

## Aula 4 - Visualização de dados e SQL

Jayme Anchante

25 de fevereiro de 2021

metabase

# por que metabase?

Competidores como

- ▶ Power BI
- ▶ Tableau
- ▶ Qlik
- ▶ Várias outras ferramentas

Vantagens:

- ▶ Código aberto, evolução e comunidade
- ▶ Auto atendimento para não especialistas
- ▶ Governança de acessos
- ▶ Simplicidade, portabilidade, baixo custo de entrada/saída

# instalação

1. Instalar java na sua máquina [acessando este link](#)
2. Baixar metabase no [site oficial](#)

Abrir o explorador de arquivos, navegar até a pasta de downloads, abrir uma linha de comando no diretório atual, e rodar `java -jar metabase.jar`. Se tudo der certo, abrir o navegador na url `localhost:3000` e seguir o passo a passo

# apresentação da ferramenta

- ▶ Barra superior
- ▶ Raio x
- ▶ Painéis
- ▶ Dados

## dados

- ▶ People: id, address, email, password, name, city, longitude, state, source, birth\_date, zip, latitude, created\_at
- ▶ Orders: id, user\_id, product\_id, subtotal, tax, total, discount, created\_at, quantity
- ▶ Products: id, ean, title, category, vendor, price, rating, created\_at
- ▶ Reviews: id, product\_id, reviewer, rating, body

## fazendo uma pergunta

- ▶ Pergunta simples
- ▶ Pergunta customizada
- ▶ Consulta nativa

sql

Structured Query Language

Imperativo



## anatomia do sql

SELECT - SELECIONE o que eu quero FROM - DA ONDE eu quero WHERE - EM QUE certas cláusulas sejam atendidas

## seleção de dados

```
SELECT name  
FROM people
```

- ▶ SELECT name
- ▶ SELECT name, password
- ▶ SELECT name, password AS pass
- ▶ SELECT \*

## filtros

```
SELECT *  
FROM people  
--WHERE
```

- ▶ WHERE birth\_date >= '1999-01-01'
- ▶ WHERE ... AND longitude > -90
- ▶ WHERE password IS NOT NULL OR email IS NOT NULL

## junção de dados

- ▶ Ver as informações de pedidos que os usuários fizeram

```
SELECT *  
FROM people  
INNER JOIN orders  
ON people.id = orders.user_id
```

- ▶ Ver as informações de produtos dos pedidos que os usuários fizeram

```
-- anterior +  
INNER JOIN products  
ON orders.product_id = product.id
```

## agregação de dados

- ▶ Quantos pedidos fez cada usuário:

```
SELECT people.id, COUNT(*)  
FROM people  
INNER JOIN orders  
ON people.id = orders.user_id  
GROUP BY people.id
```

- ▶ Qual o ticket total e médio de cada usuário:

```
SELECT people.id, SUM(orders.total) AS ticketTotal,  
              AVG(orders.total) AS ticketMedio  
FROM people  
INNER JOIN orders  
ON people.id = orders.user_id  
GROUP BY people.id
```

# funções

- ▶ Miríade de funções
- ▶ Depende do banco de dados (PostgreSQL, MariaDB, SQLite etc)
- ▶ É possível definir funções customizadas

```
-- sqlite
```

```
SELECT EXTRACT(YEAR FROM birth_date) AS ano, COUNT(*) AS qt  
FROM people  
GROUP BY birth_date
```

## ordenação

```
-- sqlite
SELECT EXTRACT(YEAR FROM birth_date) AS ano, COUNT(*) AS qt
FROM people
GROUP BY birth_date
ORDER BY 1 ASC
```

## tipo de gráficos

- ▶ Linha
- ▶ Barra
- ▶ Mapa
- ▶ Funil
- ▶ Olhar exemplos do raio-x



## exercícios

1. Selecione os reviews feitos após 31 de dezembro de 2019.
2. Quem foi a pessoa que fez mais reviews?
3. Qual a categoria de produto que recebeu os melhores reviews?
4. Qual a correlação entre o preço de um produto e os reviews recebidos por ele?
5. Selecione os reviews do produto mais barato e do mais caro.

python: conexão com banco de dados

# sqlite3

- ▶ Banco de dados mais utilizado no mundo (smartphones)
- ▶ Extremamente leve e eficiente
- ▶ Pode rodar em disco ou na memória RAM
- ▶ Empacotado junto com Python

> Fonte: [site oficial do SQLite](#)

# base de dados exemplo

Base chamada chinook no formato sqlite disponibilizado pelo site [sqlitetutorial](http://sqlitetutorial.com)



## conexão com pandas

```
1  import sqlite3
2  import pandas as pd
3  conn = sqlite3.connect("chinook.db")
4  # define sql query as string
5  pd.read_csv(sql, conn)
```

## exercícios

1. Qual o nome e sobrenome dos clientes que mais compraram músicas?
2. Qual os gêneros de música que mais venderam?
3. Um músico famoso está prestes a lançar um novo hit e contratou a sua consultoria para definir qual deve ser o preço final da faixa.

A música é do gênero Metal, tem MediaTypeId igual a 3, tem duração de 1min42s e tem 7613817 bytes

matplotlib

## referência

Material baseado o capítulo 4 do livro [Python Data Science Handbook](#) por Jake VanderPlas



## instalação e import

```
# no terminal  
pip install matplotlib
```

```
1 import matplotlib.pyplot as plt  
2 %matplotlib inline # se estiver no jupyter  
3 plt.style.use('seaborn-whitegrid') # sugestão
```

## figura e eixos

Figura contém todos os elementos, sejam eixos, gráficos, texto, rótulos

Eixos são o que vemos quando rodamos o trecho abaixo (caixa com rótulos, grid etc)

```
1 fig = plt.figure()  
2 ax = plt.axes()
```

## plotando dados

Podemos adicionar dados no objeto eixos

```
1 x = np.linspace(0, 10, 1000)
2 ax.plot(x, np.sin(x));
```

## forma simplificada

Também podemos utilizar a interface do pylab e permitir que a figura e o eixo sejam criados para nós automaticamente

```
1 plt.plot(x, np.sin(x));
```

## múltiplas visualizações na mesma figura

```
1 plt.plot(x, np.sin(x));  
2 plt.plot(x, np.cos(x));
```

## estilos

- ▶ color: blue, g, 0.75 (escala de cinza), #FFFFFF (HEX de frontend), (1, 0.2, 0.3) (RGB), chartreuse (nomes em HTML)
- ▶ linestyle: solid (-), dashed (--), dashdot (-.), dotted (.)

```
1 plt.plot(x, np.sin(x - 0), color=color, linestyle=linestyle)
```

## limites dos eixos

```
1 plt.plot(x, np.sin(x))  
2  
3 plt.xlim(-1, 11)  
4 plt.ylim(-1.5, 1.5);
```

## rótulos

```
1 plt.plot(x, np.sin(x), '-g', label='sin(x)')
2 plt.plot(x, np.cos(x), ':b', label='cos(x)')
3 plt.axis('equal')
4
5 plt.title("Sine and Cosine Curves")
6 plt.xlabel("x")
7 plt.ylabel("sin(x)/cos(x)");
8 plt.legend();
```



## gráficos de dispersão

```
1 x = np.linspace(0, 10, 30)
2 y = np.sin(x)
3
4 plt.plot(x, y, 'o', color='black');
```

## exemplo completo

```
1  from sklearn.datasets import load_iris
2  iris = load_iris()
3  features = iris.data.T
4
5  plt.scatter(features[0], features[1], alpha=0.2,
6              s=100*features[3], c=iris.target, cmap='viridis')
7  plt.xlabel(iris.feature_names[0])
8  plt.ylabel(iris.feature_names[1]);
```

# histogramas

```
1 data = np.random.randn(1000)
2 plt.hist(data);
```

## múltiplos subgráficos

```
1  fig = plt.figure()
2  ax1 = fig.add_axes([0.1, 0.5, 0.8, 0.4],
3                      xticklabels=[], ylim=(-1.2, 1.2))
4  ax2 = fig.add_axes([0.1, 0.1, 0.8, 0.4],
5                      ylim=(-1.2, 1.2))
6
7  x = np.linspace(0, 10)
8  ax1.plot(np.sin(x))
9  ax2.plot(np.cos(x));
```

texto

```
1  fig, ax = plt.subplots(facecolor='lightgray')
2  ax.axis([0, 10, 0, 10])
3
4  # transform=ax.transData is the default, but we'll specify
5  ax.text(1, 5, ". Data: (1, 5)", transform=ax.transData)
6  ax.text(0.5, 0.1, ". Axes: (0.5, 0.1)", transform=ax.transA
7  ax.text(0.2, 0.2, ". Figure: (0.2, 0.2)", transform=fig.tra
```

## exercícios

- ```
1 import seaborn as sns
2 planets = sns.load_dataset('planets')
```
1. Plote a distribuição de massa dos planetas.
  2. Plote a distribuição de massa dos planetas por métodos, para os métodos Radial Velocity e Transit.
  3. Plote a dispersão de massa e distância. Você nota alguma correlação entre essas duas variáveis?
  4. Plote a quantidade de planetas descobertos por ano.

projeto final

## projeto final

- ▶ Um fenômeno/problema do interesse de vocês
- ▶ Pode ser respondido com dados
- ▶ Não seja extremamente complexo
- ▶ Vocês devem ser os donos do projeto de ponta a ponta



## formato

- ▶ 15min de apresentação
- ▶ 5min de perguntas
- ▶ 10min de respostas

# dados

Caso tenham dificuldades em encontrar datasets, aqui vão algumas sugestões:

- ▶ UCI
- ▶ Kaggle
- ▶ Google Datasets Search
- ▶ Dados do IBGE, como a [PNAD](#) e o [censo](#)

# template

O template indicado pode ser [encontrado aqui](#)