



**Fecomércio
Sesc**

Big Data

Prof. Marco Mialaret

Março

2024



Big Data

Onde me encontrar:

<https://www.linkedin.com/in/marco-mialaret-junior/>

e

<https://github.com/MatmJr>

Banco de dados Relacionais no Python

Big Data



Uma **banco de dados** é uma coleção integrada de dados. Um **sistema de gerenciamento de banco de dados (SGBD)** fornece mecanismos para armazenar e organizar dados de maneira consistente com o formato da base de dados.

Os SGBD permitem um acesso conveniente e armazenamento de dados sem preocupação com a representação interna dos banco de dados.

Big Data

Alguns SGBD's open-source, são: **SQLite, PostgreSQL, MariaDB and MySQL.**

Na aula de hoje vamos utilizar o SQLite por sua integração com o python.

Big Data

Vamos acessar a URL do arquivo no Google Drive

url = <https://drive.google.com/uc?id=1QrKwpVdrCfIRxPJSHhT2sffqIHij-mS>

Big Data

No python vamos precisar da biblioteca gdown

```
import gdown

# URL do Google Drive convertida para download direto
url = 'https://drive.google.com/uc?id=1kcH4xji\_A1\_FCgHoDOf6eu9EYcX0blyP'

# Caminho local para salvar o arquivo .db
output_path = 'chinook.db'

# Baixar o arquivo
gdown.download(url, output_path, quiet=False)
```


Big Data

Executando o código anterior será feito o download de um arquivo chamado “chinook.db” que será o banco de dados do primeiro momento da nossa aula.

Para trabalhar com o banco de dados em Python, primeiro use a função `connect` do `sqlite3` para se conectar ao banco de dados e obter um objeto `Connection`:

Big Data

Chinook é um banco de dados de exemplo disponível para SQL Server, Oracle, MySQL, entre outros. Ele pode ser criado executando um único script SQL. O banco de dados Chinook é uma alternativa ao banco de dados Northwind, sendo ideal para demonstrações e testes de ferramentas ORM que visam servidores de banco de dados únicos e múltiplos.

Big Data

Chinook é um banco de dados de exemplo disponível para SQL Server, Oracle, MySQL, entre outros. Ele pode ser criado executando um único script SQL. O banco de dados Chinook é uma alternativa ao banco de dados Northwind, sendo ideal para demonstrações e testes de ferramentas ORM que visam servidores de banco de dados únicos e múltiplos.

SQLite + Pandas

Big Data

Para trabalhar com o banco de dados em Python, primeiro use a função `connect` do `sqlite3` para se conectar ao banco de dados e obter um objeto `Connection`:

```
import sqlite3  
  
connection = sqlite3.connect('chinook.db')
```

Big Data

O pandas oferece um método eficiente para carregar dados de bancos de dados SQL. Geralmente, utilizamos esse método para executar uma consulta SQL, usando uma conexão já estabelecida com o banco de dados. Assim, para visualizar todas as tabelas do banco podemos utilizar a consulta:

```
import pandas as pd

pd.read_sql_query("SELECT * FROM sqlite_master WHERE type='table'", connection)
```

Big Data

O banco de dados possui onze tabelas. Analisando a tabela de 'Album', temos:

```
pd.read_sql("select * from Album", connection)
```

Big Data



Perceba que temos duas colunas de índice. Para resolver isso:

```
pd.read_sql("select * from Album", connection, index_col=["AlbumId"])
```


Big Data

Analizando outra tabela do banco:

```
pd.read_sql("select * from Invoice", connection)
```

Big Data

Vamos criar uma função chamada `sq` para simplificar o nosso trabalho. Com ela não vamos precisar ficar escrevendo `pd.read_sql` a todo momento...

```
def sq(str, con=connection):  
    return pd.read_sql('{}'.format(str), con)
```

Big Data

Frequentemente selecionaremos linhas em um banco de dados que atendem a certos critérios de seleção, especialmente em grandes volumes de dados, onde um banco de dados pode conter muitas linhas. Apenas as linhas que satisfazem os critérios de seleção (formalmente chamados de predicados) são selecionadas. A cláusula WHERE do SQL especifica os critérios de seleção de uma consulta. Valores de string em consultas SQL são delimitados por aspas simples (').

Big Data

```
sq('' select * from Invoice where total > '10' '')
```

Big Data

Complicando um pouco:

```
sq('''select *  
from invoice  
where total < (select avg(total) from invoice)  
''')
```

Big Data

A cláusula WHERE pode conter os operadores <, >, <=, >=, =, <> (diferente) e LIKE. O operador LIKE é usado para correspondência de padrões—procurando por strings que combinam com um padrão dado. Um padrão que contém o caractere curinga de porcentagem (%) procura por strings que tenham zero ou mais caracteres na posição do caractere de porcentagem no padrão. Por exemplo, vamos localizar todos os artistas cujo nome começa com a letra D:

Big Data

```
sq(''select * from Artist where name like 'D%' ''')
```


Big Data

A cláusula ORDER BY ordena os resultados de uma consulta em ordem ascendente (do menor para o maior) ou descendente (do maior para o menor), especificados com ASC e DESC, respectivamente. A ordem de classificação padrão é ascendente, portanto, ASC é opcional. Vamos ordenar os títulos dos álbuns em ordem ascendente:

Big Data

```
sq(''select * from Album order by title asc'')
```

Big Data

Você pode mesclar dados de várias tabelas, o que é referido como juntar as tabelas, com o JOIN.

```
sq(''' SELECT * FROM album  
JOIN artist ON artist.artistid = album.artistid ''')
```

Big Data

Para evitar a duplicação de colunas no seu resultado, você deve especificar explicitamente quais colunas deseja retornar na sua consulta e pode usar aliases para renomeá-las conforme necessário:

```
sq(''' select album.albumid,  
        album.title,  
        artist.artistid AS artist_id,  
        artist.name  
from album  
JOIN artist ON artist.artistid = album.artistid ''')
```

Big Data

Exercício 1: Qual funcionário tem o maior número total de clientes?

```
sq(''' SELECT
    e.FirstName || ' ' || e.LastName AS Employee,
    COUNT(c.customerid) AS Total_Customer
FROM Employee AS e
INNER JOIN Customer AS c
ON e.EmployeeId = c.SupportRepId
GROUP BY 1
ORDER BY 1 DESC''')
```



ETL

Big Data

Extrair, Transformar e Carregar (ETL) é um processo utilizado por organizações orientadas a dados para coletar informações de diversas fontes, processá-las e reuni-las em um formato útil. Este processo suporta atividades como descoberta de dados, geração de relatórios, análise e tomada de decisões.

Big Data

As fontes de dados variam em tipo, formato, volume e confiabilidade, necessitando de adequação para uso efetivo. Os destinos desses dados processados incluem bancos de dados, data warehouses ou data lakes, variando conforme as necessidades e implementações técnicas específicas.

Big Data



Segundo

a

Oracle:

Extrair

Durante a extração, o ETL identifica os dados e os copia de suas origens, de forma que possa transportar os dados para o armazenamento de dados de destino. Os dados podem vir de fontes estruturadas e não estruturadas, incluindo documentos, emails, aplicações de negócios, bancos de dados, equipamentos, sensores, terceiros e muito mais.

Big Data

Transformar

Como os dados extraídos são brutos em sua forma original, eles precisam ser mapeados e transformados para prepará-los para o armazenamento de dados eventual. No processo de transformação, o ETL valida, autentica, desduplica e/ou agrega os dados de formas que tornam os dados resultantes confiáveis e consultáveis.

Big Data

Carregar

O ETL move os dados transformados para o armazenamento de dados de destino. Esta etapa pode implicar o carregamento inicial de todos os dados de origem ou pode ser o carregamento de alterações incrementais nos dados de origem. Você pode carregar os dados em tempo real ou em lotes programados.

Big Data

Mais

detalhes:

<https://www.oracle.com/br/integration/what-is-etl/>

Nosso Primeiro ETL

Big Data

No endereço <http://universities.hipolabs.com/>, você encontra uma API que facilita a listagem de universidades por país. Podemos utilizar os dados em formato JSON fornecidos por essa API para criar um banco de dados. Este processo envolve extrair os dados, transformá-los conforme necessário e carregá-los em um sistema de banco de dados adequado.

Big Data

```
[{"web_pages": ["https://www.uniceub.br"], "domains": ["sempreceub.com", "uniceub.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Centro Universit\u00e1rio de Bras\u00edlia, UNICEUB"}, {"web_pages": ["http://www.baraodemaua.br/"], "domains": ["baraodemaua.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Centro Universit\u00e1rio Barao de Maua"}, {"web_pages": ["http://www.brazcubas.br/"], "domains": ["brazcubas.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Braz Cubas"}, {"web_pages": ["http://www.candidomendes.br/"], "domains": ["candidomendes.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Candido Mendes"}, {"web_pages": ["http://www.castelobranco.br/"], "domains": ["castelobranco.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Castelo Branco"}, {"web_pages": ["http://www.claretiano.edu.br/"], "domains": ["claretiano.edu.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Centro Universit\u00e1rio Claretiano"}, {"web_pages": ["http://www.creupi.br/"], "domains": ["creupi.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Centro Regional Universit\u00e1rio de Espir\u00edto Santo do Pinhal"}, {"web_pages": ["http://www.emescam.br/"], "domains": ["emescam.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "EMESCAM - Escola Superior de Ci\u00eancias da Santa Casa de Miseric\u00f3rdia de Vit\u00f3ria"}, {"web_pages": ["http://www.epm.br/"], "domains": ["epm.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Federal de S\u00e3o Paulo"}, {"web_pages": ["http://www.estacio.br/"], "domains": ["estacio.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Est\u00e1cio de S\u00e3o"}, {"web_pages": ["http://www.faap.br/"], "domains": ["faap.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "FAAP - Funda\u00e7\u00e3o Armando Alvares Penteado"}, {"web_pages": ["http://www.faculdadescuritiba.br/"], "domains": ["faculdadescuritiba.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Faculdades Integradas Curitiba"}, {"web_pages": ["http://www.fae.edu/"], "domains": ["fae.edu"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "FAE Business School - Faculdade de Administra\u00e7\u00e3o e Economia"}, {"web_pages": ["http://www.feituverava.com.br/"], "domains": ["feituverava.com.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Fund\u00e7\u00e3o Educacional de Ituverava"}, {"web_pages": ["http://www.fic.br/"], "domains": ["fic.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Faculdade Integradas do Cear\u00e1"}, {"web_pages": ["http://www.fua.br/"], "domains": ["fua.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade do Amazonas"}, {"web_pages": ["http://www.furb.rct-sc.br/"], "domains": ["furb.rct-sc.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Regional de Blumenau"}, {"web_pages": ["http://www.furg.br/"], "domains": ["furg.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Universidade Federal do Rio Grande"}, {"web_pages": ["http://www.impa.br/"], "domains": ["impa.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "Instituto Nacional de Matem\u00e1tica Pura e Aplicada - IMPA"}, {"web_pages": ["http://www.ime.eb.mil.br/"], "domains": ["ime.eb.mil.br"], "alpha_two_code": "BR", "state-province": null, "country": "Brazil", "name": "IME - Instituto Militar de Engenharia"}]
```

Big Data

Vamos nos conectar a um banco de dados SQLite chamado `universities.db` e usar um cursor para executar comandos SQL. Vamos criar uma tabela com várias colunas para armazenar informações sobre universidades, como ID, nome, país, estado/província, páginas da web e domínios.

Big Data

```
# Fazer a requisição à API
response = requests.get(url)
universities = response.json()

# Criar ou conectar ao banco de dados SQLite
conn = sqlite3.connect('universities.db')
c = conn.cursor()

# Criar a tabela, se não existir
c.execute('''
    CREATE TABLE IF NOT EXISTS universities (
        id INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT,
        country TEXT,
        state_province TEXT,
        web_pages TEXT,
        domains TEXT
    );
''')
```

Big Data

Vamos iterar sobre uma lista de universidades, onde cada universidade é representada como um dicionário. Para cada universidade, o código executa uma instrução SQL INSERT para adicionar uma nova linha à tabela. Os valores inseridos são o nome da universidade, o país, o estado ou província, as páginas web e os domínios associados.

Big Data

```
# Inserir dados no banco de dados
for university in universities:
    c.execute('''
        INSERT INTO universities (name, country, state_province, web_pages, domains)
        VALUES (?, ?, ?, ?, ?);
    ''', (
        university['name'],
        university['country'],
        university['state-province'],
        ', '.join(university['web_pages']), # Convertendo listas em strings
        ', '.join(university['domains'])
    ))
```

Big Data

Conectando ao novo banco de dados:

```
connection = sqlite3.connect('universities.db')
```

Big Data



Verificando as tabelas que foram geradas:

```
pd.read_sql("SELECT * FROM sqlite_master WHERE type='table'", connection)
```

Big Data

Acessando o banco das universidades:

```
pd.read_sql("select * from universities", connection)
```


Big Data



Procurando todas as universidades de Pernambuco.

```
pd.read_sql("select * from universities where name like '%Pernambuco%' ", connection)
```

Dúvidas?



Marco Mialaret, MSc

Telefone:

81 98160 7018

E-mail:

marcomialaret@gmail.com

