Introdução a Ciência de Dado

Professor: Alex Pereira

Vega-Lite: Dados

- Os dados no vega-lite são tabulares
  - Semelhante a uma planilha ou um dataframe
- Tipos de data source
  - inline (values)
  - url
  - named data source
    - ✓ carregado em tempo de execução

inline data source

```
"data": {
  "values": [
    {"a": "A", "b": 28}, {"a": "B", "b": 55},
{"a": "C", "b": 43},
    {"a": "D", "b": 91}, {"a": "E", "b": 81},
{"a": "F", "b": 53},
    {"a": "G", "b": 19}, {"a": "H", "b": 87},
{"a": "I", "b": 52}
    ]
  },
```

Vega-Lite: Dados de uma URL

- Formato padrão
  - json
    - ✓ Outros: csv, tsv, dsv, topojson
- Pode fazer um parse (conversão) de tipos (number, date, boolean)
  - com a propriedade "parse"
    - ✓ Especificadores da conversão de data

```
  "data": {
    "url":"https://raw.githubusercontent.com/alexlopespereira/enapespcd2021/main/data/originais/aquecimento_global/global_temperature_anomalies_tratado.csv",
    "format": {
                "type": "csv",
                "parse": {"data1": "date:'%Y-%m-%d'", "Anomaly": "number"}
        },
    "name":"temperature"
  }
```

Em geral, o vega-lite interpreta corretamente

Vega-Lite: Dados de uma URL formato JSON

- Quando os dados estiverem numa estrutura interna do JSON
  - usar a propriedade "property"


```
{  "data":{
   "values":[
     {"Anomaly":-2.7,"ano":1880,"mes":"jan","mes_num":1,"data1":"1880-1-01"},
     {"Anomaly":-2.33,"ano":1880,"mes":"feb","mes_num":2,"data1":"1880-2-01"},
     {"Anomaly":-1.58,"ano":1880,"mes":"mar","mes_num":3,"data1":"1880-3-01"},
     {"Anomaly":-0.67,"ano":1880,"mes":"apr","mes_num":4,"data1":"1880-4-01"},
     {"Anomaly":0.35,"ano":1880,"mes":"may","mes_num":5,"data1":"1880-5-01"},
     ]  }}
```


```
"data": {
    "url":"https://raw.githubusercontent.com/alexlopespereira/enapespcd2021/mai
n/data/originais/aquecimento_global/global_temperature_anomalies_tratado.json",
    "format": {
              "type": "json", "property": "data.values",
              "parse": {"data1": "date:'%Y-%m-%d'", "Anomaly": "number"}
        },
    "name":"temperature"
  },
```

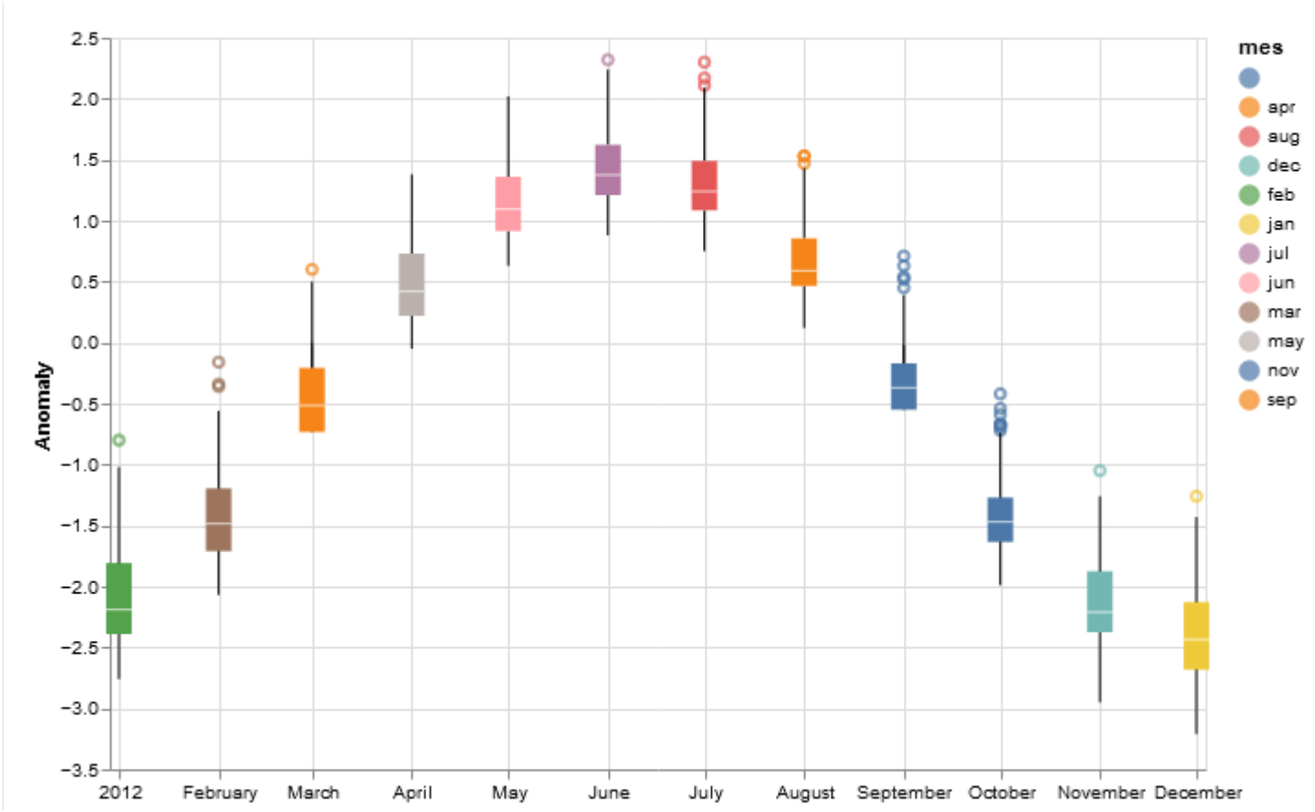
Encoding (link para o manual)

- Processo de mapear os dados para propriedades visuais
  - Ou seja, as primitivas gráficas (marks)
- Encoding channels (canais de codificação)
  - Posição (x e y),
  - cor (color),
  - tamanho (size)
  - anotação (text)
  - dicas (tootips)
- Tipos de dados dos canais
  - quantitative, temporal, ordinal, nominal
    - ✓ evita redundâncias: temporal é desnecessário
      - quando especificar a propriedade timeUnit
  - datum (para um valor constante), e
  - geojson

```
// Specification of a Single View
{
  "data": ... ,
  "mark": ... ,
  "encoding": {      // Encoding
    // Position Channels
    "x": ...,
    "y": ...,
    "x2": ...,
    "y2": ...,
    "xError": ...,
    "yError": ...,
    "xError2": ...,
    "yError2": ...,

    // Polar Position Channels
    "theta": ...,
    "radius": ...,
    "theta2": ...,
    "radius2": ...,

    // Geographic Position Channels
    "longitude": ...,
    "latitude": ...,
    "longitude2": ...,
    "latitude2": ...,

    // Mark Properties Channels
    "color": ...,
    "opacity": ...,
```

Atividade 9.1 (5 min)

- Plote um boxplot da anomalia de temperatura
  - Conforme exemplo abaixo
    - ✓ dica: use a propriedade mark do tipo boxplot

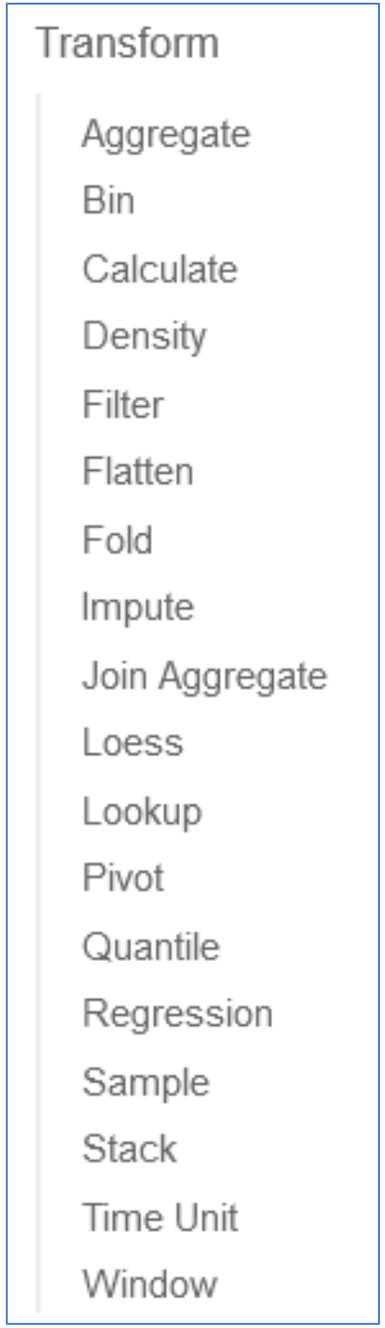


Vega-lite: Transform

- Transformações sobre o dado
  - Filtros, agregações, pivot, etc
    - ✓ Sintaxe da transformação

```
{
  ...
  "transform": [
    {"filter": DEFINIÇÃO DE PREDICADO } // Filter Transform
     ...
  ],
  ...
}
```

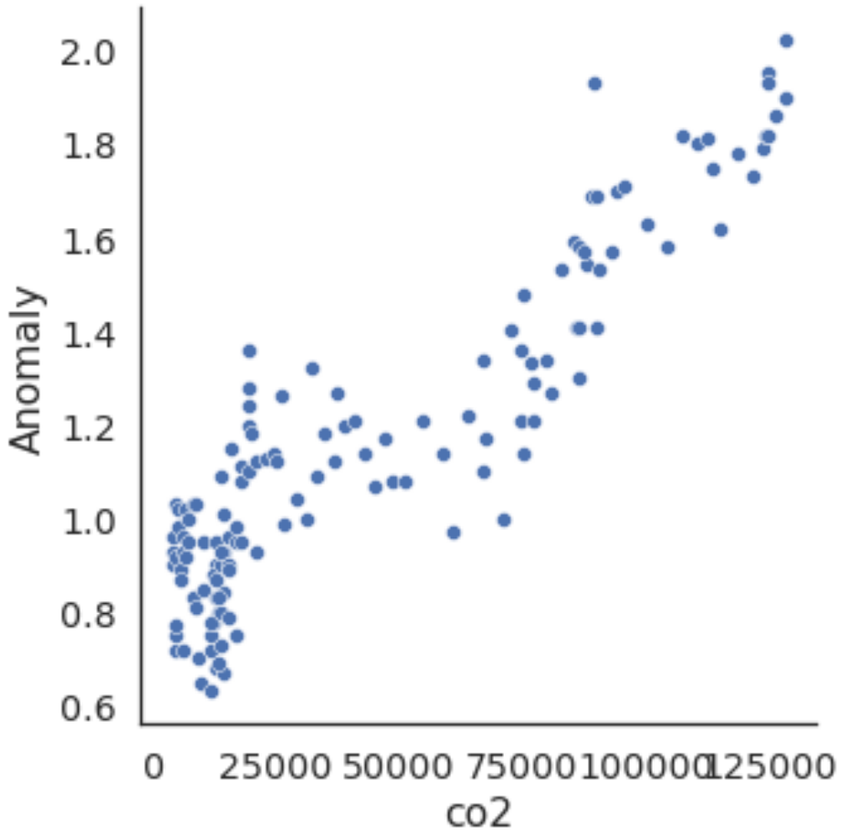
- Sintaxe dos predicados do filtro
  - {"field": "FIELD_NAME", "PREDICADO": "OPERANDO"}
    - ✓ {"field": "mes", "equal": "jun"}
  - "datum.FIELD_NAME=='OPERANDO'"
    - ✓ "datum.mes=='jun'"



Exemplo: Scatter Plot de CO2 vs Anomalia de Temperatura

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v5.json",
  "data": {
    "url":https://raw.githubusercontent.com/alexlopespereira/enapespcd2021/main/data/originais/aquecimento_global/temperature_anomalies_co2.csv, "format":{"type": "csv"}},
  "width": 400,
  "mark": "point",
  "encoding": {
    "x": {"field": "co2", "type": "quantitative"},
    "y": {"field": "Anomaly", "type": "quantitative"}
  }
}
```

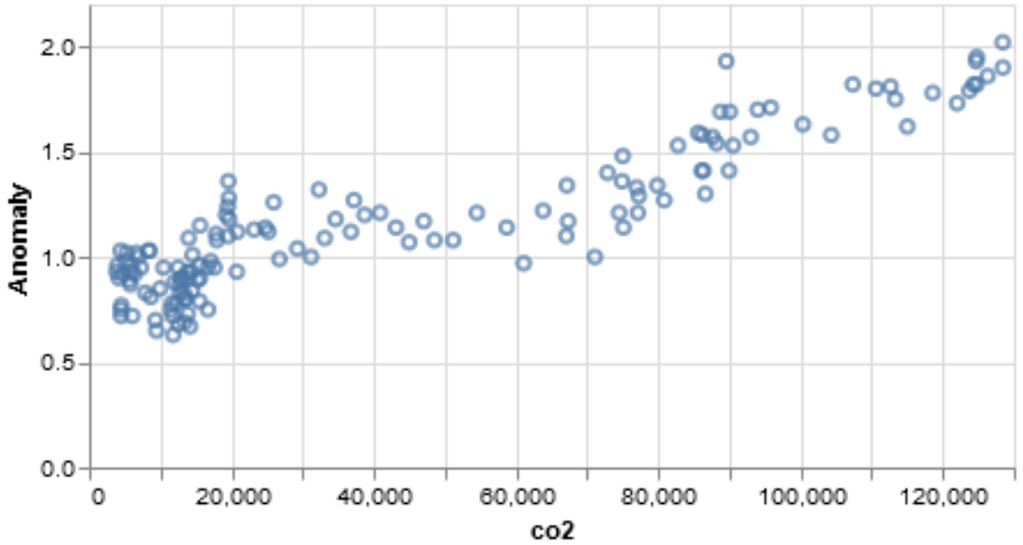
Exemplo da Aula 1



Exemplo: Scatter Plot de CO2 vs Anomalia de Temperatura

- Com filtro do mês de Junho

```
{
  "$schema": "https://vega.github.io/schema/vega-
lite/v5.json",
  "data": {
    "url":"https://raw.githubusercontent.com/alexlopesper
eira/enapespcd2021/main/data/originais/aquecimento_global
/temperature_anomalies_co2.csv",
  "format":{"type": "csv"}
  },
  "width": 400,
  "mark": "point",
  "transform": [ {"filter": "datum.mes=='jun'" }],
  "encoding": {
    "x": {"field": "co2", "type": "quantitative"},
    "y": {"field": "Anomaly", "type": "quantitative"}
  }
}
```

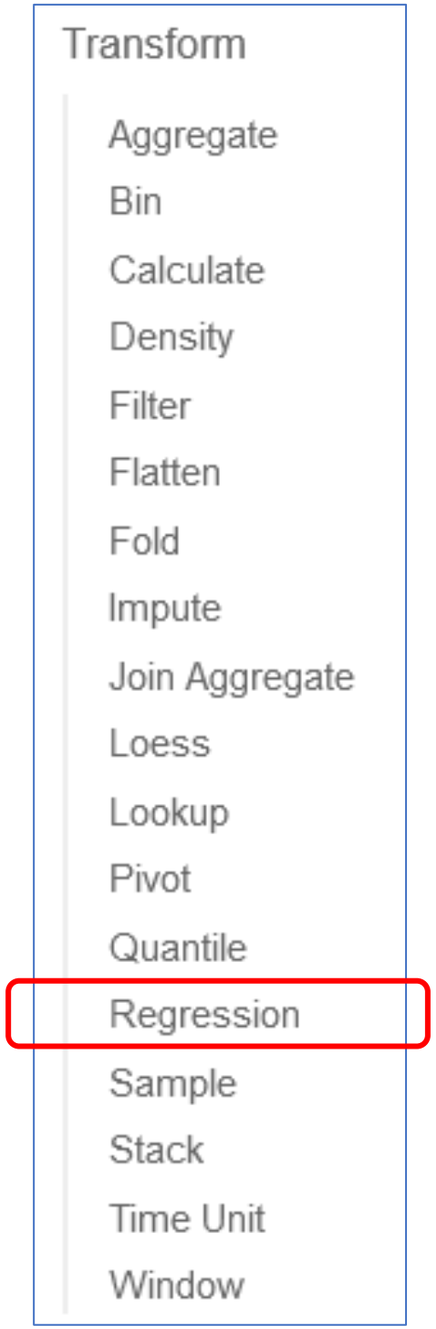


Transform: Regression

- Sintaxe da transformação de regressão

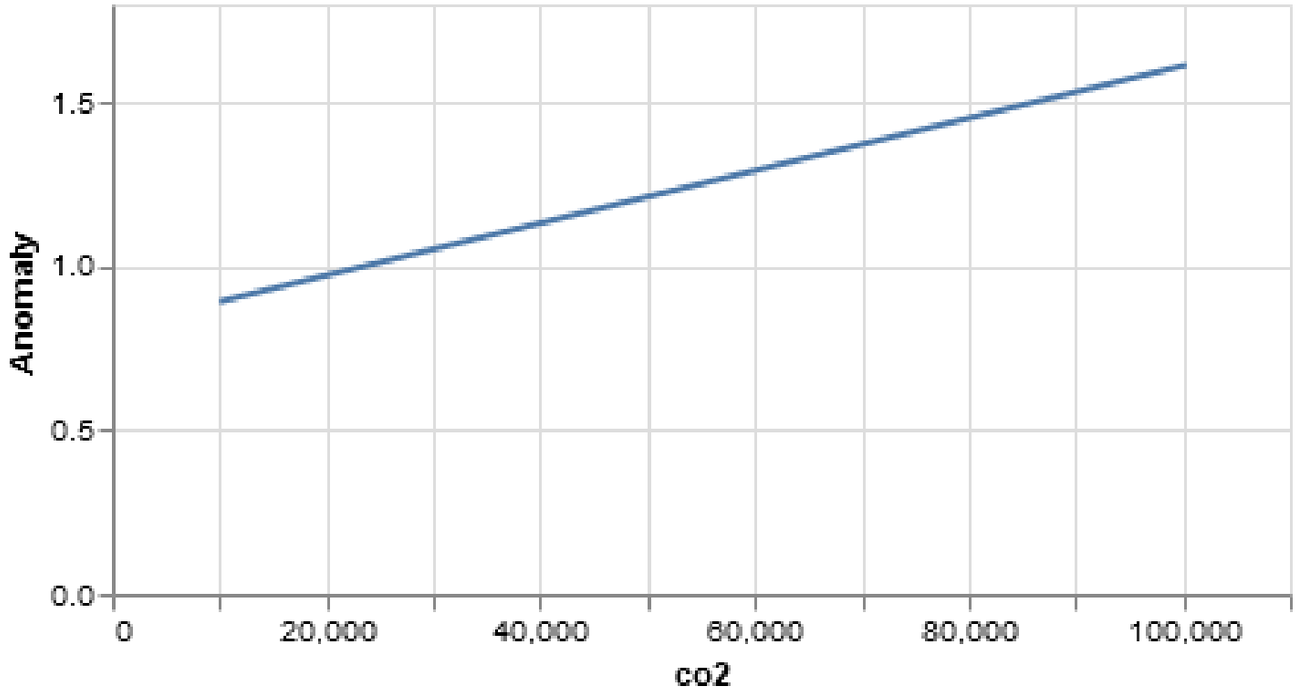
```
{
  ...
  "transform": [
    {"regression": "VARIÁVEL_DEPENDENTE", "on": "VARIÁVEL_INDEPENDENTE" }
  ],
  ...
}
```

- Existem outros argumentos
  - method (linear, log, exp, poly)
    - ✓ linear, por padrão



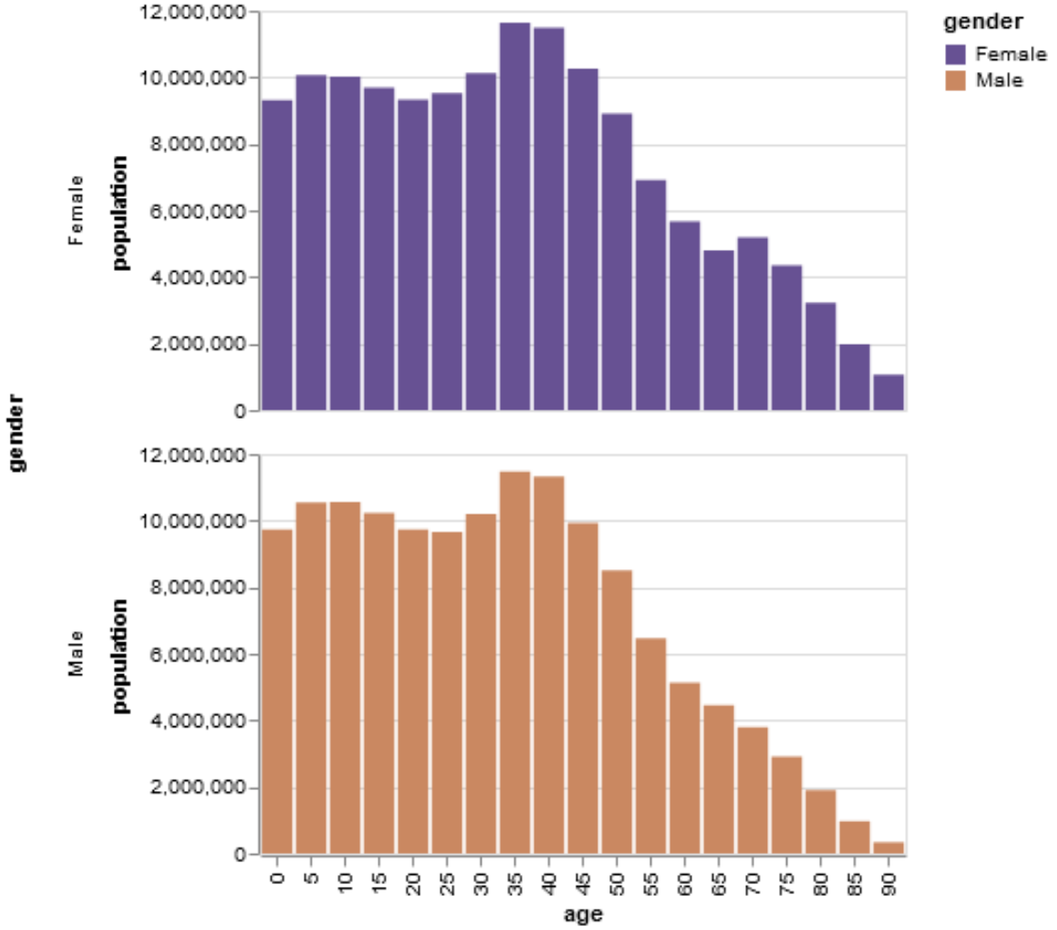
Atividade 9.2 (5 min)

- Plote a linha de uma regressão linear
  - Variável independente: CO2
  - Variável dependente: Anomalia da temperatura
- Dica: reutilize o filtro do mês

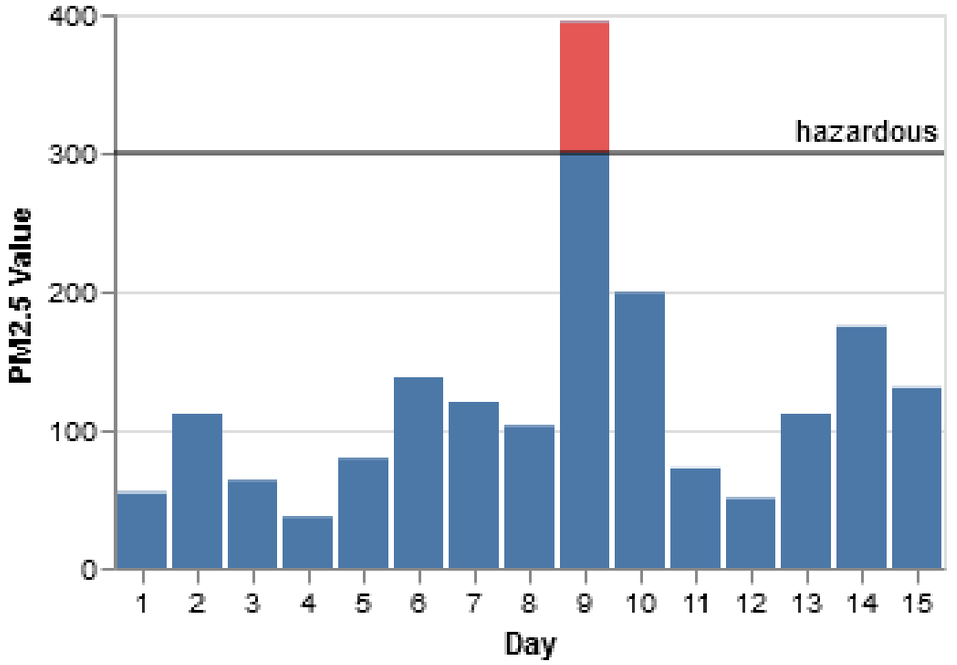


Visualizações Compostas (View Composition)

Facets



Layers



Layers

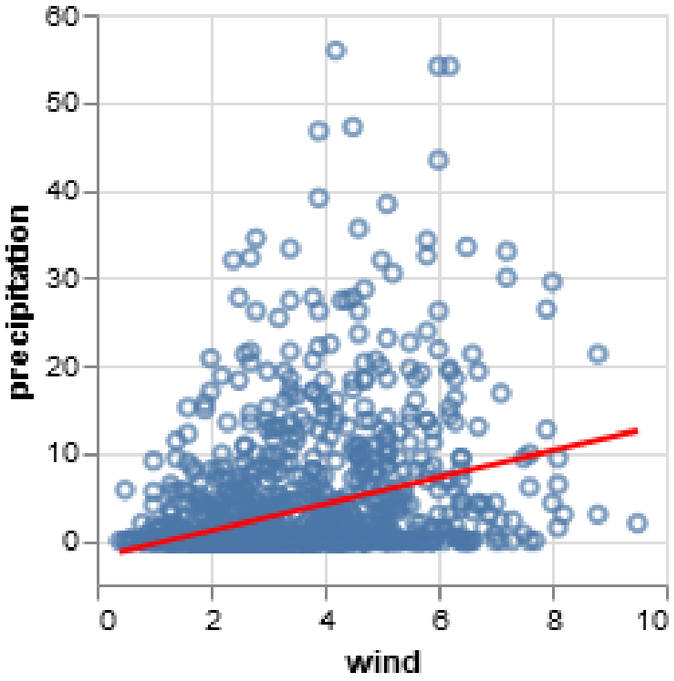
- Útil para sobrepor um gráfico sobre outro
  - É um dos tipos de composição de gráficos
    - ✓ Cada layer pode ser um gráfico, uma linha, uma anotação

```
{
  "data": {"url": "https://..."}
  "layer": [
    {...},
    {...},
    {...}
   ]
}
```

Exemplo com Layers

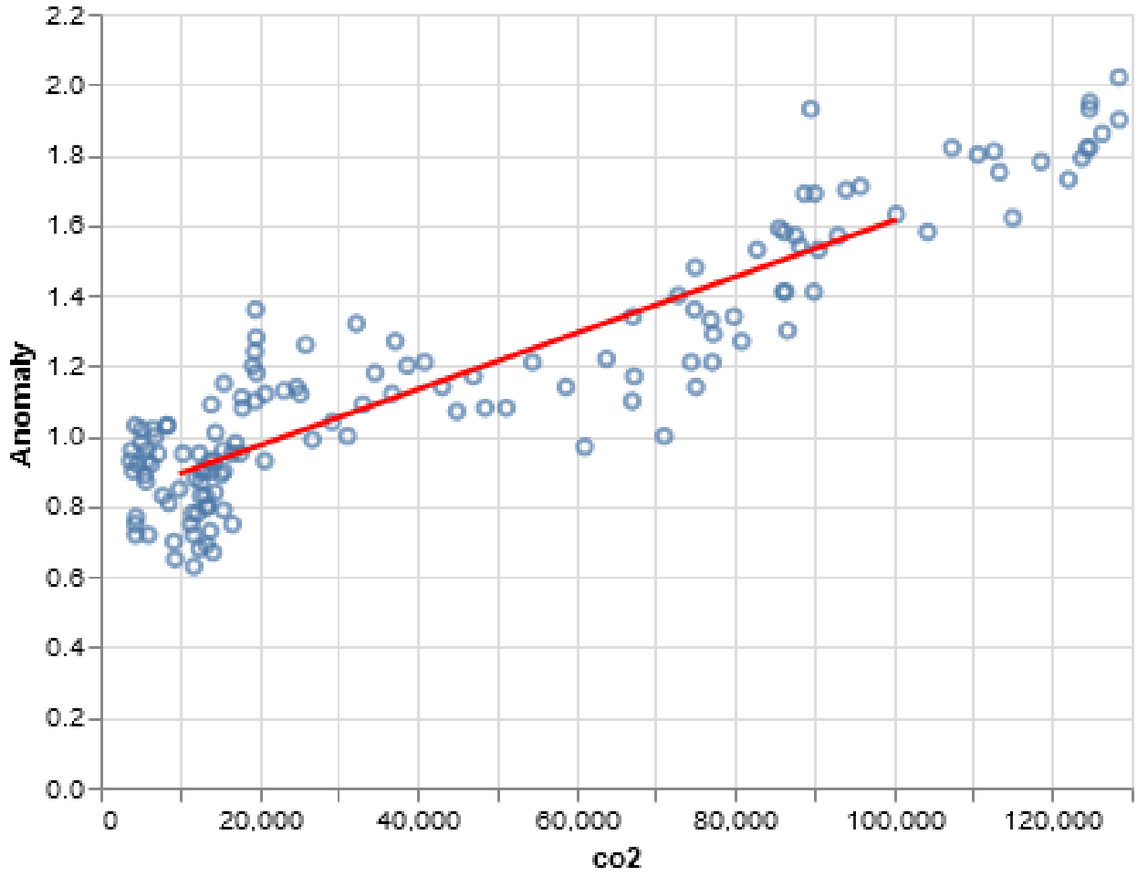
- Scatter plot + regressão linear

```
{
  "$schema": "https://vega.github.io/schema/vega-lite/v5.json",
  "data": {"url": "data/seattle-weather.csv"},
  "layer": [
    { "mark": "point",
      "encoding": {
        "x": {"field": "wind","type": "quantitative"},
        "y": {"field": "precipitation", "type": "quantitative"
    } } }
    ,
    { "mark": "line",
      "transform": [{"regression": "precipitation", "on":"wind"}],
      "encoding": {
        "x": {"field": "wind", "type": "quantitative"},
        "y": {"field": "precipitation","type": "quantitative"},
        "color":{"value":"red"}
    } }
  ]
}
```



Atividade 9.2 (5 min)

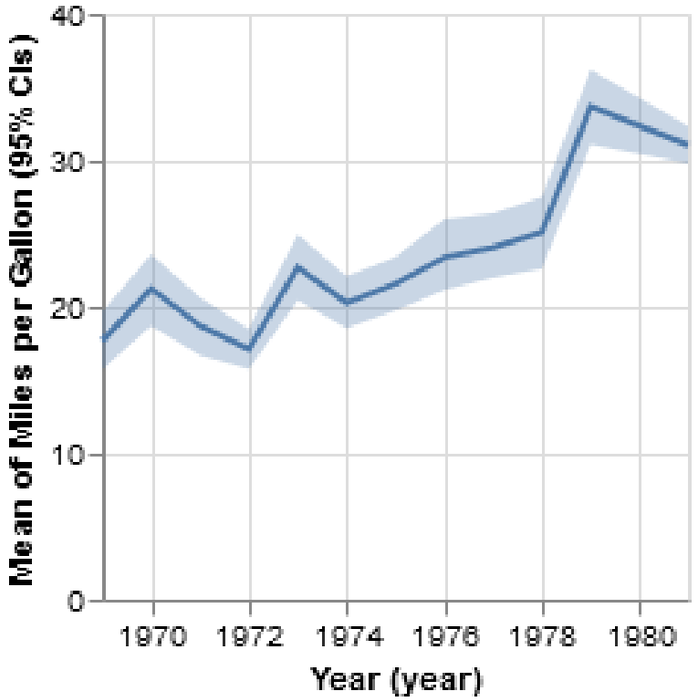
- Scatter Plot + Regressão linear dos dados de
  - Anomalia de Temperatura e Emissão de CO2



Intervalo de Confiança e Desvio Padrão

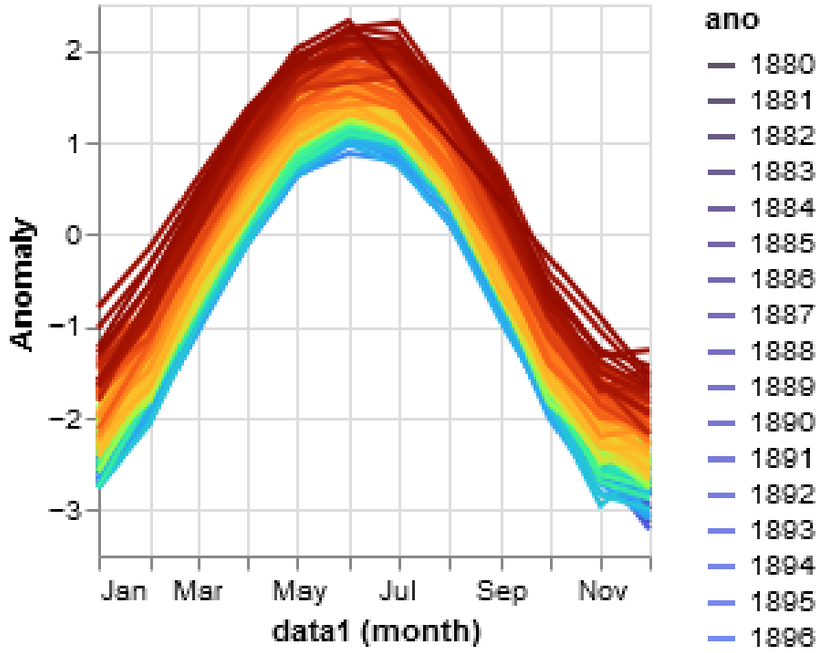
- Plotar layer com o intervalo de confiança (ci),
  - Erro padrão (stderr) ou o desvio padrão (stddev)

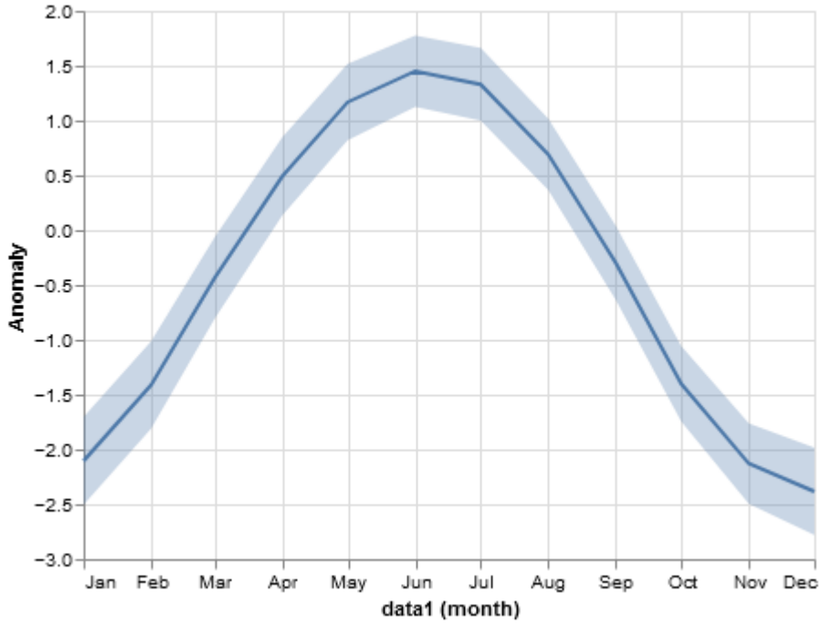
```
{ "$schema": "https://vega.github.io/schema/vega-lite/v5.json",
  "data": {"url": "data/cars.json"},
  "encoding": {
    "x": {"field": "Year", "timeUnit": "year"}
  },
  "layer": [
    { "mark": {"type": "errorband", "extent": "ci"},
      "encoding": {
        "y": {"field": "Miles_per_Gallon", "type": "quantitative", "title":
"Consumo (95% CIs)"}
      }
    },
    { "mark": "line",
      "encoding": {
        "y": {"aggregate": "mean","field": "Miles_per_Gallon"}
      }
    }
  ]
}
```



Atividade 9.3 (5 min)

- Altere o gráfico de linha de anomalia de temperatura separado por anos
  - Para um gráfico com a média e o desvio padrão

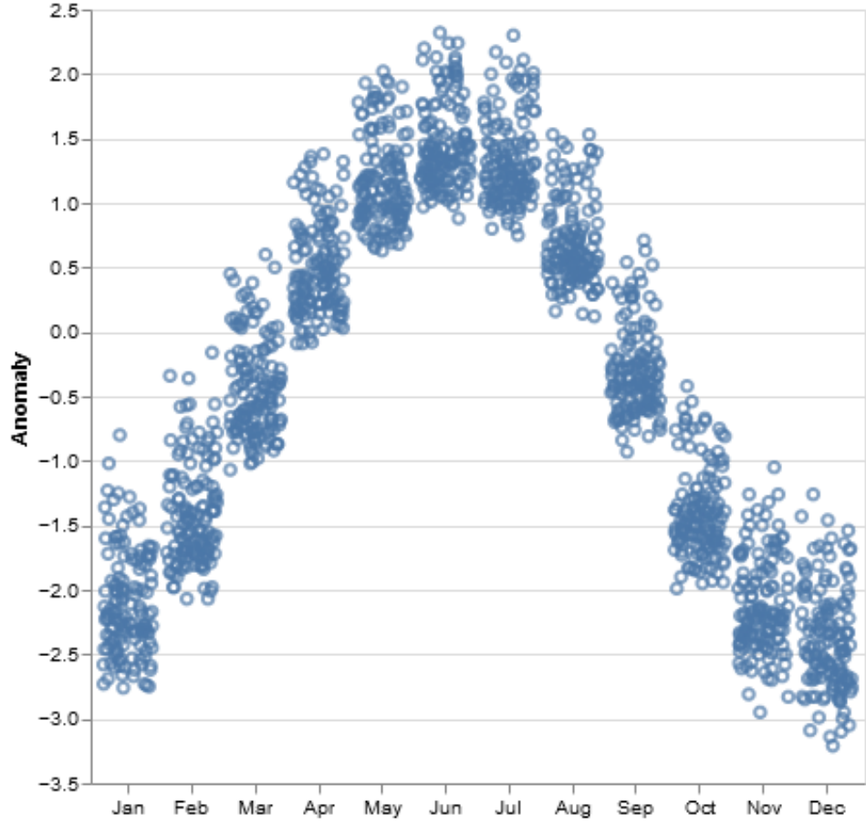




Plotar um Jitter Plot da Anomalia de Temperatura

- x: Quantidade de cilindros de um motor de carro
  - y: Potência do motor

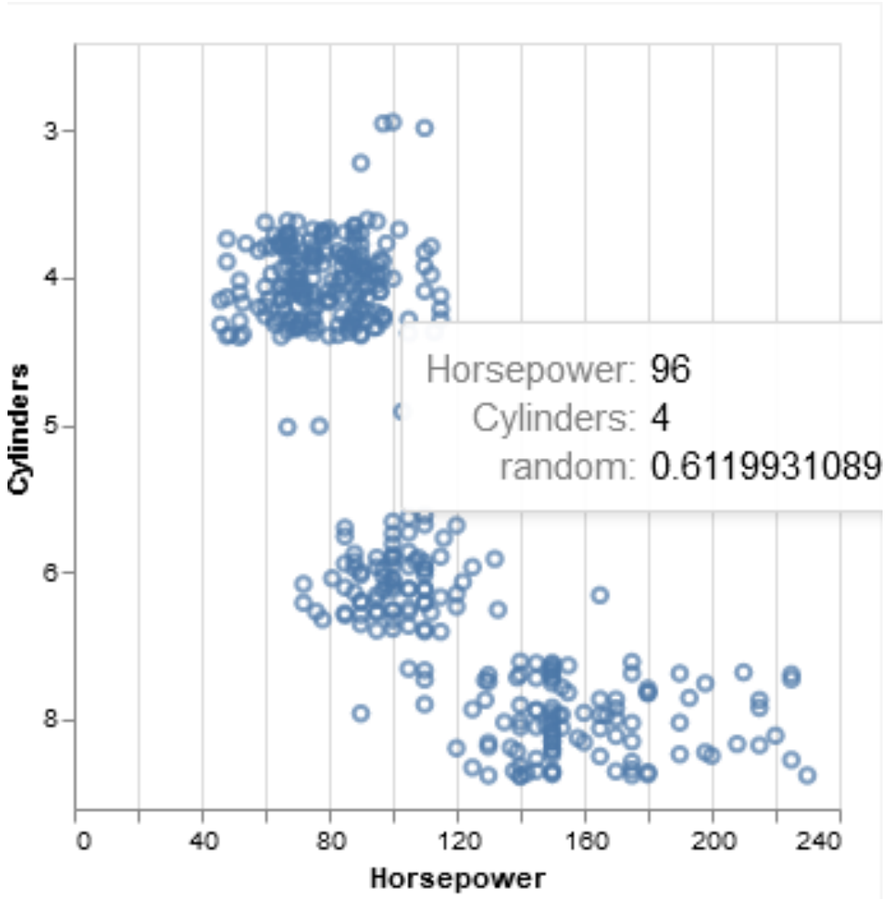
```
{
  "$schema": "https://vega.github.io/schema/vega-
lite/v5.json",
  "data": {"url": "https://...", "format": {"type": "csv"}},
  "transform": [{"calculate": "random()", "as": "random"}],
  "height": 400, "width": 400,
  "mark": "point",
  "encoding": {
    "x": {"field": "data1", "timeUnit":"month", "type":
"ordinal"},
    "y": {"field": "Anomaly", "type": "quantitative"},
    "xOffset": {"field": "random", "type": "quantitative"}
  }
}
```



Plotar um Jitter Plot do dataset Cars

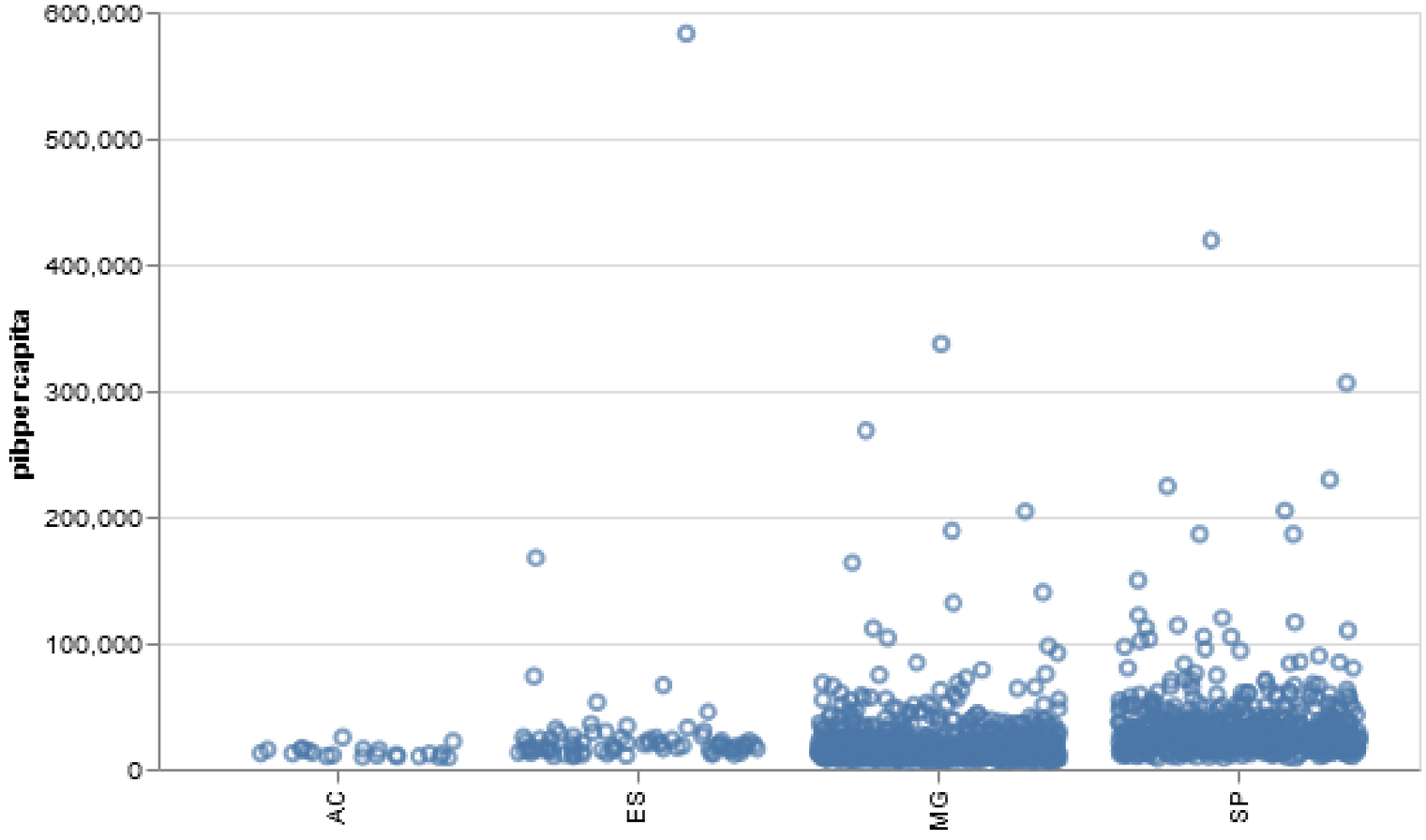
- x: Quantidade de cilindros de um motor de carro
  - y: Potência do motor

```
{
  "$schema": "https://vega.github.io/schema/vega-
lite/v5.json",
  "data": {"url": "data/cars.json"},
  "transform": [{"calculate": "random()", "as": "random"}],
  "height": 300, "width": 300,
  "mark": {"type":"point", "tooltip": true},
  "encoding": {
    "x": {"field": "Horsepower", "type": "quantitative"},
    "y": {"field": "Cylinders", "type": "ordinal"},
    "yOffset": {"field": "random", "type": "quantitative"}
  }
}
```



Atividade 9.4 (5 min)

- Jitter Plot do PIB per capita dos municípios
  - Use este filtro
    - ✓ `{"filter": {"oneOf":["SP","AC","ES","MG"], "field": "sigla_uf"}}`



Integridade de Dados

- Integridade de dados
  - refere-se à acurácia e consistência dos dados ao longo de seu ciclo de vida.
  - é um aspecto crítico para o design, implementação e uso de qualquer sistema que armazena, processa ou recupera dados
- Dado é uma representação da verdade objetiva
  - à posteriori
- Então, é importante
  - Produzir, armazenar e recuperar os dados
    - ✓ Com **garantia** de integridade (consistência e acurácia)
- A ciência da computação já resolveu esse problema!
  - Basta aprender e usar

Fonte

ACID: Quatro qualidades de uma transação

- Essas qualidades contribuem para a capacidade
  - de uma transação de **garantir** a integridade dos dados.
- Atomicidade
  - uma transação deve exibir um comportamento de "tudo ou nada". Todas as instruções acontecem com sucesso ou nenhuma delas acontece. A atomicidade preserva a integridade do processo de negócios.
- **Consistência**
  - **garante** que uma transação só pode trazer o banco de dados de um estado válido para outro,
  - quaisquer dados gravados no banco de dados devem ser válidos de acordo com todas as regras definidas,
    - ✓ restrições, cascatas, gatilhos e qualquer combinação deles.
      - A integridade referencial **garante** a relação chave primária - chave estrangeira.
- Isolamento
  - significa que as transações podem ser executadas ao mesmo tempo.
- Durabilidade
  - refere-se ao impacto de uma interrupção ou falha em uma transação em execução.
  - Uma transação durável não afetará o estado dos dados se a transação terminar de forma anormal.
    - ✓ Em outras palavras, os dados sobrevivem a quaisquer falhas.

Fonte

Consistência: Restrições de Integridade (1)

- As restrições de integridade
  - mecanismo para **garantir** que os dados estejam em conformidade com as diretrizes especificadas pelo administrador do banco de dados.
- **A integridade de entidade**
  - relaciona-se ao conceito de chave primária.
    - ✓ estabelece que cada tabela deve ter uma chave primária e que a coluna ou colunas escolhidas para ser a chave primária devem ser únicas e não nulas.
- **A integridade referencial**
  - uma chave estrangeira deve ter uma chave primária correspondente ou deve ser nula.
  - é especificada entre duas tabelas (pai e filho);
    - ✓ mantém a correspondência entre as linhas nessas tabelas. Isso significa que a referência de uma linha em uma tabela para outra tabela deve ser válida.

Fonte

Consistência: Restrições de Integridade (2)

- **A integridade do domínio**
  - todas as colunas em um banco de dados relacional devem ser declaradas em um domínio definido.
  - Um domínio é um conjunto de valores do mesmo tipo.
    - ✓ conjunto padronizado de valores que os dados de uma coluna podem assumir.
- **Integridade definida pelo usuário (Regras de Negócio)**
  - refere-se a um conjunto de regras especificadas por um usuário,
    - ✓ que não pertencem às categorias entidade, domínio e integridade referencial.
      - Exemplo 1: Numa Carta Convite (modalidade de licitação) deve ter pelo menos 3 participantes.
      - Exemplo 2: Só se pode tomar a 2ª dose de uma vacina depois de tomar a 1ª.

Fonte

Restrições de Integridade: Exemplos problemáticos

- A integridade de entidade
  - Uma tabela de Excel com muitos registros e várias colunas identificadoras
    - ✓ Fica difícil lembrar que já existe outro registro com a mesma combinação de chaves
      - É comum ver registros repetidos
- A integridade referencial
  - Numa base de dados montada numa tabela de Excel
    - ✓ Cadastrar um filho sem o respectivo pai
      - Associar um filho à um pai parecido (Empresa, PJ, Pessoa Jurídica)
    - ✓ Deletar o pai, mas não o filho
- A integridade do domínio
  - Não, Nao, NÃO, NAO, não, nao, Sim, SIM, sim
  - 2021-01-19, 19/02/21, 19/02/2021, 02/19/21, 19/fev/21
  - RG, Identidade, Carteira de Identidade, Título, Título Eleitoral, Título de Eleitor
- Integridade definida pelo usuário (Regras de Negócio)
  - Os mais diversos tipos de transações não permitidas/desejadas.
- Como produzir análises estatísticas neste contexto ?
  - "Não se gerencia o que não se mede; não se mede o que não se define; não se define o que não se entende; não há sucesso no que não se gerencia" (W. Deming)

Validação de Esquema e Verificação de Integridade no Pandas

- Motivação
  - Alguém está produzindo dados em planilha e você precisa atestar a validade dos dados produzidos anteriormente, e
  - Recomendar a verificação das restrições de integridade concomitantemente com a produção dos dados
    - ✓ prevenindo a produção de dados inconsistentes
- Solução intermediária
  - Baixo custo de mudança,
  - Atesta as restrições que se quiser testar.

Pandera – Dataframe Schemas

```python
import pandera as pa

from pandera import Column, DataFrameSchema, Check, Index

schema = DataFrameSchema(
    {
        "column1": Column(int),
        "column2": Column(float, Check(lambda s: s < -1.2)),
        # you can provide a list of validators
        "column3": Column(str, [
           Check(lambda s: s.str.startswith("value")),
           Check(lambda s: s.str.split("_", expand=True).shape[1] == 2)
        ]),
    },
    index=Index(int),
    strict=True,
    coerce=True,
)
```

Pandera – Dataframe Schemas

```python
import numpy as np
import pandas as pd
import pandera as pa

from pandera import Check, Column, DataFrameSchema

df = pd.DataFrame({"column1": [5, 1, np.nan]})

non_null_schema = DataFrameSchema({
    "column1": Column(float, Check(lambda x: x > 0))
})

non_null_schema.validate(df)
```

```
Traceback (most recent call last):
...
SchemaError: non-nullable series contains null values: {2: nan}
```

Pandera – Coluna indispensável (padrão)

```python
df = pd.DataFrame({"column2": ["hello", "pandera"]})
schema = DataFrameSchema({
    "column1": Column(int, required=False),
    "column2": Column(str)
})

validated_df = schema.validate(df)
print(validated_df)
```

```
   column2
0    hello
1  pandera
```

```python
schema = DataFrameSchema({
    "column1": Column(int),
    "column2": Column(str),
})

schema.validate(df)
```

```
Traceback (most recent call last):
...
pandera.SchemaError: column 'column1' not in dataframe
   column2
0    hello
1  pandera
```

Pandera – Regras de checagem

```python
import numpy as np
import pandas as pd
import pandera as pa

categories = ["A", "B", "C"]

np.random.seed(100)

dataframe = pd.DataFrame({
    "cat_var_1": np.random.choice(categories, size=100),
    "cat_var_2": np.random.choice(categories, size=100),
    "num_var_1": np.random.uniform(0, 10, size=100),
    "num_var_2": np.random.uniform(20, 30, size=100),
})
```

	cat_var_1	cat_var_2	num_var_1	num_var_2
0	A	A	6.804147	24.743304
1	A	C	3.684308	22.774633
2	A	C	5.911288	28.416588
3	C	A	4.790627	21.951250
4	C	B	4.504166	28.563142

Pandera – Unicidade conjunta de colunas

```python
schema = pa.DataFrameSchema(
    columns={col: pa.Column(int) for col in ["a", "b", "c"]},
    unique=["a", "c"],
)
df = pd.DataFrame.from_records([
    {"a": 1, "b": 2, "c": 3},
    {"a": 1, "b": 2, "c": 3},
])
schema.validate(df)
```

	a	b	c
0	1	2	3
1	1	2	3

```
SchemaError: columns '('a', 'c')' not unique:
  column  index  failure_case
0      a      0             1
1      a      1             1
2      c      0             3
3      c      1             3
```

Atividade 9.5

- Aplique a validação de esquema usando o pandera ao seu modelo de dados
  - Que contem os dados de população, pib, centroid, bandeira e consumo de energia
  - Crie uma restrição de cada tipo:
    - ✓ Checar o tipo de cada coluna
    - ✓ Checar o conteúdo da coluna lat_long com uma regex
    - ✓ Checar o conteúdo da coluna de consumo de energia com uma função lambda
    - ✓ Checar se a coluna sigla_uf contém apenas as 27 Ufs
    - ✓ Checar se não há registros repetidos com a mesma chave

Atividade 9.6

- Trace um Ranged Dot Plot do consumo de energia elétrica
  - de 4 Estados da federação
- Recomendações:
  - Crie uma versão resumida do dataset que você construiu na aula 8,
  - Use este exemplo como ponto de partida.