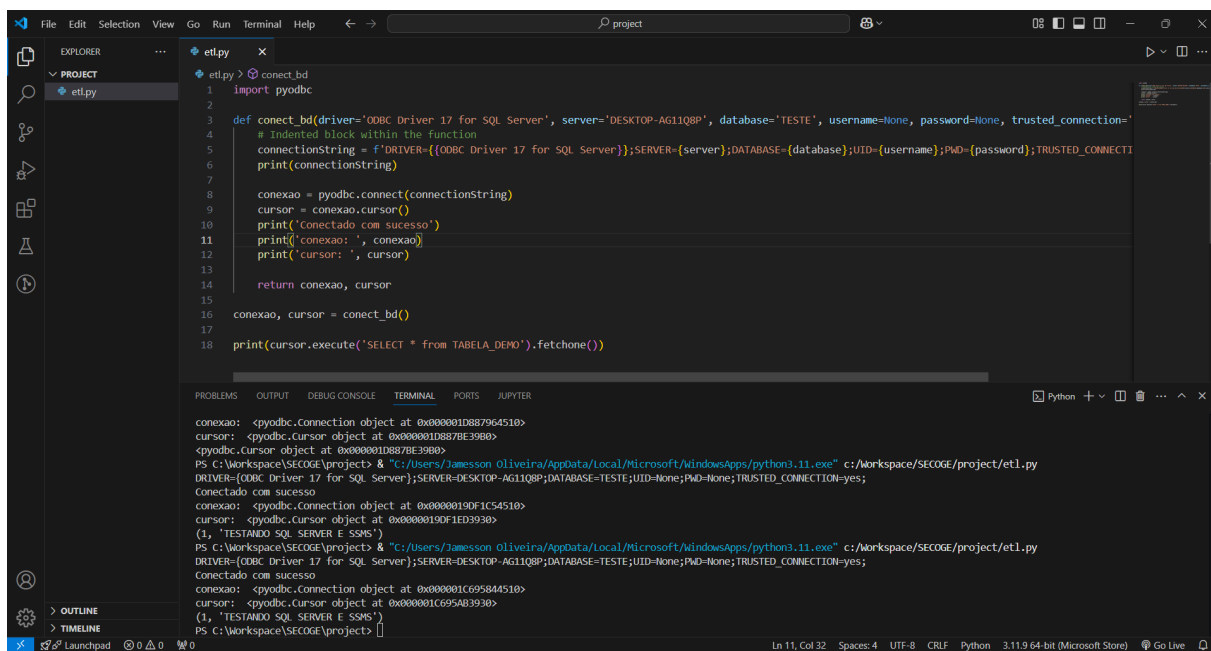


Secoge

bibliotecas baixadas

```
pip install requests
pip install pyodbc
pip install pandas
```

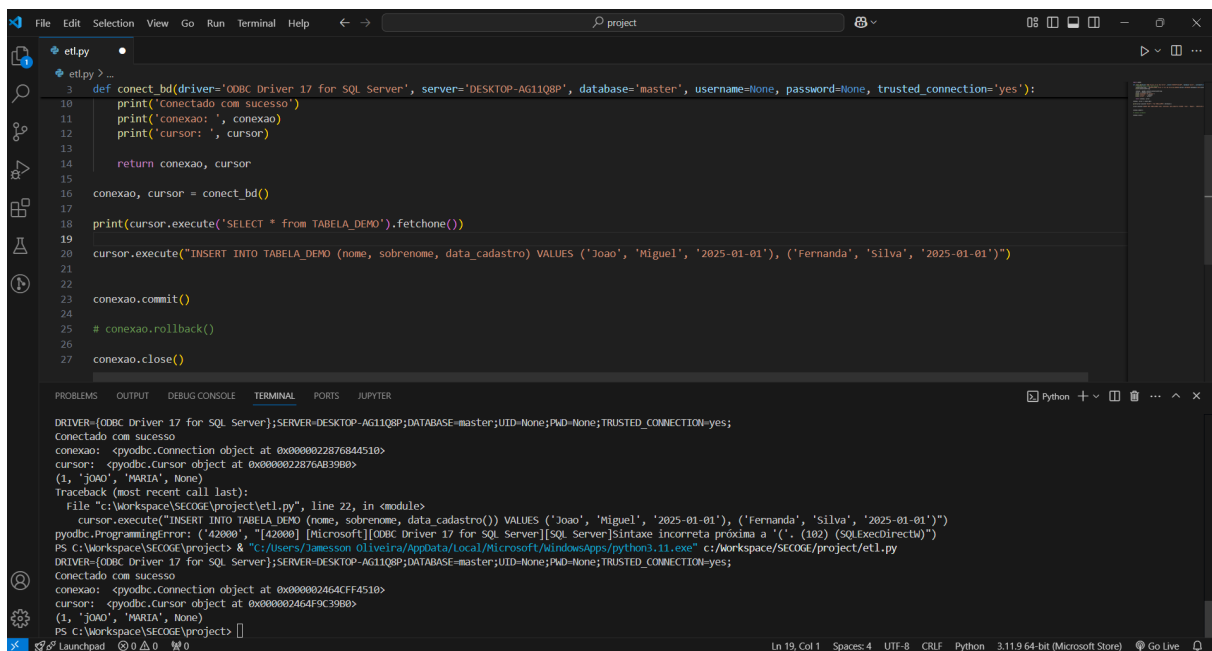
executando o primeiro script de teste para verificar o banco de dados SQL Server



```
File Edit Selection View Go Run Terminal Help
project
EXPLORER
PROJECT
  etl.py
etl.py
1 import pyodbc
2
3 def conect_bd(driver='ODBC Driver 17 for SQL Server', server='DESKTOP-AG1Q8P', database='TESTE', username=None, password=None, trusted_connection='
4     # Indented block within the function
5     connectionString = f'DRIVER={{ODBC Driver 17 for SQL Server}};SERVER={server};DATABASE={database};UID={username};PWD={password};TRUSTED_CONNECTION={trusted_connection}'
6     print(connectionString)
7
8     conexao = pyodbc.connect(connectionString)
9     cursor = conexao.cursor()
10    print('Conectado com sucesso')
11    print('conexao: ', conexao)
12    print('cursor: ', cursor)
13
14    return conexao, cursor
15
16 conexao, cursor = conect_bd()
17
18 print(cursor.execute('SELECT * from TABELA_DEMO').fetchone())

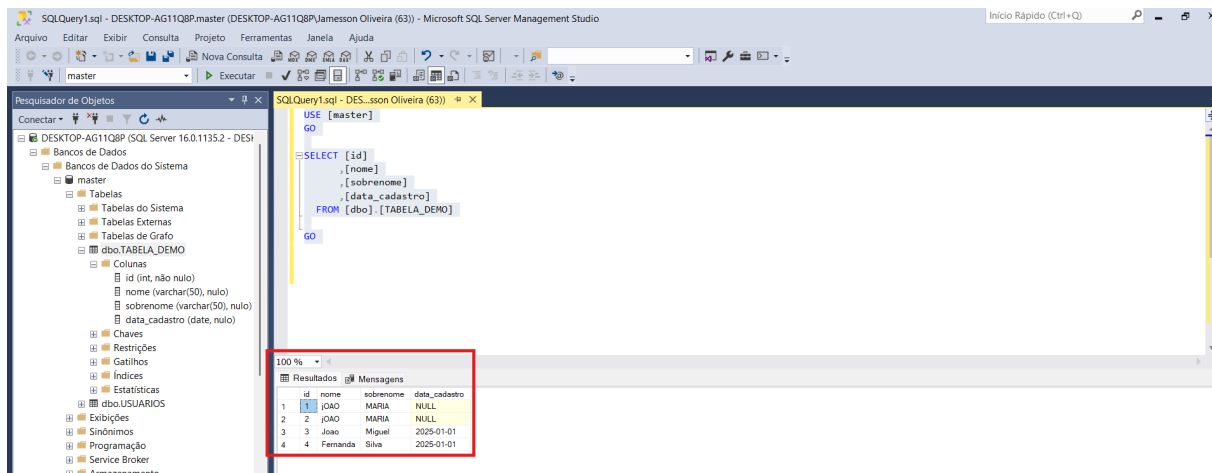
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER
Python
conexao: <pyodbc.Connection object at 0x000001D887964510>
cursor: <pyodbc.Cursor object at 0x000001D887BE3980>
<pyodbc.Cursor object at 0x000001D887BE3980>
PS C:\Workspace\SECOCG\project> "C:/Users/jamesson.Oliveira/AppData/Local/Microsoft/WindowsApps/python3.11.exe" c:\Workspace\SECOCG\project\etl.py
DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-AG1Q8P;DATABASE=TESTE;UID=None;PWD=None;TRUSTED_CONNECTION=yes;
Conectado com sucesso
conexao: <pyodbc.Connection object at 0x00000190F1C54510>
cursor: <pyodbc.Cursor object at 0x00000190F1ED3930>
(1, 'TESTANDO SQL SERVER E SSMS')
PS C:\Workspace\SECOCG\project> "C:/Users/jamesson.Oliveira/AppData/Local/Microsoft/WindowsApps/python3.11.exe" c:\Workspace\SECOCG\project\etl.py
DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-AG1Q8P;DATABASE=TESTE;UID=None;PWD=None;TRUSTED_CONNECTION=yes;
Conectado com sucesso
conexao: <pyodbc.Connection object at 0x000001C695844510>
cursor: <pyodbc.Cursor object at 0x000001C695A83930>
(1, 'TESTANDO SQL SERVER E SSMS')
PS C:\Workspace\SECOCG\project>
```

executando o segundo script de teste, realizando a gravação dos dados no banco de dados SQL Server.



```
def conect_bd(driver='ODBC Driver 17 for SQL Server', server='DESKTOP-AG11Q8P', database='master', username=None, password=None, trusted_connection='yes'):  
    print('Conectado com sucesso')  
    print('conexao:', conexao)  
    print('cursor:', cursor)  
    return conexao, cursor  
conexao, cursor = conect_bd()  
print(cursor.execute('SELECT * from TABELA_DEMO').fetchone())  
cursor.execute("INSERT INTO TABELA_DEMO (nome, sobrenome, data_cadastro) VALUES ('Joao', 'Miguel', '2025-01-01'), ('Fernanda', 'Silva', '2025-01-01'))"  
conexao.commit()  
# conexao.rollback()  
conexao.close()
```

DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-AG11Q8P;DATABASE=master;UID=None;PWD=None;TRUSTED_CONNECTION=yes;
Conectado com sucesso
conexao: <pyodbc.Connection object at 0x000002287684510>
cursor: <pyodbc.Cursor object at 0x000002287684510>
(1, 'JOAO', 'MARIA', None)
Traceback (most recent call last):
 File "c:\Workspace\SEC0GE\project\etl.py", line 22, in <module>
 cursor.execute("INSERT INTO TABELA_DEMO (nome, sobrenome, data_cadastro) VALUES ('Joao', 'Miguel', '2025-01-01'), ('Fernanda', 'Silva', '2025-01-01'))"
pyodbc.ProgrammingError: ('42000', "[42000] [Microsoft][ODBC Driver 17 for SQL Server][SQL Server]sintaxe incorreta próxima a '('. (102) (SQLExecDirectW)")
PS C:\Workspace\SEC0GE\project> & "C:/Users/Jamesson Oliveira/AppData/Local/Microsoft/WindowsApps/python3.11.exe" c:\Workspace\SEC0GE\project\etl.py
DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-AG11Q8P;DATABASE=master;UID=None;PWD=None;TRUSTED_CONNECTION=yes;
Conectado com sucesso
conexao: <pyodbc.Connection object at 0x000002464FC4510>
cursor: <pyodbc.Cursor object at 0x000002464FC4510>
(1, 'JOAO', 'MARIA', None)
PS C:\Workspace\SEC0GE\project>



Modelo que utilizei para criar a tabela no SQLServer

SQLQuery3.sql - DES...sson Oliveira (611)* SQLQuery1.sql - DES...sson Oliveira (54)

```

USE SECOGE
CREATE TABLE futebol_statistica(
    id INT IDENTITY(1,1),
    Período VARCHAR(50),
    Grupo VARCHAR(50),
    Statística VARCHAR(50),
    Casa VARCHAR(50),
    Visitante VARCHAR(50)
)

SELECT * FROM futebol_statistica

```

100 %

Resultados Mensagens

	id	Período	Grupo	Statística	Casa	Visitante
1	1	ALL	Match overview	Ball possession	46%	54%
2	2	ALL	Match overview	Expected goals	1.88	0.32
3	3	ALL	Match overview	Big chances	4	1
4	4	ALL	Match overview	Total shots	17	5
5	5	ALL	Match overview	Goalkeeper saves	0	5
6	6	ALL	Match overview	Corner kicks	3	0
7	7	ALL	Match overview	Fouls	8	3
8	8	ALL	Match overview	Passes	503	601
9	9	ALL	Match overview	Tackles	13	21
10	10	ALL	Match overview	Free kicks	3	8
11	11	ALL	Shots	Total shots	17	5
12	12	ALL	Shots	Shots on target	8	1
13	13	ALL	Shots	Hit woodwork	0	0
14	14	ALL	Shots	Shots off target	8	3
15	15	ALL	Shots	Blocked shots	1	1
16	16	ALL	Shots	Shots inside box	10	1
17	17	ALL	Shots	Shots outside box	7	4

✓ Consulta executada com êxito. DESKTOP-AG11Q8P (16.0 RTM) DESKTOP-AG11Q8PJamess... SECOGE 00:00:00

execucao final, script automatizado

```
67
68 conexao, cursor = conect_bd()
69
70
71 # Inserir cada linha do DataFrame na tabela SQL
72 try:
73     for index, row in df.iterrows():
74         cursor.execute("INSERT INTO futebol_statistica (Visitante, Casa, Sta
75             row['Visitante'], row['Casa'], row['Statistica'], row
76         conexao.commit()
77 except pyodbc.DatabaseError as e:
78     print(e)
79     conexao.rollback()
80 finally:
81     # print(cursor.execute('SELECT id from futebol').fetchall())
82     conexao.close()
83
84
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER Python + - □ □ ... ^ ×

```
PS C:\Workspace\SECOGE\project> & "C:/Users/Jamesson Oliveira/AppData/Local/Microsoft/windowsApps/python3.11.exe" c:/Workspace/SECOGE/project/etl.py
DRIVER={ODBC Driver 17 for SQL Server};SERVER=DESKTOP-AG11Q8P;DATABASE=SECOGE;UID=None;PWD=None;TRUSTED_CONNECTION=yes;
Conectado com sucesso
conexao: <pyodbc.Connection object at 0x0000025E0349EB40>
cursor: <pyodbc.Cursor object at 0x0000025E0397D130>
PS C:\Workspace\SECOGE\project> □
```

> OUTLINE
> TIMELINE

Launchpad 0 0 0 0 Ln 79, Col 23 Spaces: 4 UTF-8 CRLF Python 3.11.9 64-bit (Microsoft Store) Go Live

modelo gerado com pandas no excel

	A	B	C	D	E
1	Periodo	Grupo	Statistica	Casa	Visitante
2	ALL	Match overview	Ball possession	46%	54%
3	ALL	Match overview	Expected goals	1.88	0.32
4	ALL	Match overview	Big chances	4	1
5	ALL	Match overview	Total shots	17	5
6	ALL	Match overview	Goalkeeper saves	0	5
7	ALL	Match overview	Corner kicks	3	0
8	ALL	Match overview	Fouls	8	3
9	ALL	Match overview	Passes	503	601
10	ALL	Match overview	Tackles	13	21
11	ALL	Match overview	Free kicks	3	8
12	ALL	Shots	Total shots	17	5
13	ALL	Shots	Shots on target	8	1
14	ALL	Shots	Hit woodwork	0	0
15	ALL	Shots	Shots off target	8	3
16	ALL	Shots	Blocked shots	1	1
17	ALL	Shots	Shots inside box	10	1
18	ALL	Shots	Shots outside box	7	4
19	ALL	Attack	Big chances scored	2	1
20	ALL	Attack	Big chances missed	2	0
21	ALL	Attack	Through balls	1	0
22	ALL	Attack	Touches in penalty area	18	2
23	ALL	Attack	Offsides	1	0
24	ALL	Passes	Accurate passes	466	545
25	ALL	Passes	Throw-ins	11	9
26	ALL	Passes	Final third entries	60	22

Dados utilizados:

Escolhi dados de futebol por ter afinidade, não encontrei o do meu time do coração então peguei outro qualquer mas pude aplicar alguns filtros de qualidade nos dados

link dos dados:

<https://www.sofascore.com/pt/football/match/sao-paulo-botafogo/iOsGO#id:12117350>

escolhi esses dados por gostar de esportes e apostas esportivas, pude ter uma base de como funciona as estatísticas de futebol, claro que não apliquei nenhuma técnica de machine learning, mas pude verificar os dados e tratá-los.

Extração e Transformação:

apliquei um laço de repetição para extrair todas as informações que eu queria, salvei tudo num dataframe do pandas e apliquei algumas transformações no pandas mesmo.

Consultas:

sobre as bibliotecas utilizadas:

```
requests; pyodbc; pandas
```

escolhi pois são simples de utilizar, tem bastante documentação e também foi indicação do desafio.

request faz requisições bem simples na web, foi onde eu pude aplicar o scraping através da WEB através do HTML.

pyodbc foi o mais complicado entre todas as bibliotecas, mas conseguir após várias tentativas de conectar com o SQLServer. Depois da barreira inicial se torna uma lib bem simples e fácil de aplicar.

O pandas já utilizo há algum tempo foi onde eu pude resolver alguns problemas para tratar os dados e foi uma parte bem tranquila.

Considerações finais:

no início preferia realizar o projeto no colab por ser mais prático, porém foram surgindo problemas com instabilidade e por ser uma ferramenta web, no final acabei optando pelo bom e velho VSCode, que nunca me deixa na mão e quando tem algum problema consigo resolver sem muitas dificuldades.

Também pensei em utilizar o pySpark mas como era um volume de dados pequeno não valia o esforço, o pandas foi ótimo nessa situação.