Extreme Digital Solutions

Avaliação Data Engineer

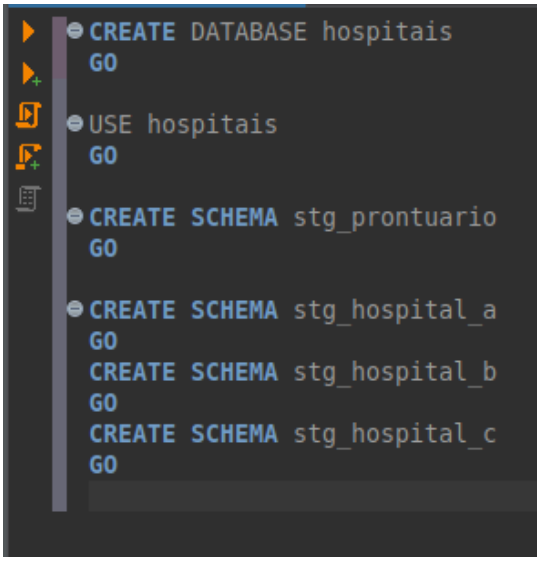
Resolução do Exercício Proposto

Hugo Leonardo Gomes da Silva

Recife
2022

Contexto

Vamos contextualizar os passos para implementar as questões solicitadas:

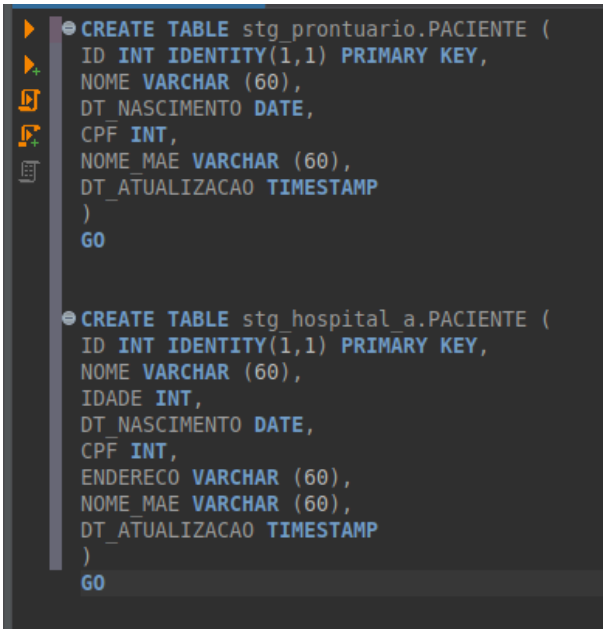


```
CREATE DATABASE hospitais
GO

USE hospitais
GO

CREATE SCHEMA stg_prontuario
GO

CREATE SCHEMA stg_hospital_a
GO
CREATE SCHEMA stg_hospital_b
GO
CREATE SCHEMA stg_hospital_c
GO
```

No contexto é descrito os ‘Schemas’ que será utilizado nos passos seguintes.

Problema 1

Criamos a tabela Paciente da questão 1 no schema stg_prontuario. Aproveitamos também para criar a tabela Paciente no schema stg_hospital_a para realizar a questão 2.



```
CREATE TABLE stg_prontuario.PACIENTE (
ID INT IDENTITY(1,1) PRIMARY KEY,
NOME VARCHAR (60),
DT_NASCIMENTO DATE,
CPF INT,
NOME_MAE VARCHAR (60),
DT_ATUALIZACAO TIMESTAMP
)
GO


CREATE TABLE stg_hospital_a.PACIENTE (
ID INT IDENTITY(1,1) PRIMARY KEY,
NOME VARCHAR (60),
IDADE INT,
DT_NASCIMENTO DATE,
CPF INT,
ENDERECO VARCHAR (60),
NOME_MAE VARCHAR (60),
DT_ATUALIZACAO TIMESTAMP
)
GO
```

Podemos observar que criamos mais duas colunas na tabela Paciente no schema stg_hospital_a para diferenciar da tabela stg_prontuario.Paciente.

Problema 2

Inserimos mais dois usuários na tabela para identificarmos os pacientes copiados.

```
INSERT INTO stg_hospital_a.PACIENTE (NOME, IDADE, DT_NASCIMENTO, CPF, ENDERECO, NOME_MAE) VALUES ('Hugo', 55, '2000-11-08', 089257, 'casa de voinha', 'Gorete')
go
INSERT INTO stg_hospital_a.PACIENTE (NOME, IDADE, DT_NASCIMENTO, CPF, ENDERECO, NOME_MAE) VALUES ('BiluBilu', 35, '2009-12-08', 099297, 'casa de kaka', 'Lorete')
go

INSERT INTO stg_prontuario.PACIENTE (NOME, DT_NASCIMENTO, CPF, NOME_MAE) select NOME, DT_NASCIMENTO, CPF, NOME_MAE from stg_hospital_a.PACIENTE
```

Após a inserção dos usuários realizamos a cópia da tabela stg_hospital_a.PAciente para stg_prontuario.Paciente. Só copiamos os campos solicitados.

Problema 3

```
INSERT INTO stg_prontuario.PACIENTE (NOME, DT_NASCIMENTO, CPF, NOME_MAE) VALUES ('Hugo', '2000-11-08', 089257, 'Gorete')
go

INSERT INTO stg_prontuario.PACIENTE (NOME, DT_NASCIMENTO, CPF, NOME_MAE) VALUES ('BiluBilu', '2009-12-08', 099297, 'Lorete')
go

INSERT INTO stg_prontuario.PACIENTE (NOME, DT_NASCIMENTO, CPF, NOME_MAE) VALUES ('LauLau', '2005-10-09', 000007, 'Borete')
go

SELECT NOME FROM stg_prontuario.PACIENTE  WHERE stg_prontuario.PACIENTE.CPF IN (SELECT CPF FROM stg_prontuario.PACIENTE GROUP BY CPF HAVING Count(*) > 1) GROUP BY NOME HAVING Count(*) > 1
GO
```

Adicionamos mais usuários com o objetivo de capturar os pacientes duplicados. Após a inserção realizamos um ‘select’ para apresentar os pacientes duplicados.

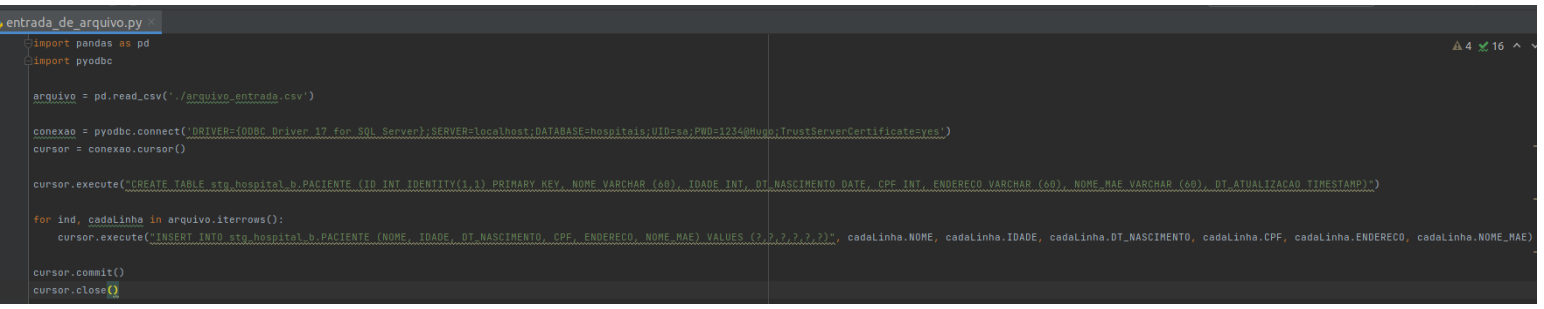
Problema 4

```
SELECT NOME, MAX(DT_ATUALIZACAO) FROM stg_prontuario.PACIENTE  WHERE stg_prontuario.PACIENTE.CPF IN \
    (SELECT CPF FROM stg_prontuario.PACIENTE GROUP BY CPF HAVING Count(*) > 1) GROUP BY NOME HAVING Count(*) > 1
```

Realizamos uma pequena alteração no código da questão anterior com o objetivo de apresentar só os nomes dos duplicatas.

Problema 5

Utilizamos a biblioteca ‘pandas’ para abrir o arquivo que criamos e em seguida utilizamos a biblioteca ‘pyodbc’ para acessar o banco.



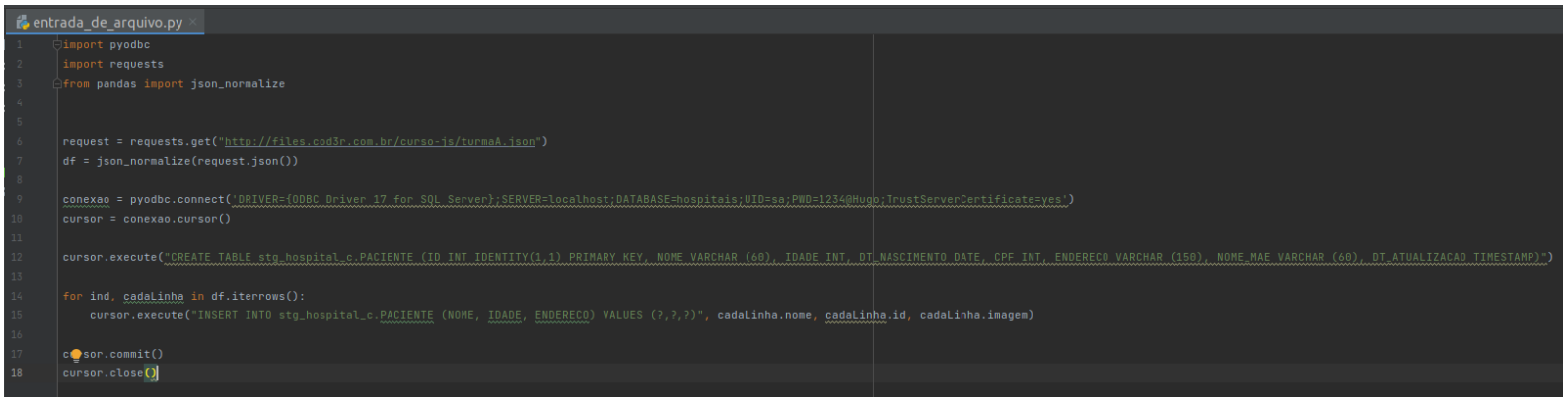
```python
import pandas as pd
import pyodbc

arquivo = pd.read_csv('./arquivo_entrada.csv')

conexao = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};SERVER=localhost;DATABASE=hospitais;UID=sa;PWD=1234@Hugo;TrustServerCertificate=yes')
cursor = conexao.cursor()

cursor.execute("CREATE TABLE stg_hospital_b.PACIENTE (ID INT IDENTITY(1,1) PRIMARY KEY, NOME VARCHAR (60), IDADE INT, DT_NASCIMENTO DATE, CPF INT, ENDERECO VARCHAR (60), NOME_MAE VARCHAR (60), DT_ATUALIZACAO TIMESTAMP)")

for ind, cadaLinha in arquivo.iterrows():
    cursor.execute("INSERT INTO stg_hospital_b.PACIENTE (NOME, IDADE, DT_NASCIMENTO, CPF, ENDERECO, NOME_MAE) VALUES (?,?,?,?,?,?)", cadaLinha.NOME, cadaLinha.IDADE, cadaLinha.DT_NASCIMENTO, cadaLinha.CPF, cadaLinha.ENDERECO, cadaLinha.NOME_MAE)

cursor.commit()
cursor.close()
```

Arquivo CSV:

	A	B	C	D	E	F
1	NOME	IDADE	DT_NASCIMENTO	CPF	ENDERECO	NOME_MAE
2	1A	13	2009-11-08	47301	lacteo casa	lululu
3	2B	8	2010-10-07	78216	refri amarelo	cacaca
4	3C	26	2019-12-09	60075	carnes bovina	cococo
5						

Problema 6

Utilizamos uma nova biblioteca em relação à questão anterior, ‘requests’ uma biblioteca para fazer requisição http. Obs: Utilizamos um link disponibilizado em um curso em Js que retorna um .json, com o objetivo de implementar a requisição em uma API REST.
Como esse .json não continha muita informação, só extraímos os campos ‘NOME’, ‘IDADE’, ‘ENDERECO’



```python
import pyodbc
import requests
from pandas import json_normalize


request = requests.get("http://files.cod3r.com.br/curso-js/turmaA.json")
df = json_normalize(request.json())

conexao = pyodbc.connect('DRIVER={ODBC Driver 17 for SQL Server};SERVER=localhost;DATABASE=hospitais;UID=sa;PWD=1234@Hugo;TrustServerCertificate=yes')
cursor = conexao.cursor()

cursor.execute("CREATE TABLE stg_hospital_c.PACIENTE (ID INT IDENTITY(1,1) PRIMARY KEY, NOME VARCHAR (60), IDADE INT, DT_NASCIMENTO DATE, CPF INT, ENDERECO VARCHAR (150), NOME_MAE VARCHAR (60), DT_ATUALIZACAO TIMESTAMP)")

for ind, cadaLinha in df.iterrows():
    cursor.execute("INSERT INTO stg_hospital_c.PACIENTE (NOME, IDADE, ENDERECO) VALUES (?,?,?)", cadaLinha.nome, cadaLinha.id, cadaLinha.imagem)

cursor.commit()
cursor.close()
```

Problema 7

Eu faria uma relação 1 para n. Para cada ocorrência na tabela Paciente poderíamos ter várias referências para a tabela que contém os diagnósticos.

Problema 9

A ideia descrita no código abaixo foi, separar a primeira string e contar na segunda string, o resultado tem que ser igual ou superior no estoque.

Problema 10

Nessa última questão, utilizamos ‘collections’ para realizar uma contagem e em seguida plotamos o grafico utilizando a biblioteca ‘Matplotlib’.

O resultado do do código implementado a acima é a figura 01.

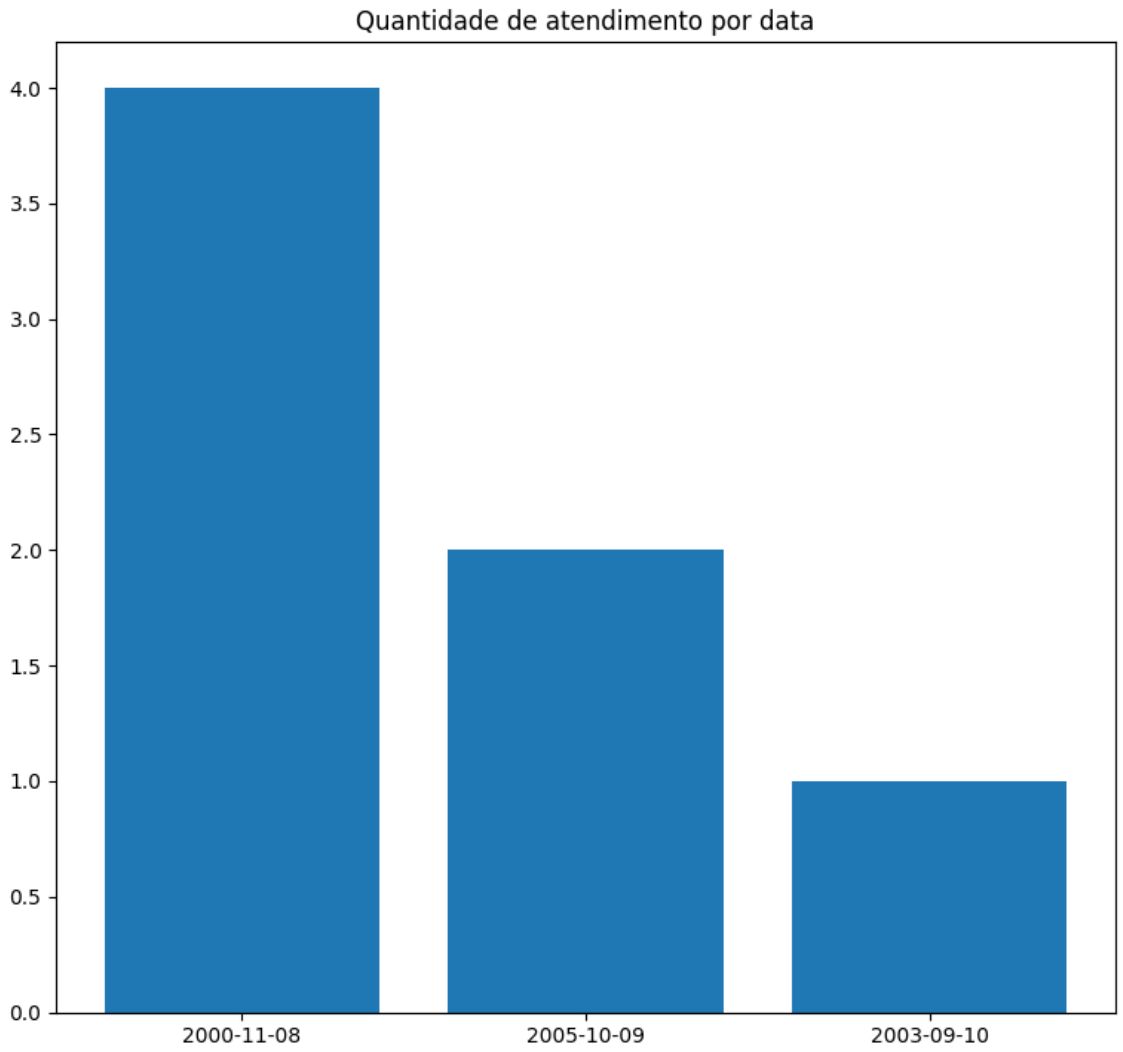


Figura 01

Descrições:

Foi implementado o SQL SERVER em um contêiner do Docker.

Seguintes comandos:

Efetue pull da imagem de contêiner do SQL Server 2019 Linux no Registro de Contêiner da Microsoft.

```bash
sudo docker pull mcr.microsoft.com/mssql/server:2019-latest
```

Para executar a imagem de contêiner com o Docker, você pode usar o comando a seguir de um shell bash (Linux/macOS) ou do prompt de comando do PowerShell elevado.

```bash
sudo docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=<YourStrong@Passw0rd>" \
   -p 1433:1433 --name sql1 -hostname sql1 \
   -d mcr.microsoft.com/mssql/server:2019-latest
```

Para exibir seus contêineres do Docker, use o comando docker ps:

```bash
sudo docker ps -a
```

Use o comando docker exec -it para iniciar um shell bash interativo dentro do contêiner em execução.

```bash
sudo docker exec -it sql1 "bash"
```

Quando estiver dentro do contêiner, conecte-se localmente com a sqlcmd.

```bash
/opt/mssql-tools/bin/sqlcmd -S localhost -U SA -P "<YourNewStrong@Passw0rd>"
```