Participante: Gabriela Araujo de Oliveira Email: gabrielabela1196@gmail.com

Processo: Mercado Livre

Documentação de todos os passos executados

1. Realiza a importação das bibliotecas necessárias para rodar o código

```
import pandas as pd
import pyodbc
import pysftp
import sqlalchemy
```

a.

- b. Pandas O pyodbc é um módulo Python que fornece uma interface para conectar e interagir com bancos de dados relacionais usando o padrão ODBC (Open Database Connectivity). O ODBC é um padrão de API que permite que aplicativos acessem e manipulem dados em uma variedade de bancos de dados relacionais, tornando o pyodbc uma escolha popular para desenvolvedores que precisam se conectar a esses tipos de bancos de dados a partir de aplicativos Python.
- c. Pyodbc É um módulo Python que fornece uma interface para interagir com bancos de dados relacionais por meio do padrão ODBC (Open Database Connectivity). O ODBC é um padrão de API que permite que aplicativos se conectem a diferentes sistemas de gerenciamento de bancos de dados relacionais (RDBMS) de maneira consistente, independentemente do banco de dados subjacente. Isso torna o pyodbc uma escolha popular para desenvolvedores que desejam conectar aplicativos Python a diversos bancos de dados.
- d. Pysftp É um módulo Python que oferece uma interface para transferência de arquivos segura por meio do protocolo SFTP (SSH File Transfer Protocol). O SFTP é uma extensão segura do SSH (Secure Shell) que permite transferir arquivos de forma segura entre um cliente e um servidor, com autenticação e criptografia.
- e. Sqlalchemy Biblioteca de mapeamento objeto-relacional (ORM) para Python. Ela é usada para simplificar a interação com bancos de dados relacionais, como MySQL, PostgreSQL, SQLite e Oracle, permitindo que os desenvolvedores acessem o banco de dados usando objetos Python em vez de escrever consultas SQL diretamente.
- 2. Parâmetros de conexão com a base de dados do SQL Server

```
server = 'Nombre del servidor'
database = 'Nombre de la base de datos'
username = 'Nombre de usuario'
password = 'Su contraseña'
```

3. Cria uma conexão com o servidor SQL

```
conn = pyodbc.connect(
   f'DRIVER={{SQL Server}};'
   f'SERVER={server};'
   f'DATABASE={database};'
   f'UID={username};'
   f'PWD={password};'
)
```

4. Realiza a inserção do DataFrame no banco de dados.

```
df.to_sql(name='Nombre de la tabla', con=conn, if_exists='replace', index=False)
```

5. Cria um cursor para executar uma consulta SQL na tabela que deseja compartilhar

```
cursor = conn.cursor()
query = "SELECT * FROM TablaEjemplo"
cursor.execute(query)
```

6. Recupera o resultado das Querys

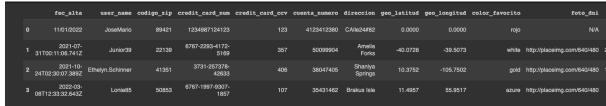
а

а

a.

a.

a.



7. Parâmetros de conexão com SFTP

```
sftp_host = 'sftp_server_hostname'
sftp_port = 22
sftp_user = 'tu_usuario_sftp'
sftp_password = 'tu_contraseña_sftp'
```

b. Caminho e nome do arquivo a ser criado no servidor SFTP

```
# Ruta y nombre del archivo que se creará en el servidor SFTP
remote_filepath = '/ruta/remoto/nombre_archivo.csv'

# Conéctese al servidor SFTP
with pysftp.Connection(host=sftp_host, username=sftp_user, password=sftp_password, port=sftp_port) as sftp:
    # Convertir los resultados de la consulta a un archivo CSV
    csv_data = '\n'.join([','.join(map(str, row)) for row in results])

# Cargue el archivo CSV al servidor SFTP
    sftp.putfo(io.StringIO(csv_data), remote_filepath)
```

Este código autentica-se com as ferramentas necessárias e realiza a inserção do arquivo gerado pela consulta no caminho desejado do servidor SFTP. Para criar este código, utilizamos as linguagens Python e SQL, com a ferramenta Colab Research.

É importante ressaltar que o código não realiza a comunicação com um banco de dados ou servidor SFTP real, pois essas informações não foram fornecidas na atividade. No entanto, ele demonstra o procedimento correto para cada ferramenta. Optamos pelo uso do Colab devido à sua acessibilidade e à ausência da necessidade de instalar aplicativos localmente na máquina, tornando-o facilmente acessível para qualquer avaliador.