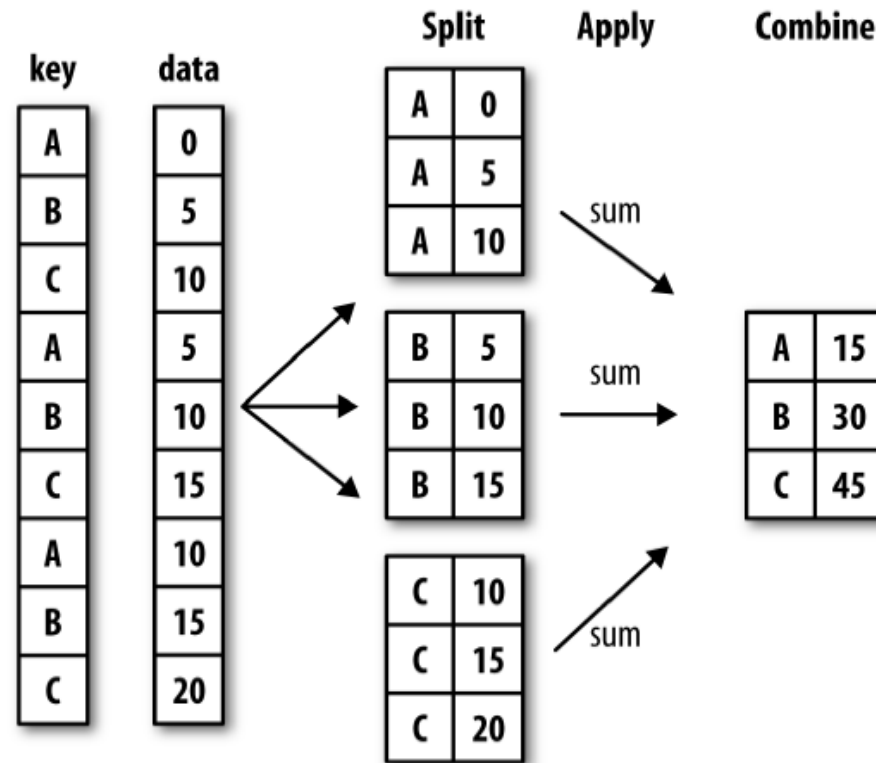


Agregação com Pandas



Podcast sobre a origem da comunicação digital

- <https://podcasts.apple.com/br/podcast/cautionary-tales-with-tim-harford/id1484511465?l=en&i=1000521668789>

Prompt para formatar uma lista de telefones

- Formate os números de telefone na planilha original Pasta1.xlsx. A saída deve estar em um dos seguintes formatos. Use esta sintaxe número 1: """"+YY (XX) TTTT-XXXX""", ou esta sintaxe número 2: """"+ZZ (XXX) XXXX-XXXX""". Se o número +YY estiver ausente, assumo que é +55. Se o número +ZZ estiver ausente, assumo que é +1. Se houver dois números de telefone na mesma linha separados por "/" ou outro separador, como ",", ou ";", divida a célula e coloque o segundo número de telefone em uma nova coluna na mesma linha do original. Exporte o resultado em um arquivo.xlsx com duas colunas. Na segunda coluna deve ser colocado um dos números de telefone das linhas que têm dois números de telefone. Mantenha as linhas em branco no arquivo de saída. Considere os seguintes exemplos de mapeamentos de conversão de um prompt few shot:

Prompt para formatar uma lista de telefones (continuação)

""""

(21) 9623-9999 => +55 (21) 99623-9999

(12) 963-9999 => +55 (12) 99963-9999

(12) 9 9791-9999 => +55 (12) 99791-9999

(21) 8141-9999 => +55 (21) 98141-9999

9776-9999 => +55 (12) 99776-9999

(51) 9268-9999 => +55 (51) 9268-9999

(21) 9961 9999 => +55 (21) 99961 9999

(11) 97382-9999 => +55 (11) 97382-9999

(21) 99779999 => +55 (21) 99977-9999

(19) 9113-9999 / 99150-1071 => +55 (19) 99113-
1566 in column A, and +55 (19) 99150-1071 in
column B.

(11) 9 9829-9999 => +55 (11) 99829-9999

(21) 9959-9999 => +55 (21) 99959-9999

(11) 998-9999 => +55 (11) 99998-9999

(17)97149999 => +55 (17) 99714-9999

21 2568-9999 => +55 (21) 92568-9999

3234 9999 => +55 (12) 93234 9999

""""

Curadoria de Prompts

- Dataset de Prompts

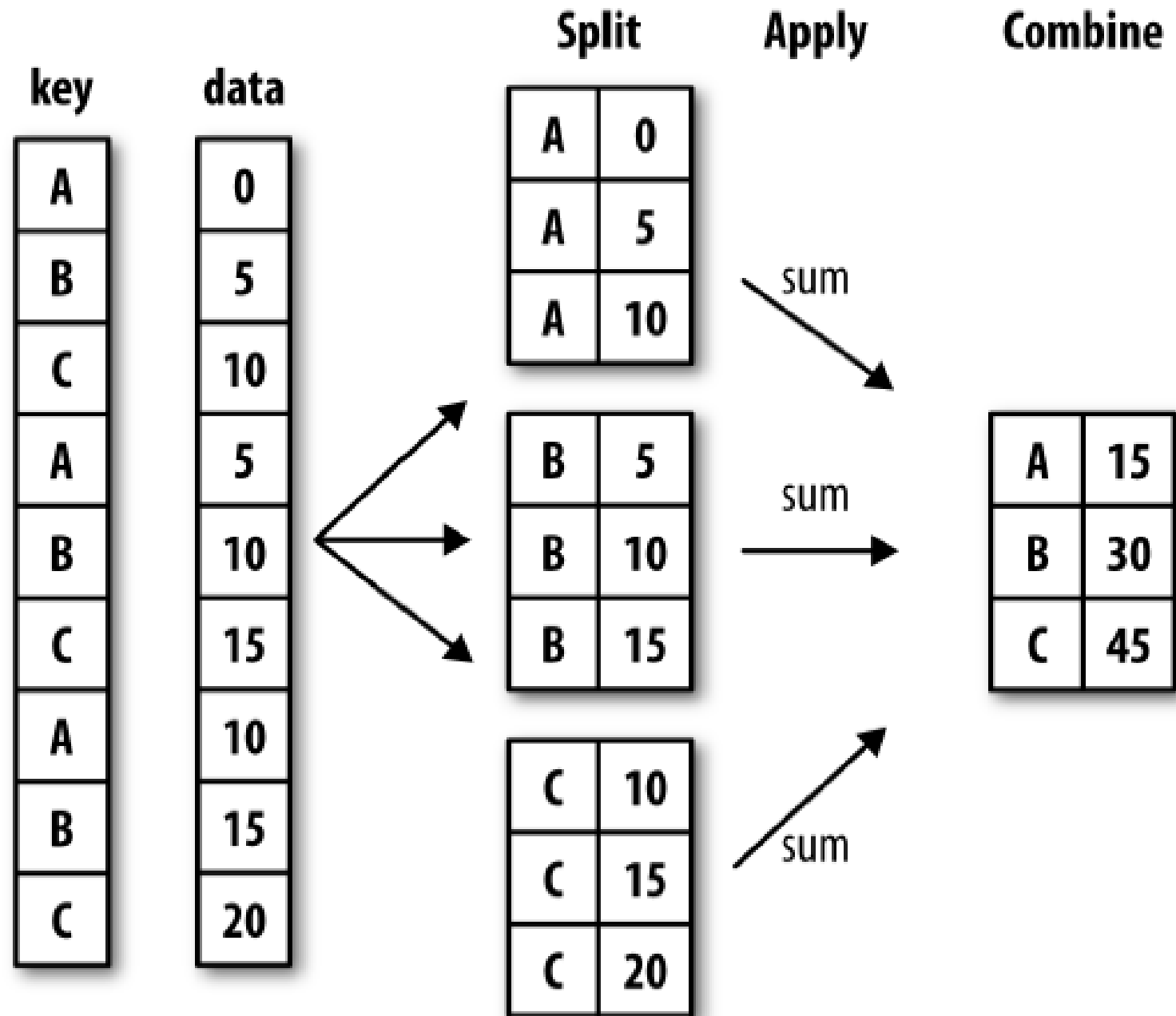
Exemplo de Agregação

OrderDetailID	OrderID	ProductID	Quantity
1	10248	1	2
2	10248	2	10
3	10248	7	5
4	10249	4	5
5	10249	1	4
6	10250	2	5
7	10250	1	6
8	10250	4	15

GROUP BY ProductID
Somando a coluna
Quantity;

ProductID	Qty
1	12
2	15
4	20
7	5

split-apply-combine (por Hadley Wickham)



Exemplo de Group By com Pandas

```
In [10]: df = pd.DataFrame({'key1' : ['a', 'a', 'b', 'b', 'a'],  
.....:                   'key2' : ['one', 'two', 'one', 'two', 'one'],  
.....:                   'data1' : np.random.randn(5),  
.....:                   'data2' : np.random.randn(5)})
```

```
In [11]: df
```

```
Out[11]:
```

	data1	data2	key1	key2
0	-0.204708	1.393406	a	one
1	0.478943	0.092908	a	two
2	-0.519439	0.281746	b	one
3	-0.555730	0.769023	b	two
4	1.965781	1.246435	a	one

```
In [12]: grouped = df['data1'].groupby(df['key1'])
```

```
In [13]: grouped
```

```
Out[13]: <pandas.core.groupby.SeriesGroupBy object at 0x7faa31537390>
```

```
In [14]: grouped.mean()
```

```
Out[14]:
```

```
key1
```

```
a      0.746672
```

```
b     -0.537585
```

```
Name: data1, dtype: float64
```

Agregação com duas colunas

```
In [15]: means = df['data1'].groupby([df['key1'], df['key2']]).mean()
```

```
In [16]: means
```

```
Out[16]:
```

```
key1  key2
```

```
a      one    0.880536
```

```
      two    0.478943
```

```
b      one   -0.519439
```

```
      two   -0.555730
```

```
Name: data1, dtype: float64
```

Curiosidade

```
In [17]: means.unstack()
```

```
Out[17]:
```

```
key2      one      two
```

```
key1
```

```
a      0.880536  0.478943
```

```
b     -0.519439 -0.555730
```

Analogia com SQL:

Aplicando a métrica em todas as colunas de dados

```
In [21]: df.groupby('key1').mean()
```

```
Out[21]:
```

	data1	data2
key1		
a	0.746672	0.910916
b	-0.537585	0.525384

```
In [22]: df.groupby(['key1', 'key2']).mean()
```

```
Out[22]:
```

		data1	data2
key1	key2		
a	one	0.880536	1.319920
	two	0.478943	0.092908
b	one	-0.519439	0.281746
	two	-0.555730	0.769023

Mostrando o tamanho dos grupos

```
In [23]: df.groupby(['key1', 'key2']).size()
```

```
Out[23]:
```

```
key1  key2
```

```
a      one      2
```

```
      two      1
```

```
b      one      1
```

```
      two      1
```

```
dtype: int64
```

Resumo da Sintaxe do Groupby

- Tipos de Argumentos aceitos pela função groupby
 - String
 - ✓ com o nome da chave de agrupamento
 - A coluna precisa estar dentre as colunas selecionadas
 - Lista de strings
 - ✓ com os nomes das chaves de agrupamento
 - As colunas precisam estar dentre as colunas selecionadas
 - Serie (ou dicionário)
 - ✓ com os dados a serem usados como agrupamento
 - Lista de series (ou dicionários)
 - ✓ com os dados a serem usados como agrupamento
- Erro comum !!!
 - Confundir essas sintaxes

Outras métricas para aplicar na agregação

Função	Descrição
count	Número de valores não NA no grupo
sum	Soma de valores não NA
mean	Média de valores não NA
median	Mediana de valores não NA
std, var	Desvio padrão e variância não enviesada (n-1 no denominador)
min, max	Mínimo e Máximo de valores não NA
prod	Produto de valores não NA
first, last	Primeiro e último valores não NA

Aplicando várias métricas ao mesmo tempo

	total_bill	tip	smoker	day	time	size	tip_pct
0	16.99	1.01	No	Sun	Dinner	2	0.059447
1	10.34	1.66	No	Sun	Dinner	3	0.160542
2	21.01	3.50	No	Sun	Dinner	3	0.166587
3	23.68	3.31	No	Sun	Dinner	2	0.139780
4	24.59	3.61	No	Sun	Dinner	4	0.146808
5	25.29	4.71	No	Sun	Dinner	4	0.186240

```
In [60]: grouped = tips.groupby(['day', 'smoker'])
```

```
In [61]: grouped_pct = grouped['tip_pct']
```

```
In [63]: grouped_pct.agg(['mean', 'std', 'peak_to_peak'])
```

```
Out[63]:
```

		mean	std	peak_to_peak
day	smoker			
Fri	No	0.151650	0.028123	0.067349
	Yes	0.174783	0.051293	0.159925
Sat	No	0.158048	0.039767	0.235193
	Yes	0.147906	0.061375	0.290095
Sun	No	0.160113	0.042347	0.193226
	Yes	0.187250	0.154134	0.644685

Várias métricas em diferentes colunas

```
In [72]: grouped.agg({'tip_pct' : ['min', 'max', 'mean', 'std'],  
.....:               'size' : 'sum'})
```

```
Out[72]:
```

		tip_pct					size
		min	max	mean	std		sum
day	smoker						
Fri	No	0.120385	0.187735	0.151650	0.028123		9
	Yes	0.103555	0.263480	0.174783	0.051293		31
Sat	No	0.056797	0.291990	0.158048	0.039767		115
	Yes	0.035638	0.325733	0.147906	0.061375		104
Sun	No	0.059447	0.252672	0.160113	0.042347		167
	Yes	0.065660	0.710345	0.187250	0.154134		49
Thur	No	0.072961	0.266312	0.160298	0.038774		112
	Yes	0.090014	0.241255	0.163863	0.039389		40

Variáveis do tipo categoria (Categorical)

```
from pandas.api.types import CategoricalDtype
cat_type = CategoricalDtype(categories=["Seg", "Ter", "Qua",
                                       "Qui", "Sex", "Sab", "Dom"], ordered=True)
df = pd.DataFrame({'Bairro1': ['Dom', 'Sab', 'Dom', 'Qui', 'Dom', 'Sab'],
                  'Bairro2': ['Sex', 'Sex', 'Sex', 'Seg', 'Ter', 'Qui']},
                  dtype=cat_type)
```

df

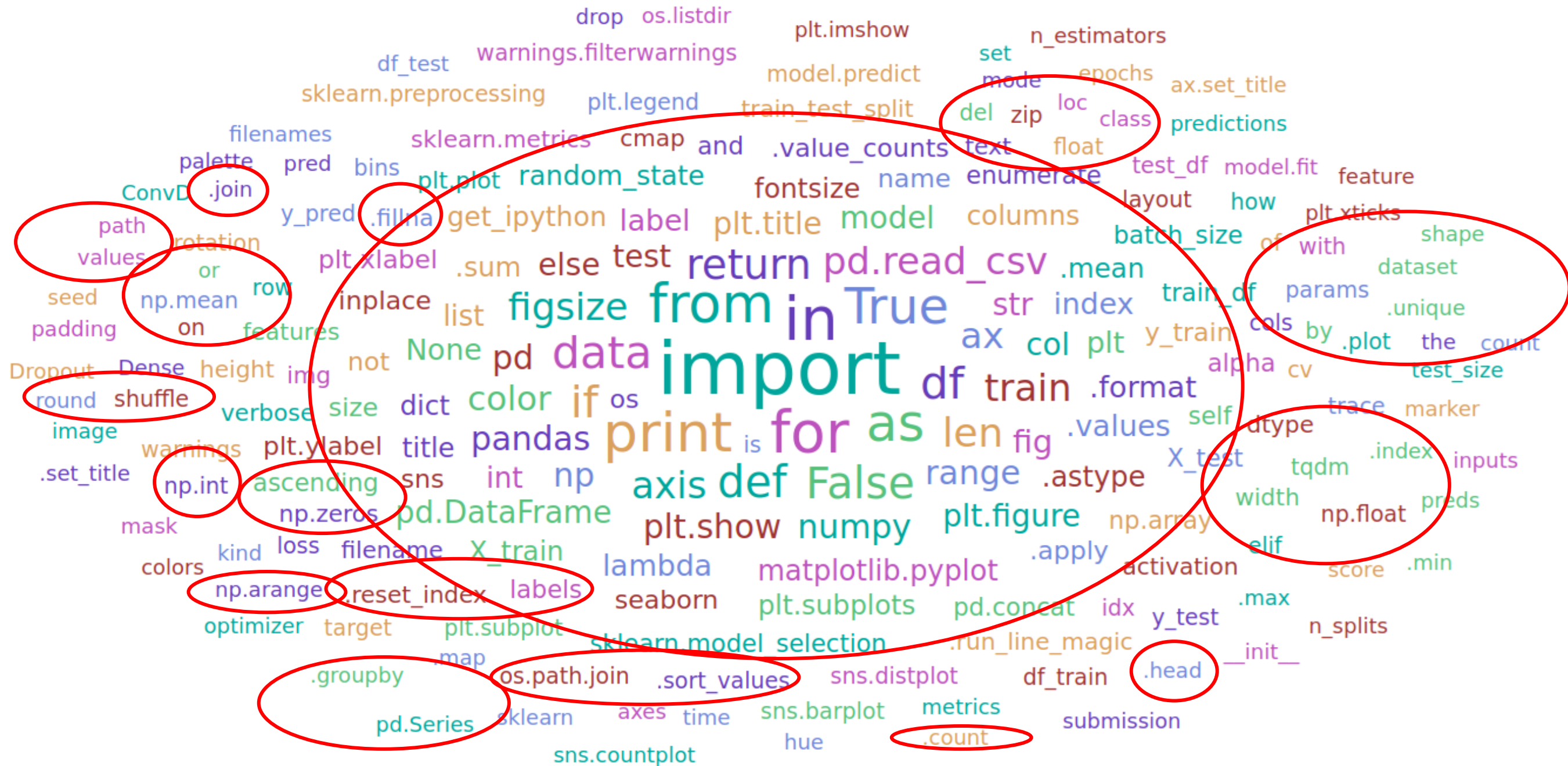
	Bairro1	Bairro2
0	Dom	Sex
1	Sab	Sex
2	Dom	Sex
3	Qui	Seg
4	Dom	Ter
5	Sab	Qui



```
df[df['Bairro1'] > 'Sab'].head()
```

	Bairro1	Bairro2
0	Dom	Sex
2	Dom	Sex
4	Dom	Ter

Roadmap do Vocabulário



Como progredir mais rápido: Desenvolver hábitos e competências para melhorar o desempenho

- Meta-análise: melhorar o seu processo individual de trabalho
 - Refletir sobre o que te levou a cometer cada erro
 - ✓ e como evita-lo da próxima vez (Ciclo de PDCA)
- Preocupar-se com o processo (melhoria contínua) = Aceleração

Competências/hábitos mais importantes

- Aprender iterativamente
 - Com a IA
- Ler mensagens de Log
 - É o caminho mais direto entre você e a solução de um problema
 - ✓ Alguém pensou na mensagem que mais te ajudaria
 - Qualquer coisa diferente disso não é racional.
- Manter-se atualizado sobre qual o melhor modelo de IA
 - Para geração de código

Competências/hábitos mais importantes

- Baby steps (diminuir o tamanho do ciclo de feedback)
 - Quebrar uma tarefa em partes elementares
 - ✓ Que você tenha certeza do resultado;
- Obter feedback visual (print)
 - de cada baby step;
- Numa situação de erro, imprimir o conteúdo (feedback)
 - que te faz entender a causa do erro; e
- Mentalizar (reconhecer/imprimir) o tipo de objeto
 - retornado por cada instrução.

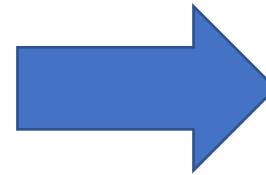
Competências/hábitos mais importantes

- Inferir quando a IA gerou um código incorreto
 - E você precisa conferir a documentação oficial das bibliotecas
- Ler atentamente o enunciado
 - Ou definir claramente (por escrito) seu problema
 - ✓ Certificar-se que você tem certeza do que precisa fazer antes de começar
- Escrever/desenhar a sua solução,
 - principalmente as etapas que demandam grande capacidade de abstração ou raciocínios longos

Resolvendo Problemas com o Pandas

- Correlação entre a variação percentual de dois ativos financeiros
 - Exemplo: Ação do Walmart e da Amazon

symbol	date	close
AMZN	2021-05-10	3190.489990
	2021-05-11	3223.909912
	2021-05-12	3151.939941
	2021-05-13	3161.469971
	2021-05-14	3222.899902
WMT	2021-05-10	140.820007
	2021-05-11	139.550003
	2021-05-12	135.940002
	2021-05-13	138.240005
	2021-05-14	139.520004



symbol	AMZN	WMT
date		
2021-05-11	0.010475	-0.009019
2021-05-12	-0.022324	-0.025869
2021-05-13	0.003024	0.016919
2021-05-14	0.019431	0.009259

Resolvendo Problemas com o Pandas

- Correlação entre a variação percentual de dois ativos financeiros
 - Exemplo: Ação do Walmart e da Amazon
 - ✓ Estratégia: Diversificar, investindo em ativos não correlacionados

```
df_reset = df.reset_index()
df_reset['ontem'] = df_reset['date'].apply(lambda x: x + datetime.timedelta(days=1))
df_merge = df_reset.merge(right=df_reset, left_on=['symbol', 'date'],
                          right_on=['symbol', 'ontem'], suffixes=["", "_desloc"])
df_merge['change_pct'] = (df_merge['close'] - df_merge['close_desloc']) / df_merge['close_desloc']
df_pivot = df_merge.pivot('date', 'symbol', 'change_pct')
```

symbol	AMZN	WMT
date		
2021-05-11	0.010475	-0.009019
2021-05-12	-0.022324	-0.025869
2021-05-13	0.003024	0.016919
2021-05-14	0.019431	0.009259

df_pivot.corr()		
symbol	AMZN	WMT
symbol		
AMZN	1.000000	0.713742
WMT	0.713742	1.000000

Resolvendo Problemas com o Pandas

- Correlação entre a variação percentual de dois ativos financeiros
 - Exemplo: Ação do Walmart e da Amazon
 - ✓ Estratégia: Diversificar, investindo em ativos não correlacionados

```
df[['close']].unstack(level=0).pct_change()
```

close		
symbol	AMZN	WMT
date		
2021-05-10	NaN	NaN
2021-05-11	0.010475	-0.009019
2021-05-12	-0.022324	-0.025869
2021-05-13	0.003024	0.016919
2021-05-14	0.019431	0.009259

close			
symbol		AMZN	WMT
symbol			
close	AMZN	1.000000	0.713742
	WMT	0.713742	1.000000

Explodir columna de atributos multi-valorados

	actors	director	imdb_title_id	title
0	Blanche Bayliss, William Courtenay, Chauncey D...	Alexander Black	tt0000009	Miss Jerry
1	Elizabeth Tait, John Tait, Norman Campbell, Be...	Charles Tait	tt0000574	The Story of the Kelly Gang
2	Asta Nielsen, Valdemar Psilander, Gunnar Helse...	Urban Gad	tt0001892	Den sorte drøm



	actors	imdb_title_id	director	title
0	Blanche Bayliss	tt0000009	Alexander Black	Miss Jerry
1	William Courtenay	tt0000009	Alexander Black	Miss Jerry
2	Chauncey Depew	tt0000009	Alexander Black	Miss Jerry
3	Elizabeth Tait	tt0000574	Charles Tait	The Story of the Kelly Gang
4	John Tait	tt0000574	Charles Tait	The Story of the Kelly Gang
5	Norman Campbell	tt0000574	Charles Tait	The Story of the Kelly Gang



Apache
Airflow

Orquestrador de workflows e pipelines

Criando DAGs e Tasks

```
from airflow import DAG
with DAG(
    "mlops",
    default_args={
        "retries": 1,
    },
    schedule=timedelta(days=1),
    start_date=datetime(2023, 1, 1)
) as dag:
    ...
```

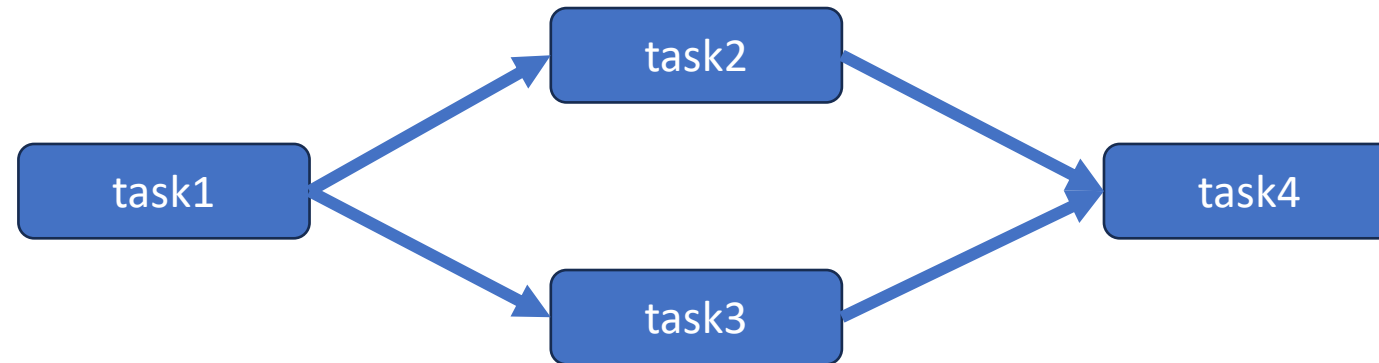
```
...
task1 = BashOperator(
    task_id="print_date",
    bash_command="date",
)

task2 = MySqlOperator(
    task_id="load_table",
    sql="/scripts/load_table.sql"
)
task1 >> task2
```



Tasks paralelas

```
task1 >> [task2, task3] >> task4
```

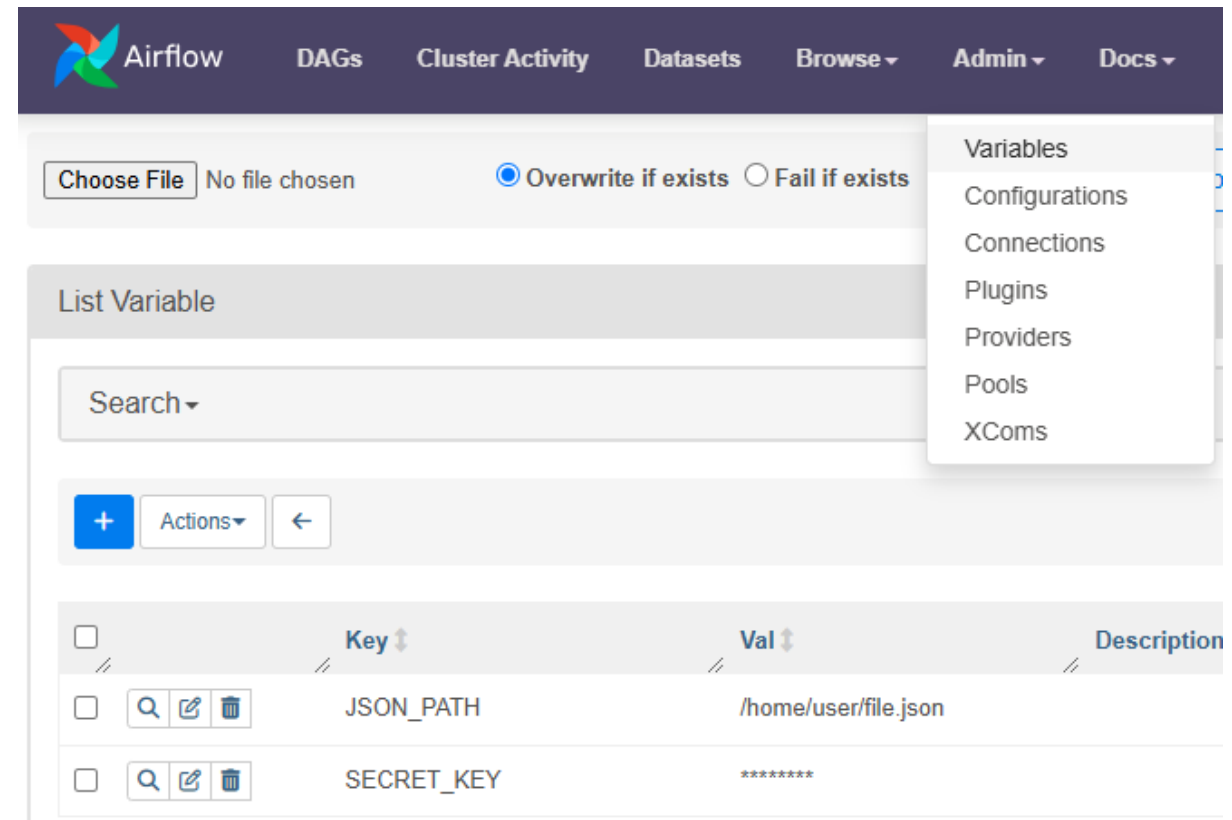


Variáveis de Ambiente no Airflow

- Servem como um armazenamento global
 - de chave-valor
 - ✓ para configurações em tempo de execução.
 - Acessível no menu Admin -> Variables
- Não coloque informações sensíveis
 - Diretamente no código.
 - ✓ Existem muitas soluções pra isso.

```
from airflow.models import Variable

# Recuperar uma variável simples
my_var = Variable.get("my_variable_name")
```



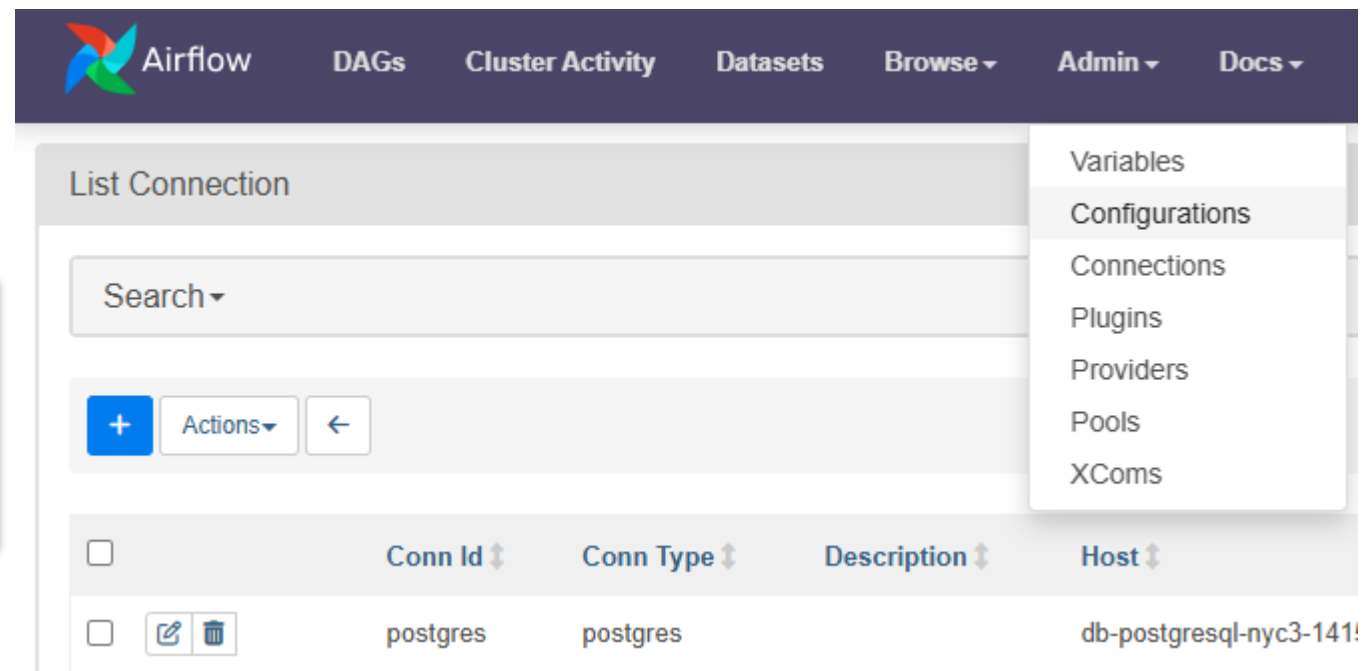
The screenshot shows the Airflow Admin interface. The top navigation bar includes links for Airflow, DAGs, Cluster Activity, Datasets, Browse, Admin, and Docs. The Admin menu is open, showing options like Variables, Configurations, Connections, Plugins, Providers, Pools, and XComs. The 'List Variable' page is displayed, featuring a search bar and a table of variables. The table has columns for a checkbox, Key, Val, and Description. Two variables are listed: JSON_PATH with value /home/user/file.json and SECRET_KEY with value *****.

	Key	Val	Description
<input type="checkbox"/>	JSON_PATH	/home/user/file.json	
<input type="checkbox"/>	SECRET_KEY	*****	

Conexões com Serviços no Airflow

- Armazenam credenciais de autenticação
 - e parâmetros de configuração
 - ✓ necessários para interagir com sistemas externos
- Não coloque informações de conexões
 - Diretamente no código.
 - ✓ Existem muitas soluções pra isso.

```
pg_hook = PostgresHook(postgres_conn_id='postgres')
engine = pg_hook.get_sqlalchemy_engine()
# Save the DataFrame to the database
df.to_sql('openfda_data', con=engine, if_exists='append', index=False)
```



Airflow DAGs Cluster Activity Datasets Browse Admin Docs

List Connection

Search

+ Actions

	Conn Id	Conn Type	Description	Host
<input type="checkbox"/>	postgres	postgres		db-postgresql-nyc3-141!

Exercício 9.5 (Vale 5 pontos)

- Replique o experimento descrito no [tutorial em vídeo](#) do Airflow
- Construa um dashboard com dados diários de 6 meses do preço do bitcoin
- Compartilhe neste [formulário](#)
 - Um print (screenshot) da sua tela mostrando a sua instância do Apache Airflow, as Tasks executadas e o log da execução.
 - A URL do github do código da sua DAG
 - A URL do seu dashboard **público** no Looker Studio

Se entender é porque já está falando a língua dos nerds



Prática no Google Colab

- Faça os exercícios da aula.