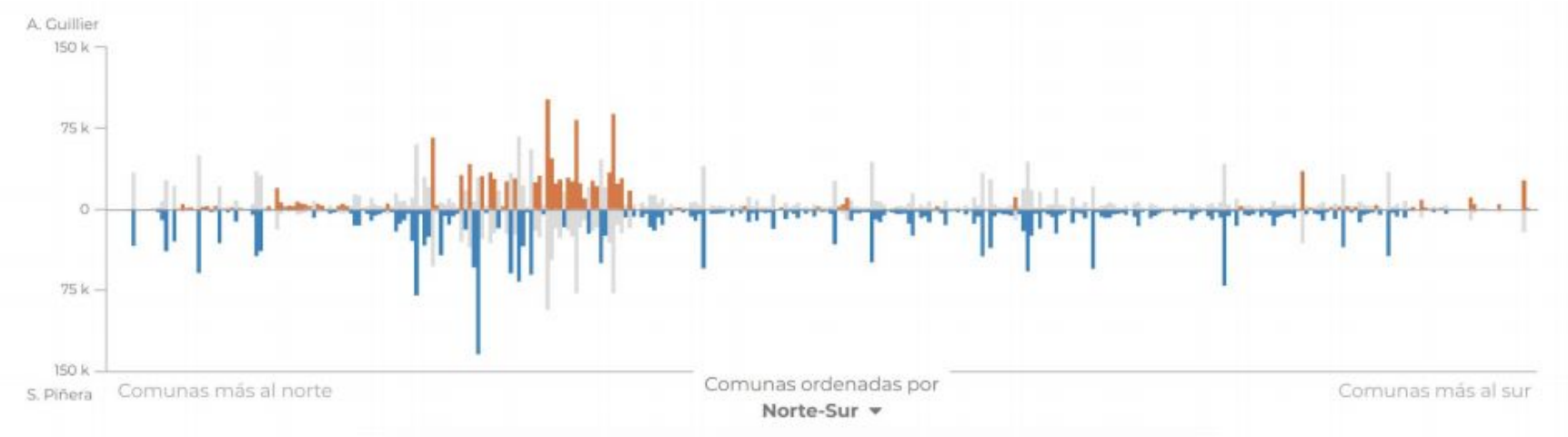
Con *arrange*, buscamos saber cómo **organizar los datos** en el espacio.

- De todos los *encodings*, es el más crucial, porque el **uso del espacio** domina el modelo mental que tiene el usuario de los datos.
- Queremos saber cómo expresar los valores, cómo separar, ordenar y alinear las regiones, y cómo usar un espacio dado (e.g. dataset geográfico)



Arrange

- Se **separan** los votos de los candidatos orientando las barras hacia arriba y hacia abajo y por comuna.
- Se **alinean** las barras desde el centro de la visualización.
- Se **ordenan** las comunas de acuerdo a la latitud.



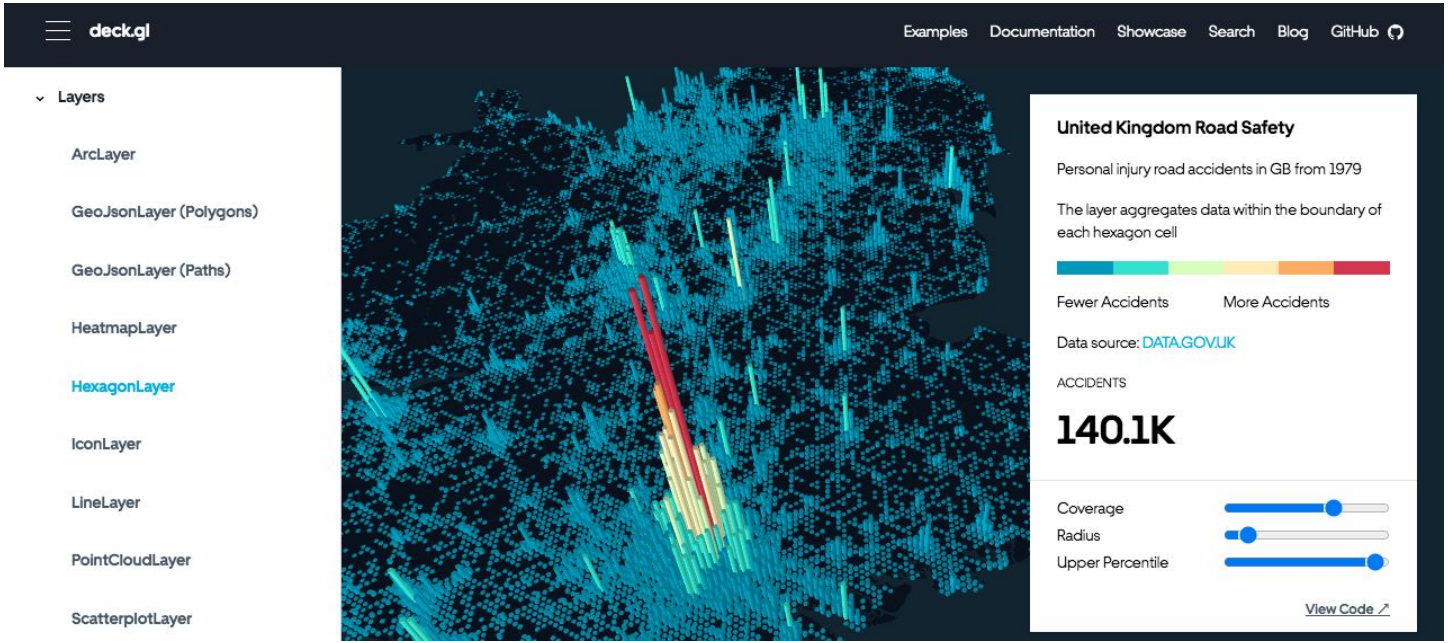
Fuente: DecideChile - Las Elecciones Voto a Voto

Arrange de datos espaciales: *Use*

- En datasets geográficos, *usamos* la información espacial que viene en el dataset para construir la visualización.
- Siguiendo el **principio de efectividad**, usamos el **canal de posición espacial** para representar las relaciones espaciales entre los ítems de nuestro dataset.

Arrange: datos espaciales

Usamos la latitud y longitud para **ubicar** la barra en el mapa de Reino Unido



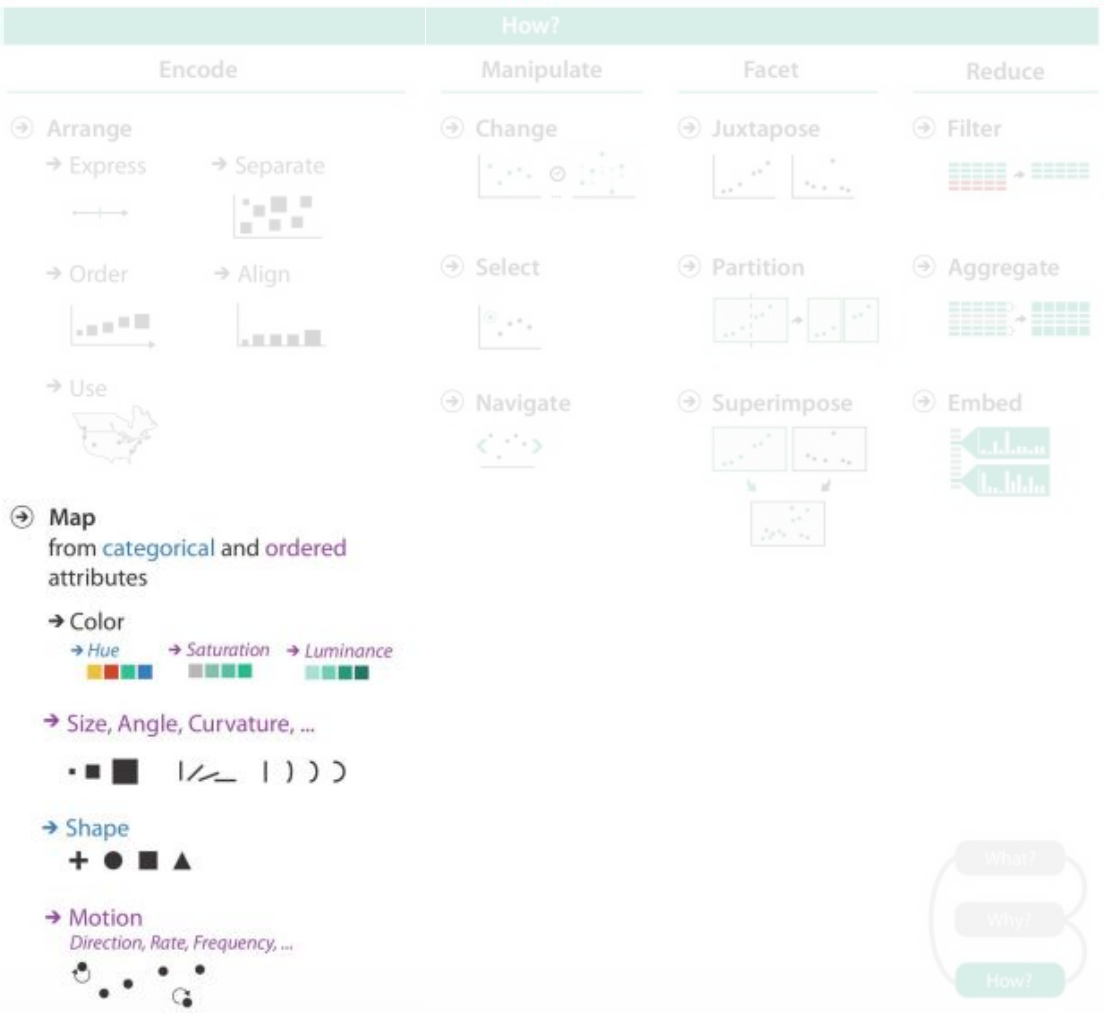
Fuente: https://deck.gl/examples/hexagon-layer/

Map

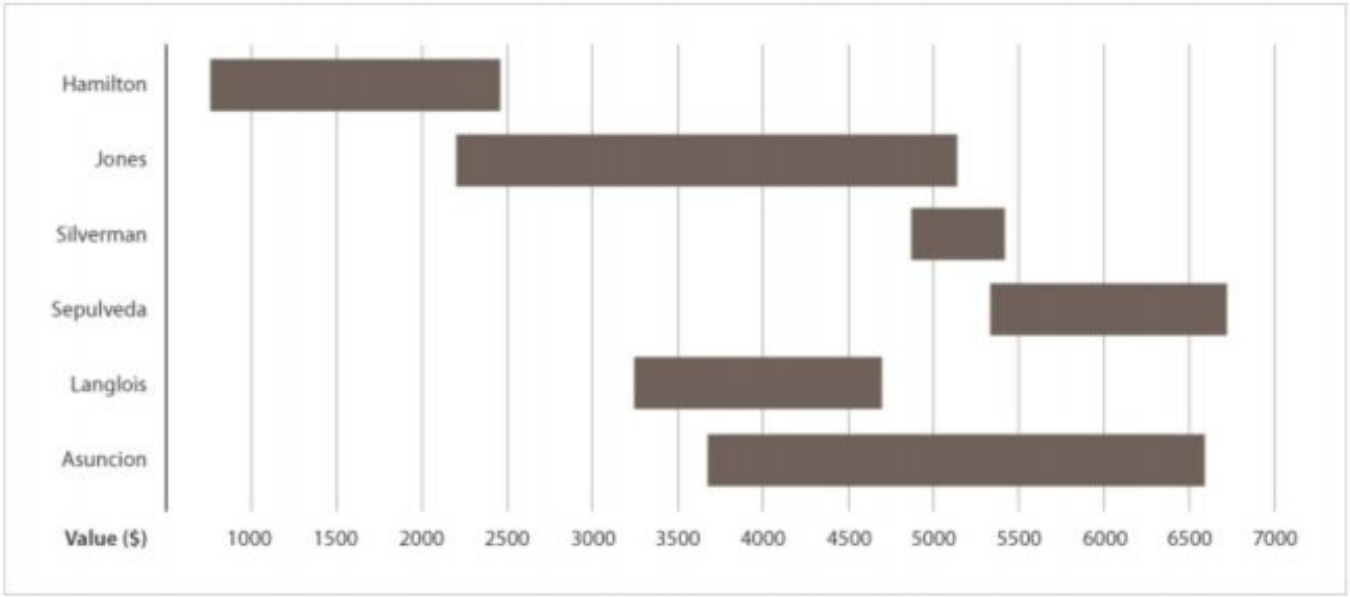
Con map, buscamos aprovechar los **canales visuales no-espaciales**.

- Podemos trabajar con color (hue, saturation, luminance), tamaño, ángulo, curvatura, formas.

- Pero también con atributos dinámicos: dirección, frecuencia, tasa de aparición.

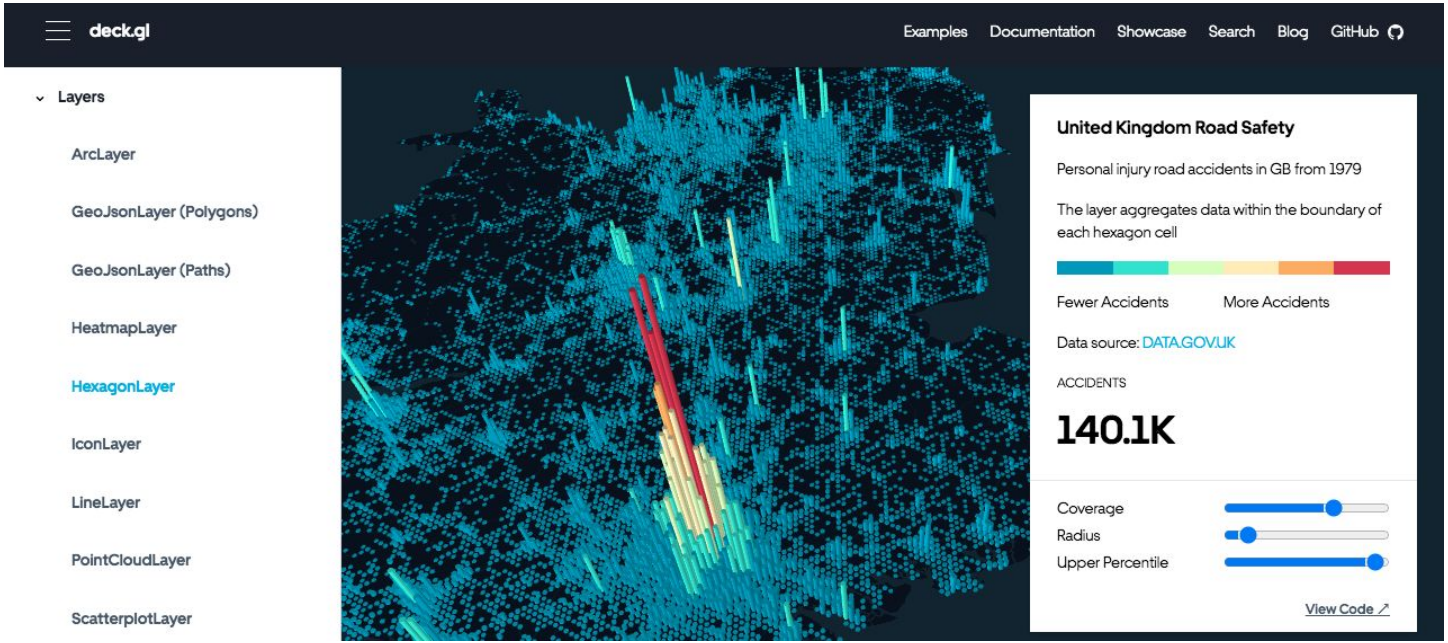


Ejemplo Map - Largo



Fuente: Span Chart - Learn about this chart and tools to create it

Ejemplo Map - Largo y color



Fuente: https://deck.gl/examples/hexagon-layer/

Código en Python

Vamos al código 👨‍💻 👩‍💻

Interactions

- Manipulate
- Facet
- Reduce

Manipulate

Llamamos manipulación a toda decisión de diseño que realice un **cambio** en lo que se muestra.

Una decisión de manipulación convierte nuestra visualización estática en una interactiva.

Este tipo de decisiones de diseño se puede dividir en 3 categorías no excluyentes:

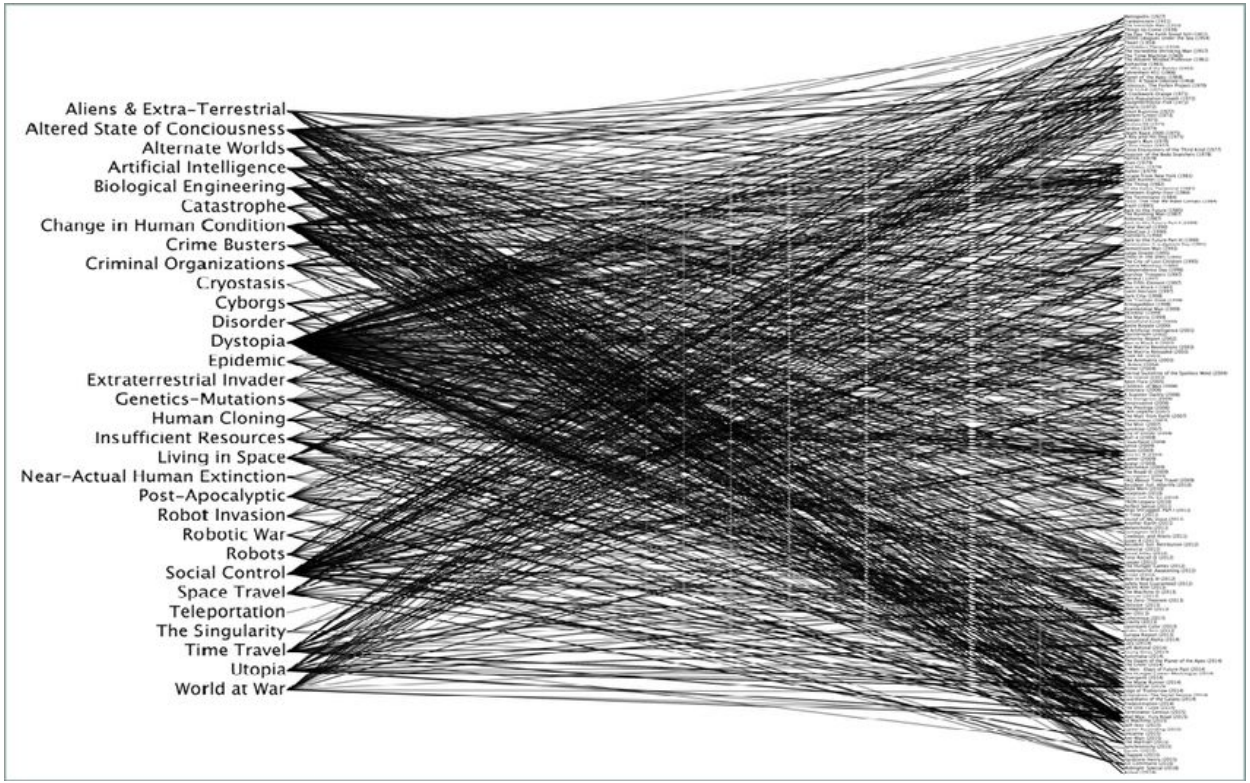
1. Cambios de la visualización en el tiempo (**change**).
2. Selección dentro de la visualización (**select**).
3. Navegación en la visualización (**navigate**).

Manipulate

¿Para qué queremos agregar interactividad?

1. El usuario logra involucrarse más con la visualización.
2. Gran mecanismo para evitar el desorden visual (***visual cluttering***):
    a. Si disponemos de forma estática todo lo que queremos mostrar, el usuario podría confundirse, no sabría por dónde empezar. Sin interactividad, tendríamos que crear muchas visualizaciones estáticas que aborden distintos puntos de partida, lo que consume muchos recursos.
    b. Los datasets son cada vez más complejos: tratar de mostrar todo su contenido al mismo tiempo, sin interacción con la visualización, lleva a desorden visual.

Manipulate - Ejemplo de *visual cluttering*



Fuente: DATA VISUALIZING POPULAR SCIENCE FICTION MOVIES WITH USE OF CIRCULAR HIERARCHIAL EDGE BUNDLING, Selçuk Artut

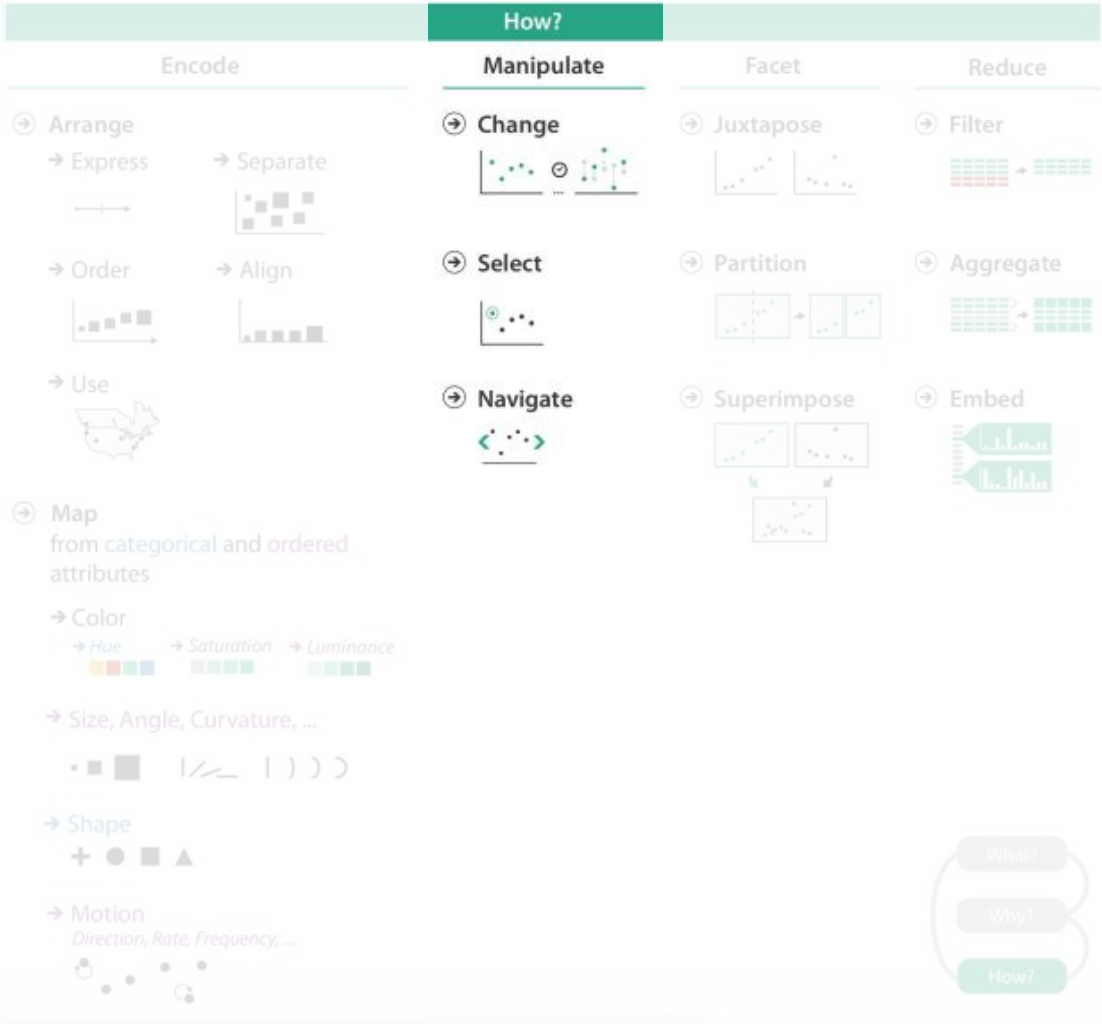
Manipulate: Change

Manipulate

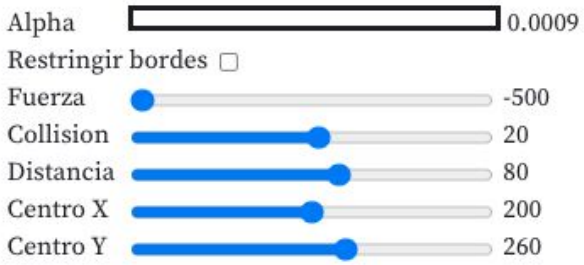
Describen acciones que puede realizar el usuario con los elementos de la visualización (marcas).

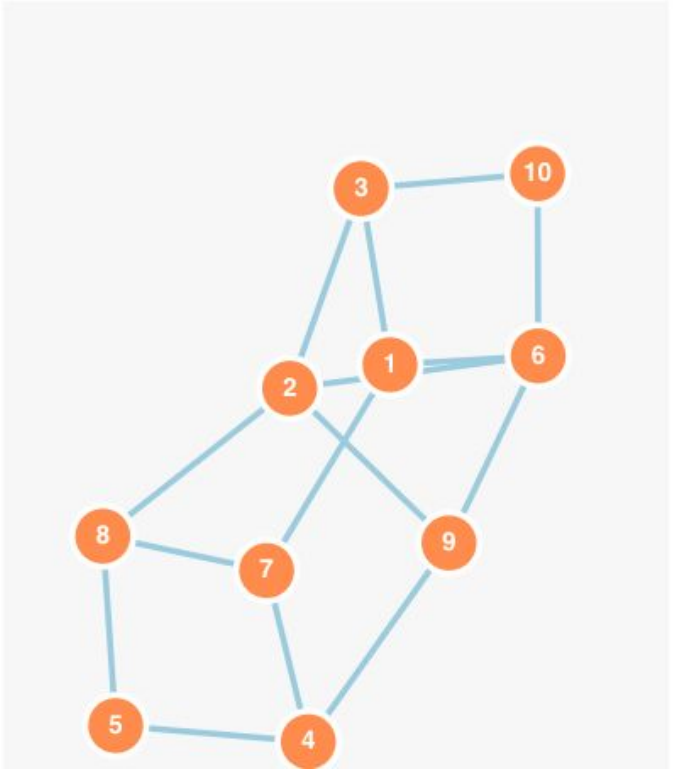
Change

- Cambiar la codificación de los datos.
- Cambiar la disposición de los datos (por ejemplo, ordenar)
- Cambiar el/los canales utilizados (Pasar de color a tamaño).
- Modificar el canal utilizado (pasar de una escala secuencial de 3 colores a 5 colores).



Manipulate - Change: Ejemplo





Fuente: Force Directed Graph / Hernan VALDIVIESO LOPEZ | Observable

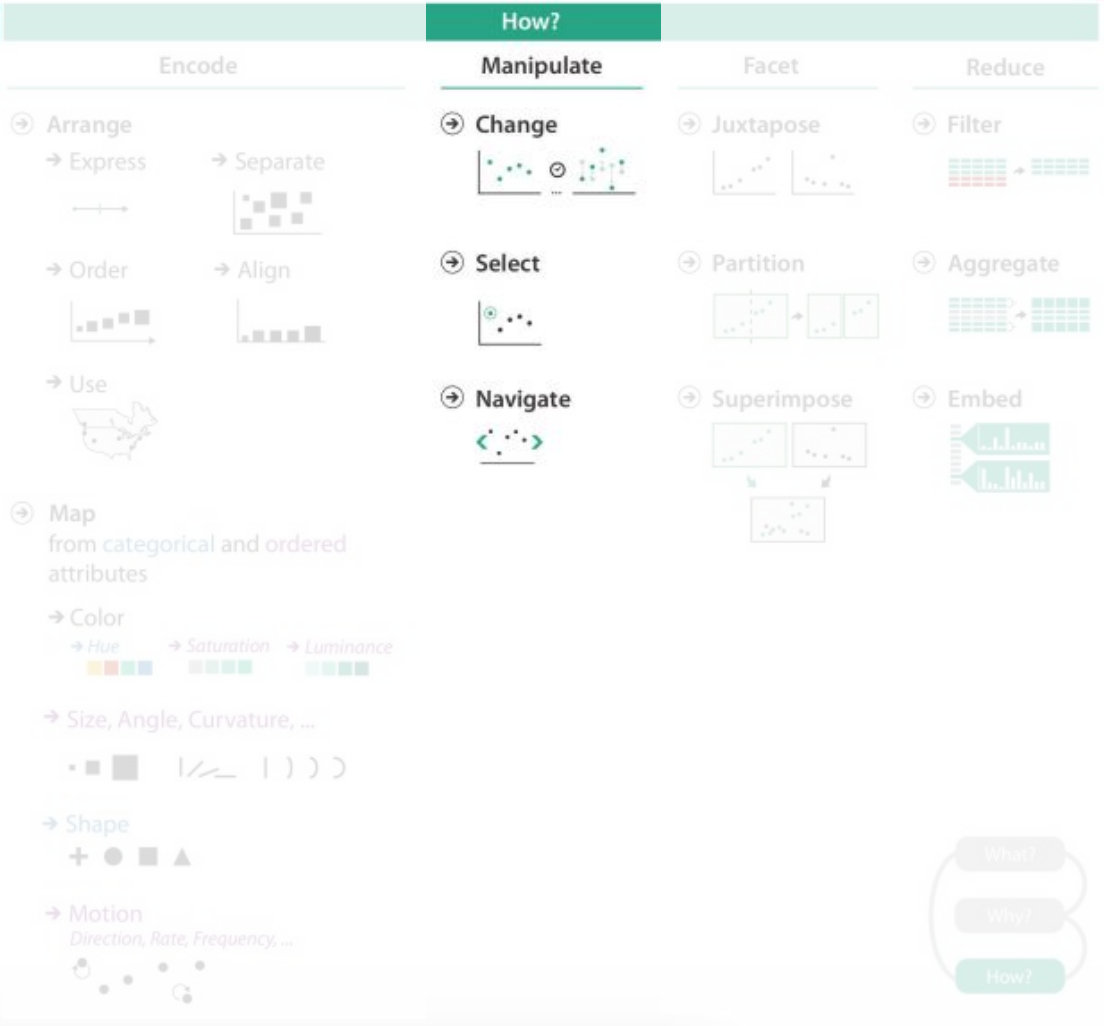
Manipulate: Select

Manipulate

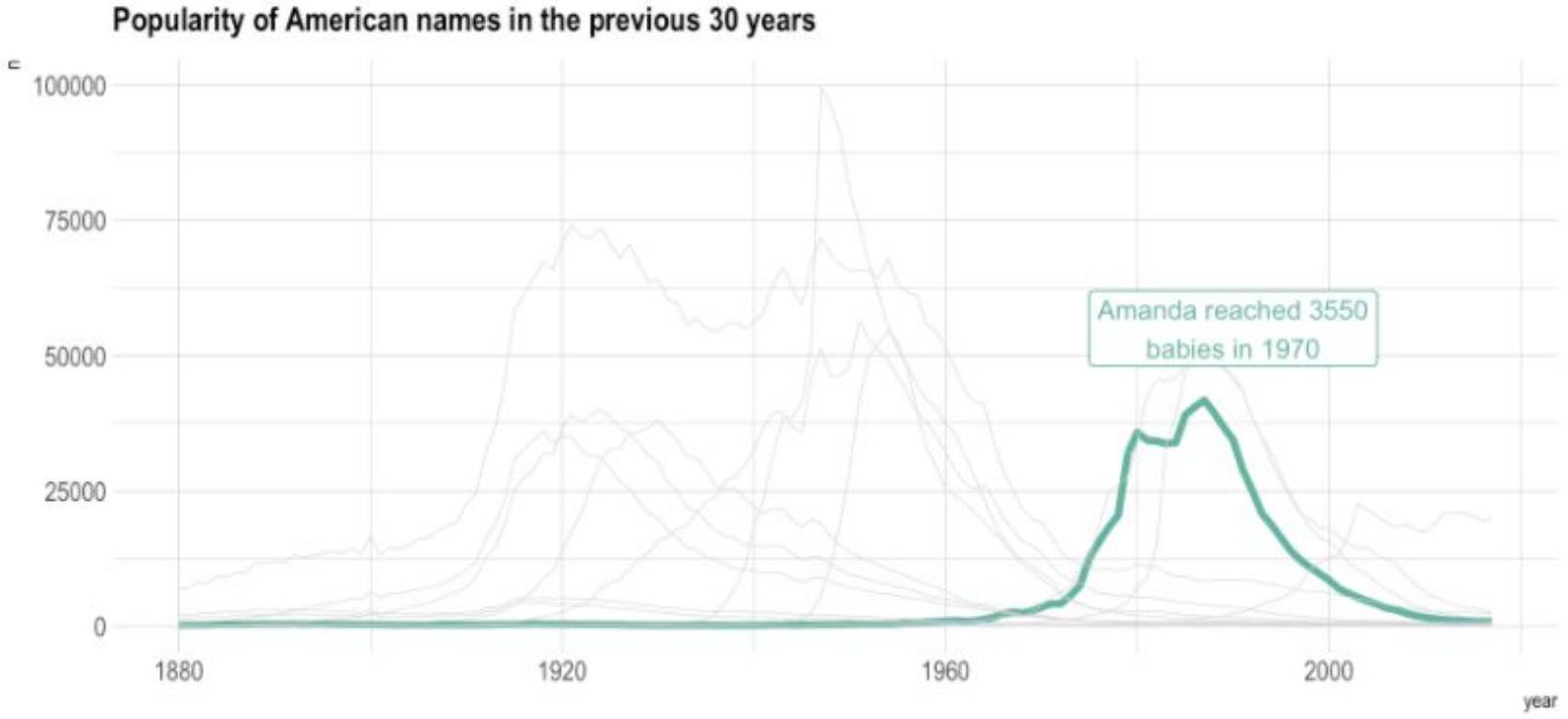
Describen acciones que puede realizar el usuario con los elementos de la visualización (marcas).

Select

- Seleccionar/enfatizar uno o más elementos.
- Seleccionar los atributos a observar.
- Dejar un registro de cuales datos ya fueron seleccionados alguna vez.



Manipulate - Select: Ejemplo



Fuente: The Spaghetti plot

Manipulate: Navigate

Manipulate

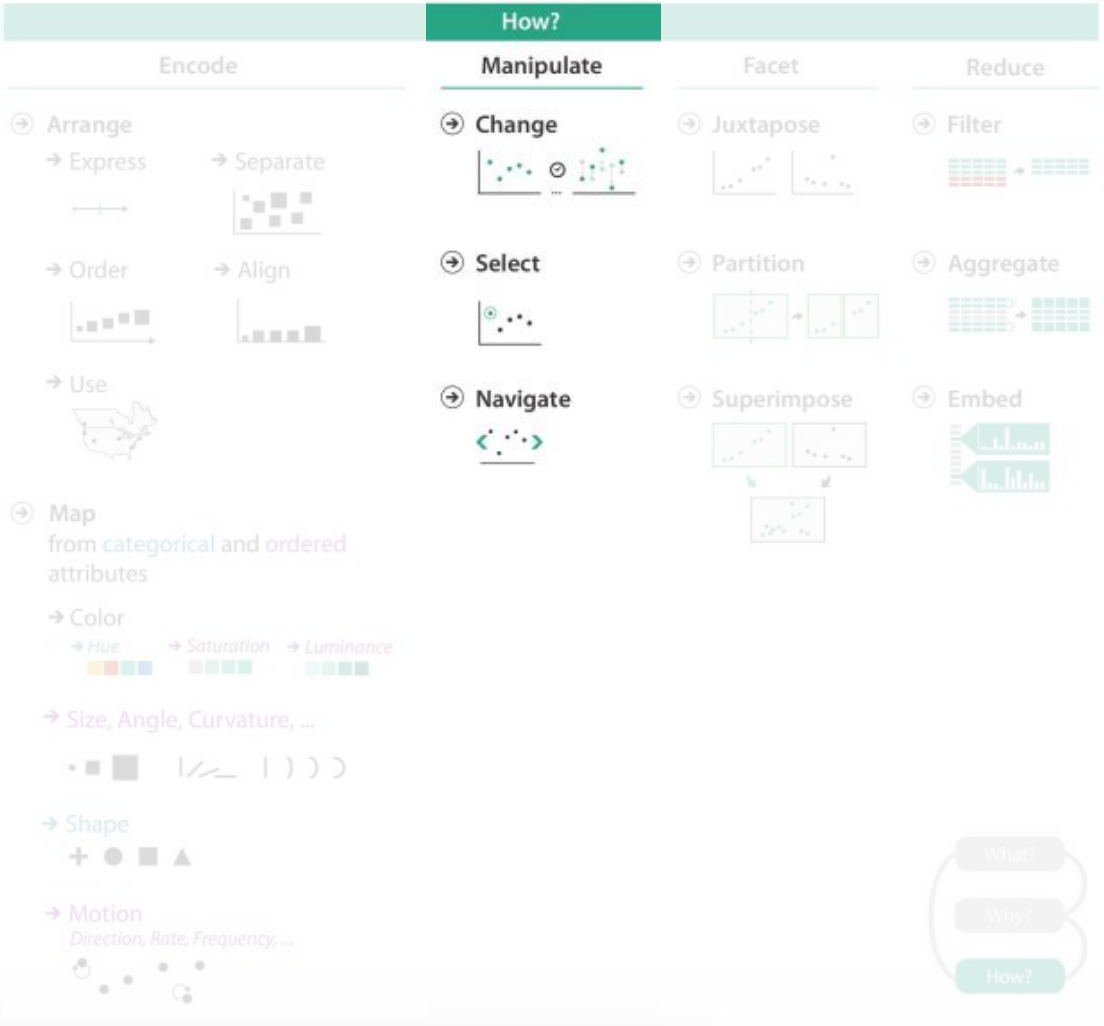
Describen acciones que puede realizar el usuario con los elementos de la visualización (marcas).

Navigate

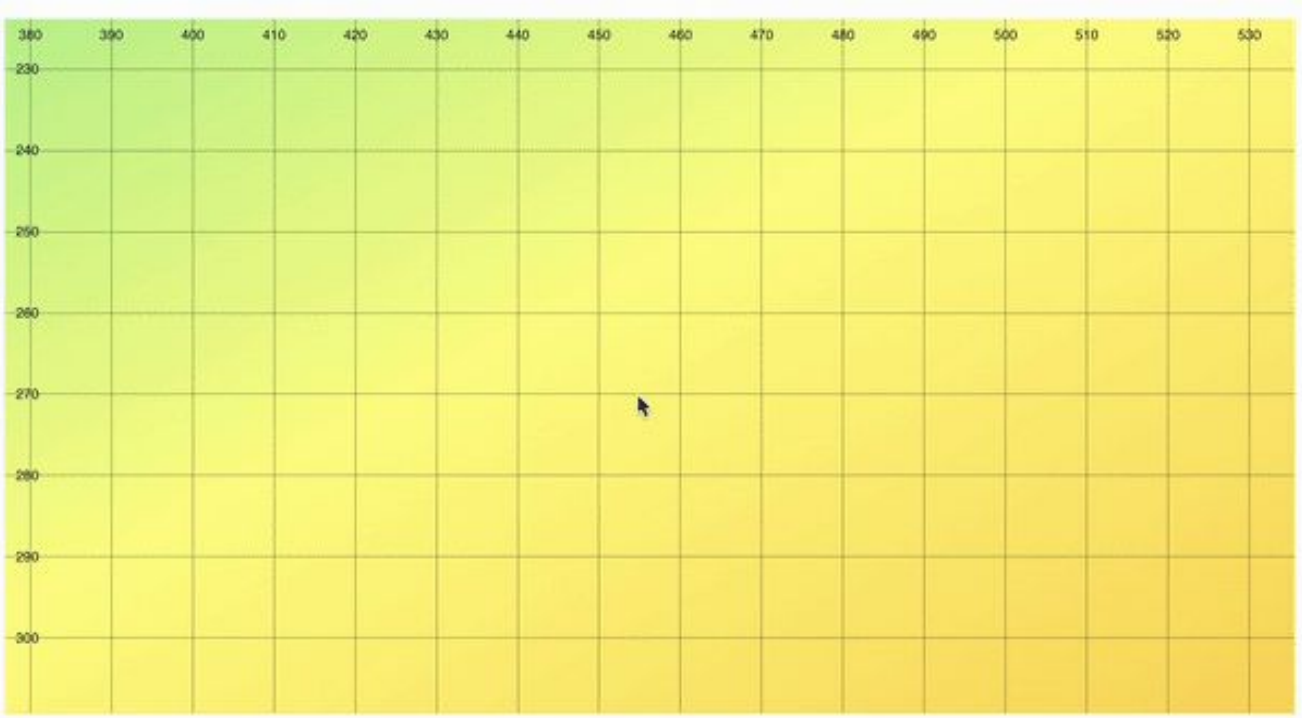
- Cambiar el *viewpoint* (punto de vista) con *panning*, *translating* y/o *zooming*.

- *Panning* (cambiar la sección de la visualización que se ve) . Cuando estamos en un contexto de tres dimensiones, se hablará de *translating*.

- *Zooming* (acercarse o alejarse de una zona)



Manipulate - Ejemplos



Fuente: Pan & Zoom Axes / D3 | Observable

Código en Python

Vamos al código 👨‍💻 👩‍💻

Facet

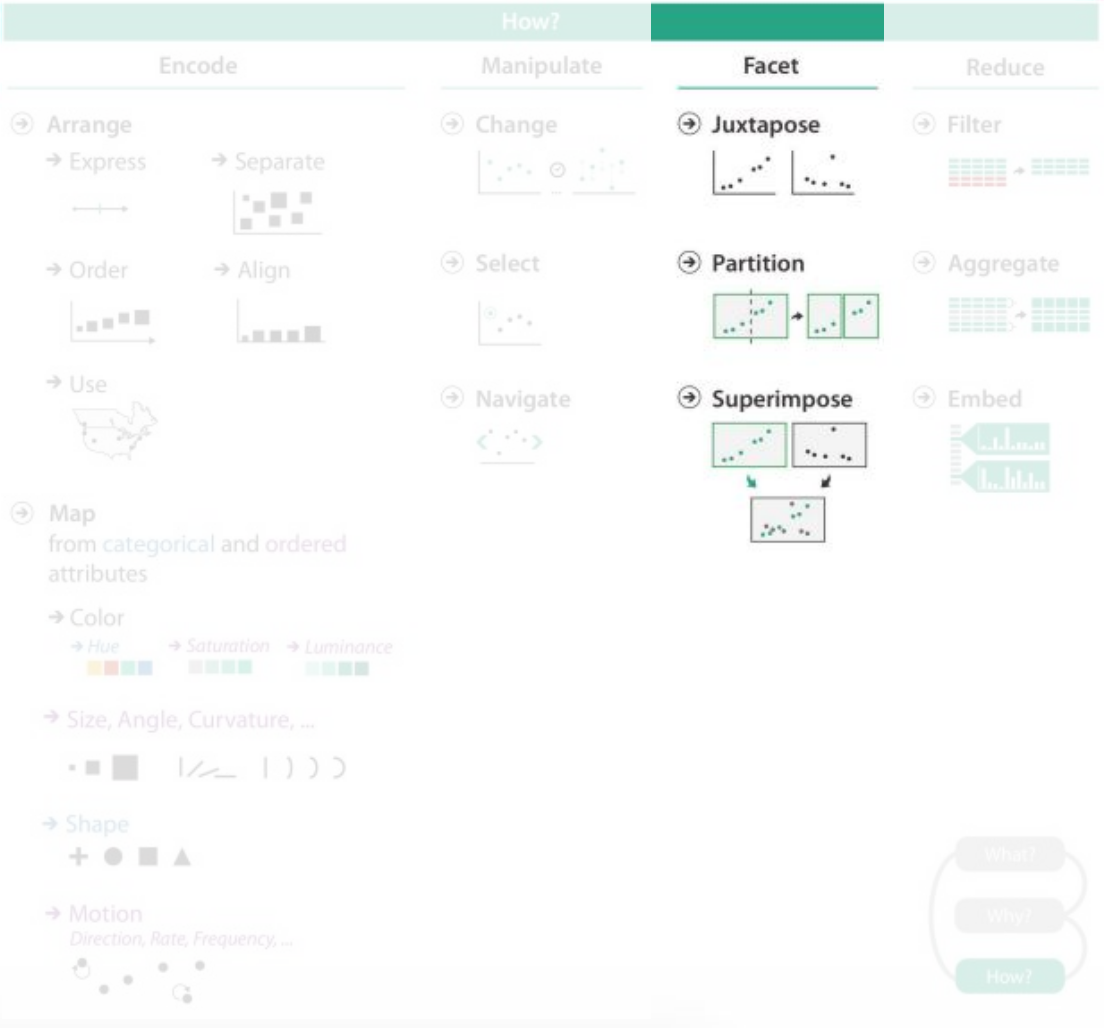
Facet

La idea de facet es mostrar diferentes ángulos de un dataset, dividiendo la visualización en diferentes vistas.

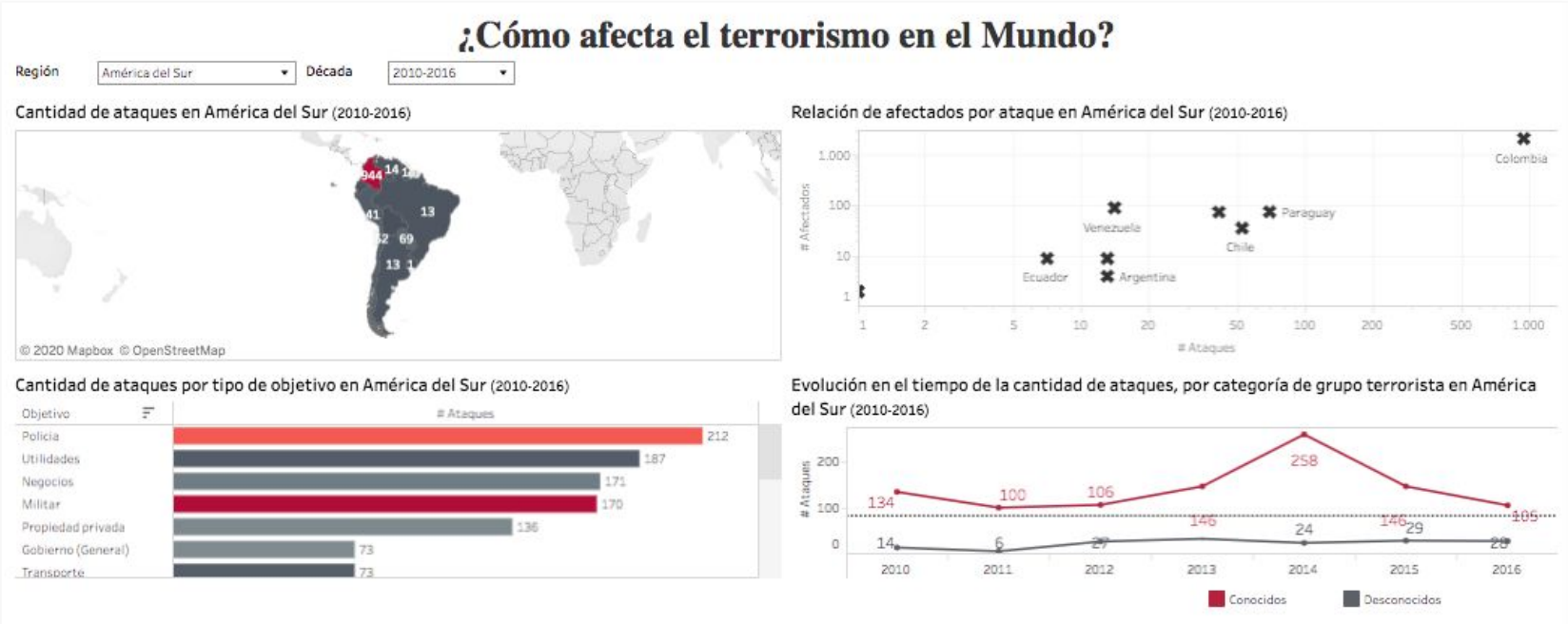
juxtapose

- Una o más vistas simultáneas. Generalmente coordinadas entre ellas.

- Se debe elegir cómo coordinar las vistas entre ellas, cuántos datos/atributos compartir, qué canales utilizar, etc.



Facet - Ejemplos



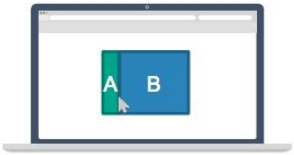
Fuente: https://public.tableau.com/profile/andr.s.parra#!/vizhome/GlobalTerrorism1970-2015_0/TerrorismoGlobal

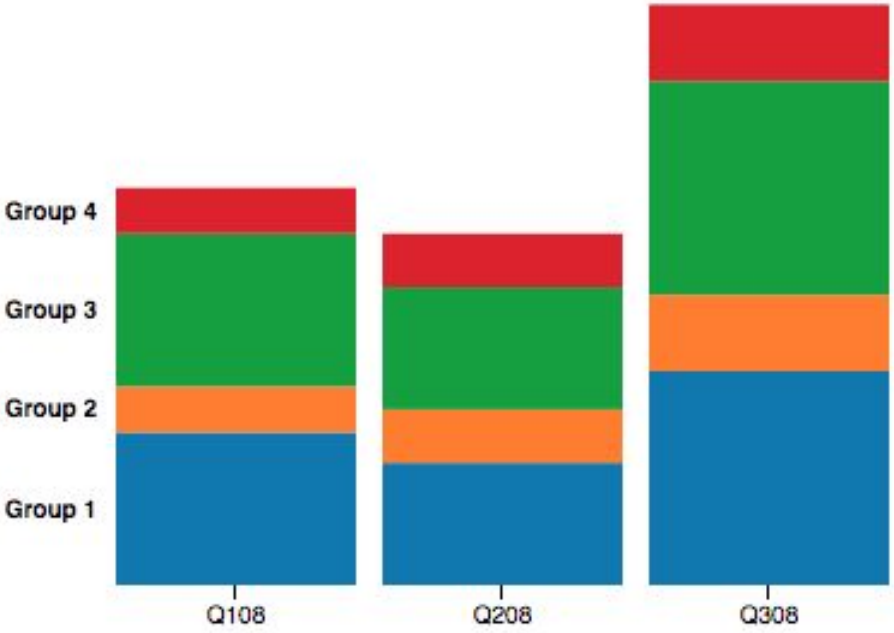
Facet

La idea de facet es mostrar diferentes ángulos de un dataset, dividiendo la visualización en diferentes vistas.

Partition

- Se debe elegir cuántas regiones utilizar, cómo dividir los datos entre ellas, o el orden de los atributos al utilizar.




Group 4
Group 3
Group 2
Group 1
Q108
Q208
Q308


Group 4
Group 3
Group 2
Group 1
Q108
Q208
Q308

Small multiple

Facet

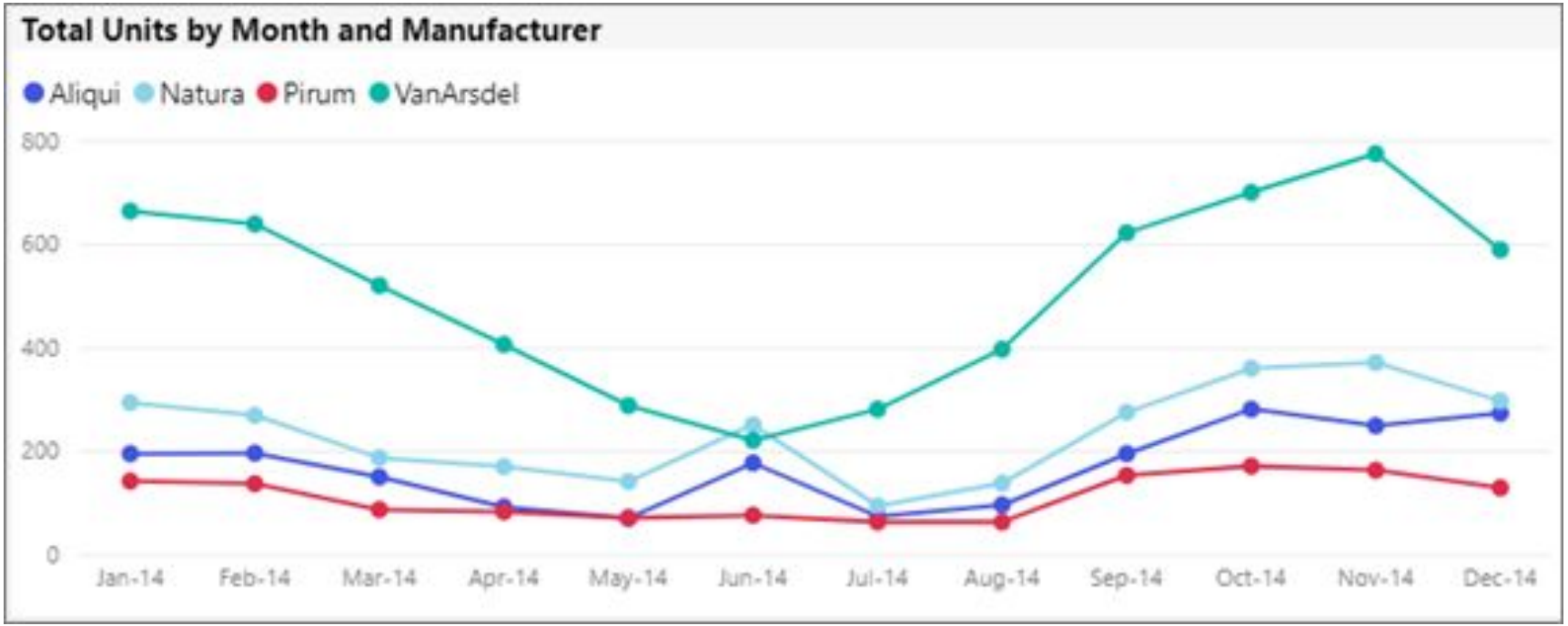
La idea de facet es mostrar diferentes ángulos de un dataset, dividiendo la visualización en diferentes vistas.

Superimpose

- Se sitúan N visualizaciones una encima de otra.

- Se debe elegir cómo los elementos serán particionados en las distintas capas, cuántas capas usar, etc.

Gráfico múltiples líneas



Código en Python

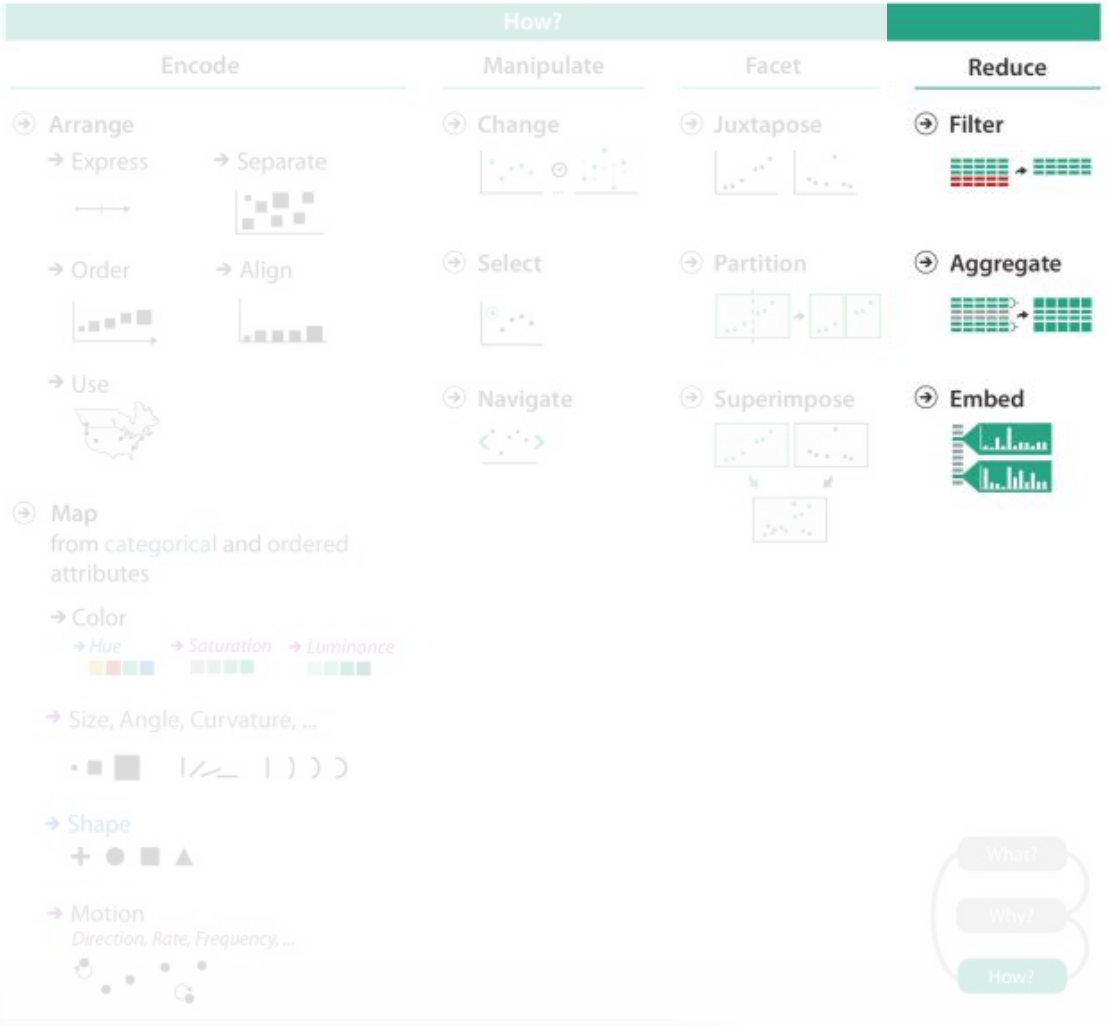
Vamos al código 👨‍💻 👩‍💻

Reduce

Reduce

Esta familia tiene por objetivo manejar la complejidad del dataset.

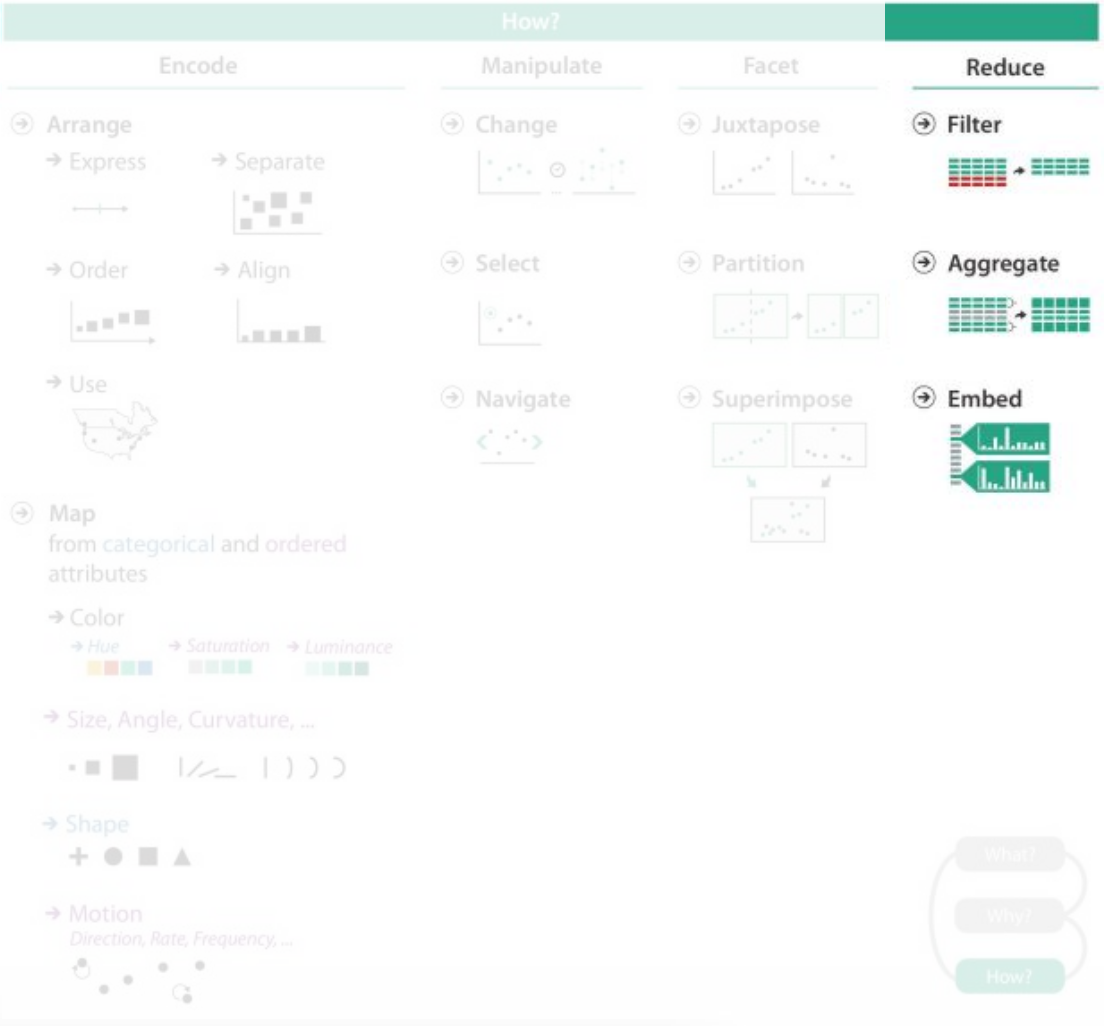
- ***Filter*** permite eliminar la cantidad de elementos mostrados (e.g. por uno o más rangos de interés)



Reduce

Esta familia tiene por objetivo manejar la complejidad del dataset.

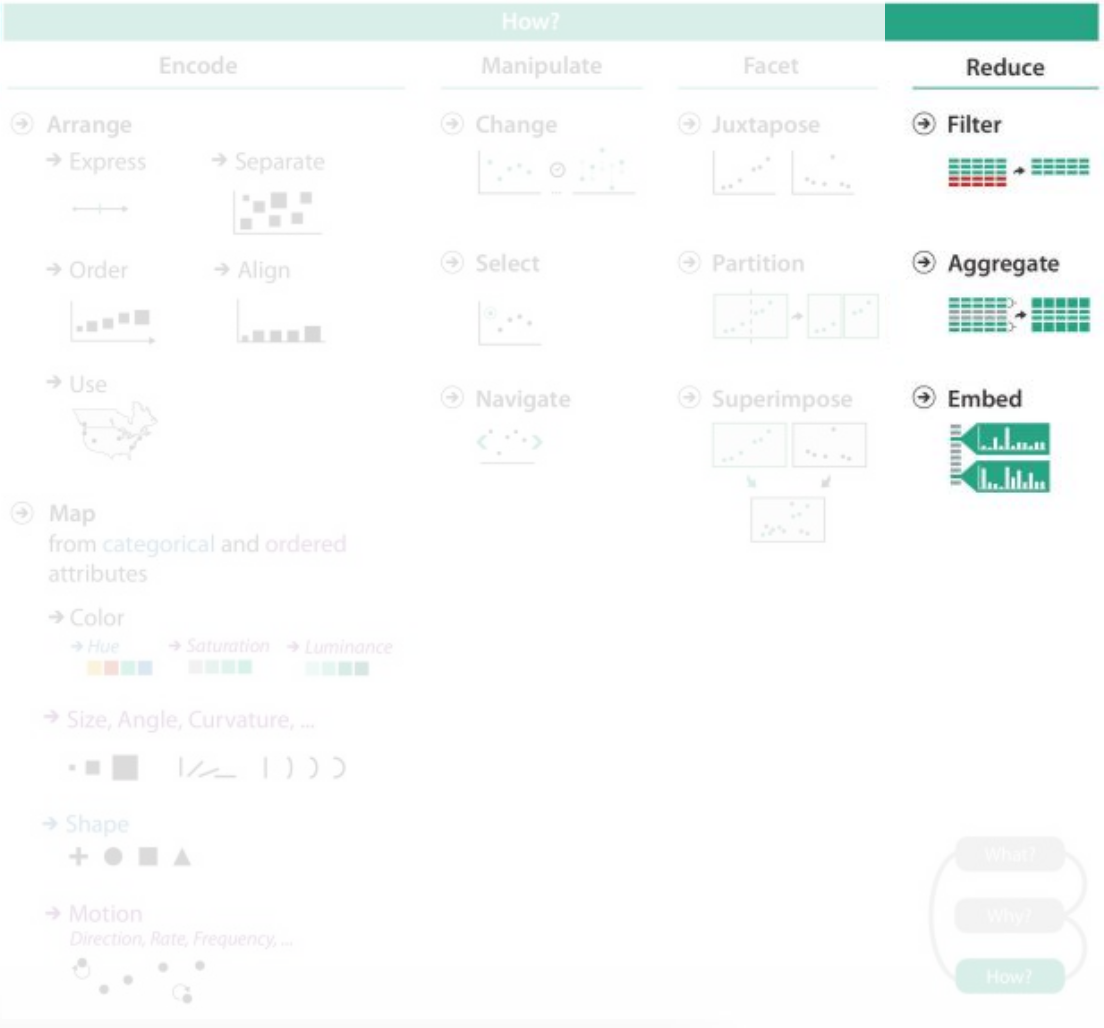
- ***Aggregate*** busca que un grupo de elementos sea representado por un nuevo elemento que los represente; de esta forma, se hace un *merge* (e.g. obtener el promedio)



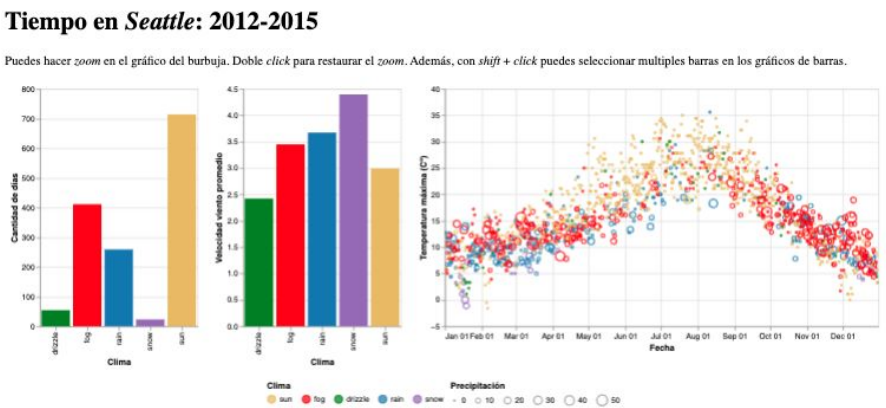
Reduce

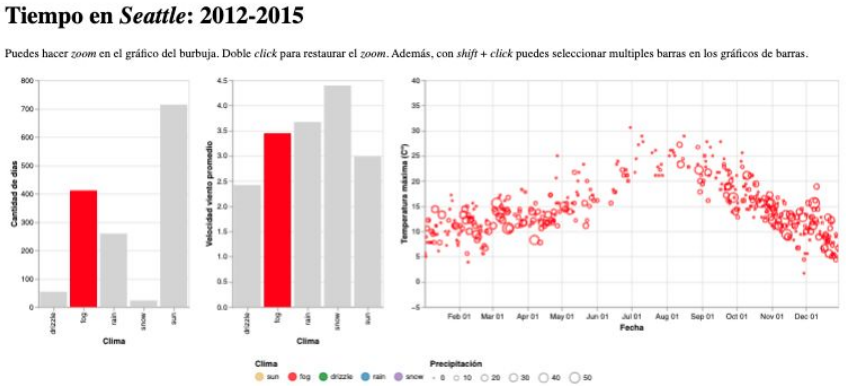
Esta familia tiene por objetivo manejar la complejidad del dataset.

- ***Embeber*** permite reducir la cantidad de elementos mediante una sofisticada combinación de filtrado y agregación. (e.g. lente especial o *tooltip*)



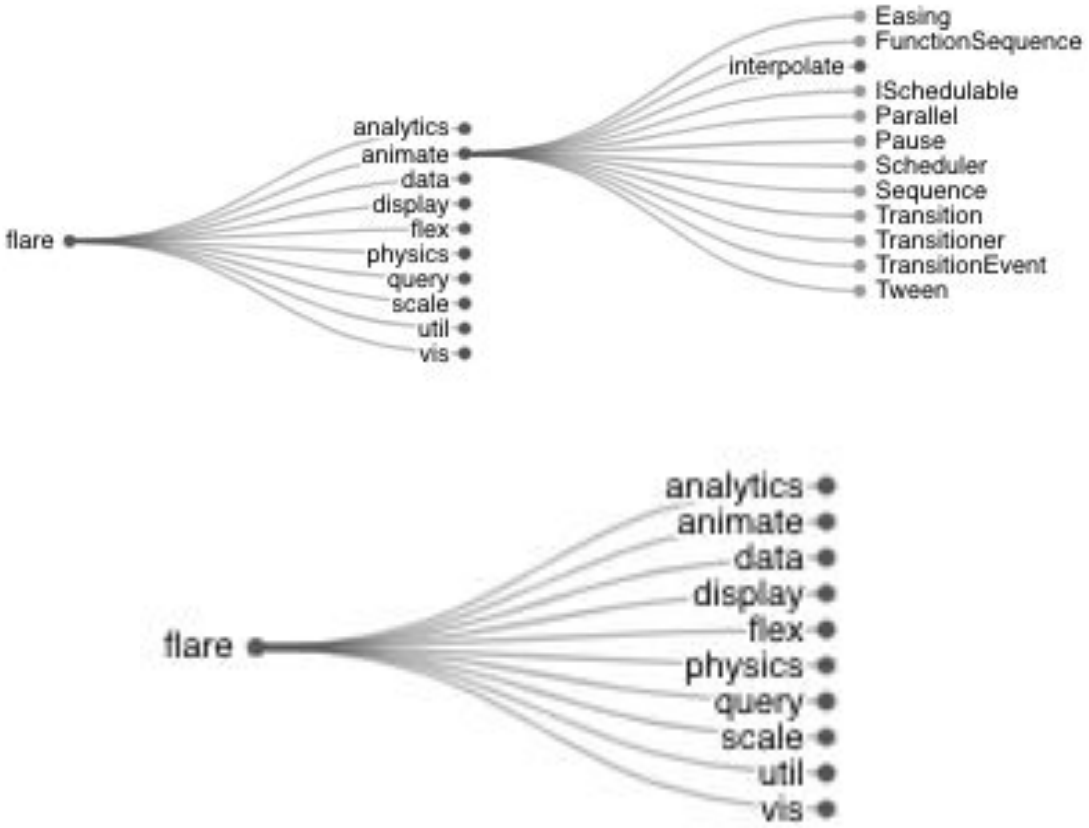
Reduce - Ejemplos





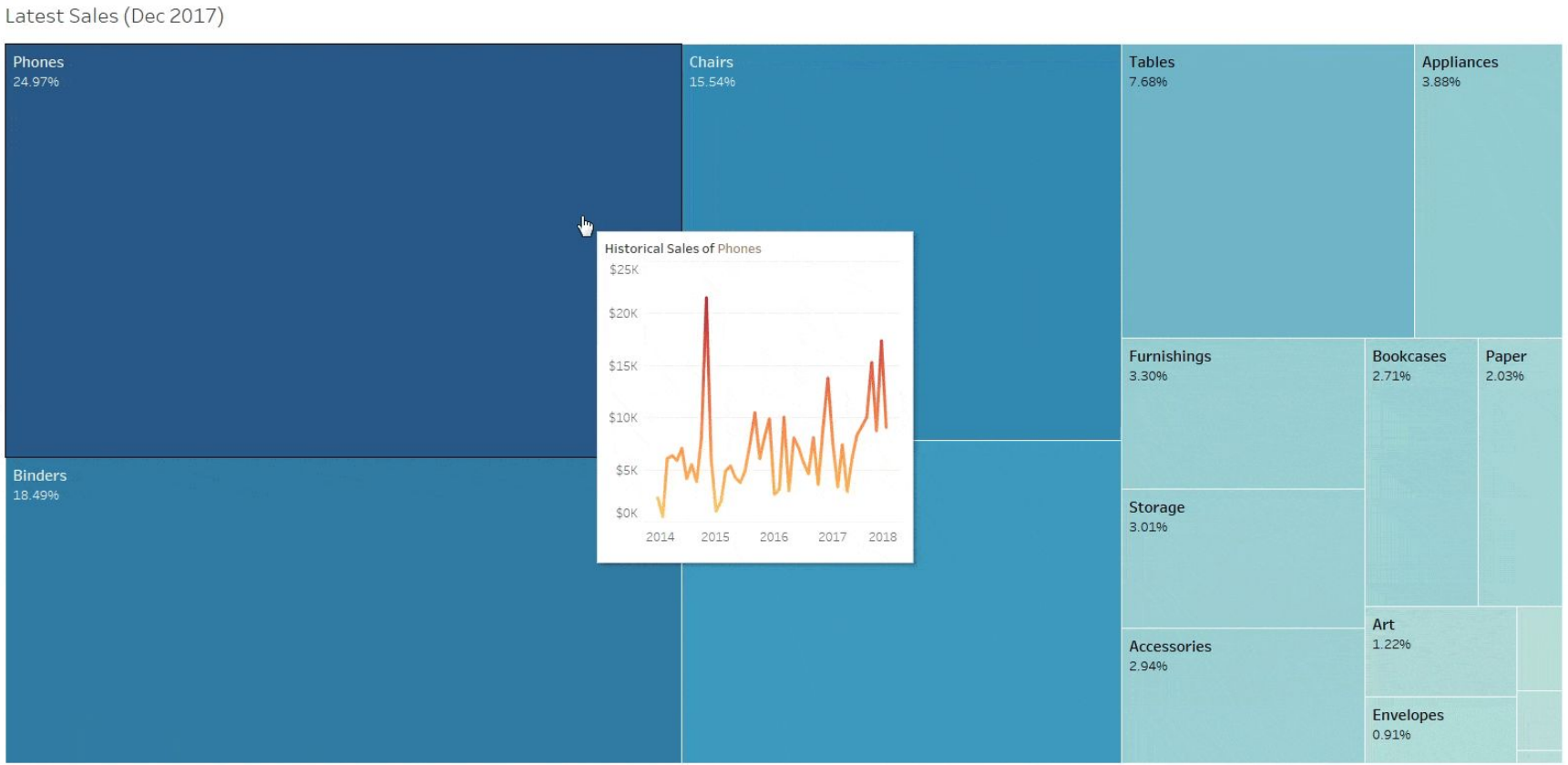
Fuente: Tiempo en Seattle 2012-2015

Reduce - Ejemplos

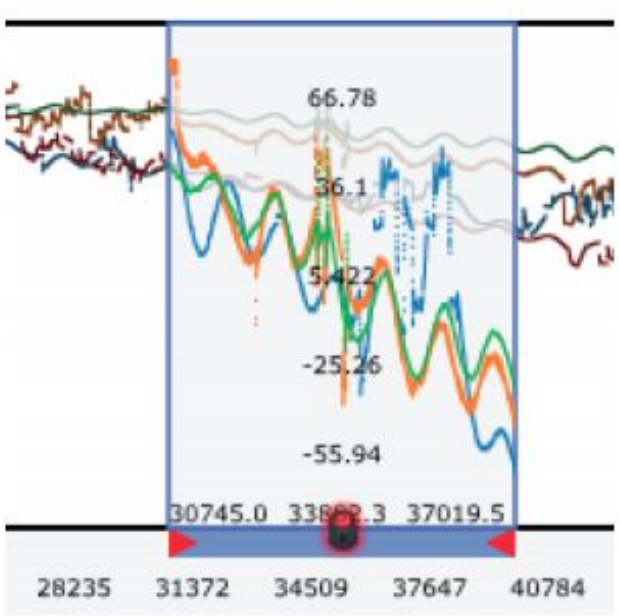


Fuente: Collapsible Tree / D3 | Observable

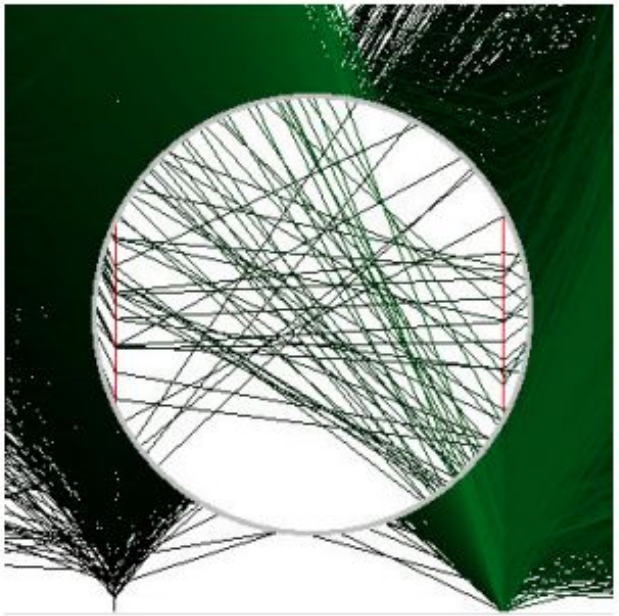
Reduce - Ejemplos



Reduce - Ejemplos



(a) Alteration



(b) Suppression

[ChronoLenses and Sampling Lens in Tominski et al., 2014]

Código en Python

Vamos al código 👨‍💻 👩‍💻

How

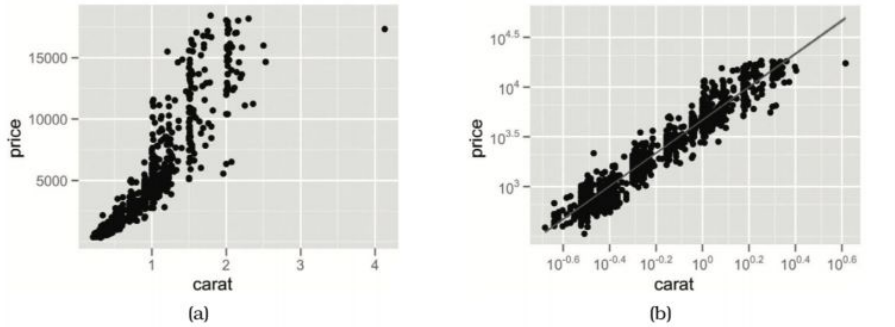
¿Cómo elegimos entre las opciones que tenemos disponibles?

Debemos considerar al menos:

- Percepción
- Memoria
- Tarea a resolver
- Usuario objetivo
- Canales disponibles
- Marcas disponibles
- Interacciones entre marcas y canales
- Eficiencia de canales
- Algunas reglas basadas en la experiencia (las veremos la próxima clase)

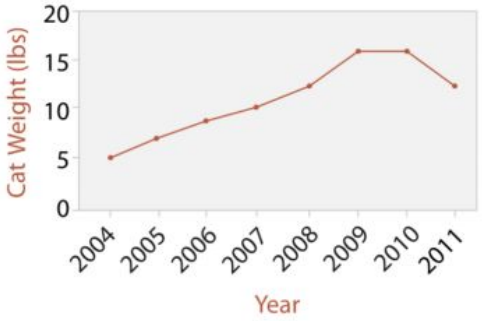
¿Cómo Tamara Munzner utiliza su framework para el análisis?

Gráfico de dispersión *(scatterplot)*



Idiom	**Scatterplots**
What: Data	Table: two quantitative value attributes.
How: Encode	Express values with horizontal and vertical spatial position and point marks.
Why: Task	Find trends, outliers, distribution, correlation; locate clusters.
Scale	Items: hundreds.

Gráfico de línea *(line chart)*



(b)

Idiom	**Line Charts**
What: Data	Table: one quantitative value attribute, one ordered key attribute.
How: Encode	Dot chart with connection marks between dots.
Why	Show trend.
Scale	Key attribute: hundreds of levels.

Control

Presentación

Visualización de Información y Analítica Visual

Daniela Flores (diflores@uc.cl)
Hernán Valdivieso (hvaldivieso@ing.puc.cl)