

Relatório do programa de manipulação do banco de dados

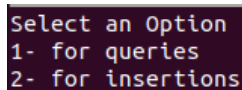
O programa foi escrito em linguagem de programação python, fazendo-se o uso da biblioteca **psycopg2**, específica para a comunicação com o postgres.

Inicialmente o programa pede o usuário e senha para se conectar ao BD. Foi criada uma flag para se usar uma entrada default, e então é feita a conexão, conforme ilustrado abaixo:

```
def my_connect():
    if DEFAULT_USER_PASS==1:
        db_user="postgres"
        db_password="postgres"
    else:
        db_user= str(input("enter database user: "))
        db_password= str(input("enter database password: "))

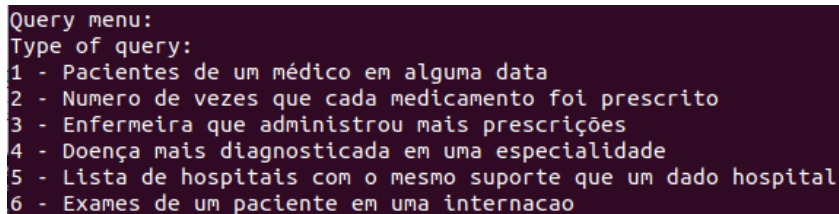
    return psycopg2.connect(host="localhost",
                           database=DB_NAME,
                           user=db_user,
                           password=db_password)
```

O programa possui um menu textual onde o usuário escolhe entre funcionalidades de consulta e inserção:



```
Select an Option
1- for queries
2- for insertions
█
```

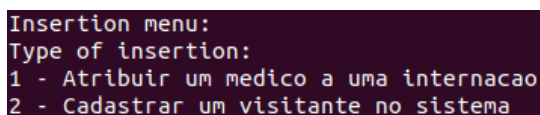
O menu de consultas(queries) possui as seguintes opções, correspondendo às 6 consultas da parte3 do trabalho:



```
Query menu:
Type of query:
1 - Pacientes de um médico em alguma data
2 - Numero de vezes que cada medicamento foi prescrito
3 - Enfermeira que administrou mais prescrições
4 - Doença mais diagnosticada em uma especialidade
5 - Lista de hospitais com o mesmo suporte que um dado hospital
6 - Exames de um paciente em uma internacao
█
```

Para cada uma dessas opções, é pedido um valor de entrada do usuário, e após o fornecimento desse valor, é exibido o resultado da consulta.

O menu de inserções possui as seguintes opções, correspondendo à inserção de tabela que dispara o procedimento sql requisitado na parte3, e mais uma outra opção de inserção, com fins de diversificação de funcionalidades:



```
Insertion menu:
Type of insertion:
1 - Atribuir um medico a uma internacao
2 - Cadastrar um visitante no sistema
█
```

Para cada inserção são pedidos os atributos necessários para se realizar a inserção e após o comando ser executado, é exibida uma mensagem que indica se o usuário foi bem sucedido.

Abaixo é apresentado o trecho de código responsável pela primeira consulta (todas as outras acontecem de forma análoga):

```
def Query1(cur):
    os.system('clear');
    print('1 - Pacientes de um médico em alguma data')
    medico= str(input('Nome do medico(a): '))
    data= str(input('Data de Internação(yyyy-mm-dd): '))
    sqlStr= "SELECT * \
            FROM Prontuario NATURAL JOIN Internacao \
            NATURAL JOIN Assiste \
            NATURAL JOIN Medica \
            JOIN Funcionario USING (codigo_func) \
            WHERE Funcionario.nome = %s AND \
            Internacao.data_inicio <= %s \
            AND (Internacao.data_fim IS NULL OR Internacao.data_fim >= %s);"

    cur.execute(sqlStr,(medico,data,data))
    rows = cur.fetchall()
    print("The number of patients: ", cur.rowcount)
    for row in rows:
        print(row)
```

A variável sqlStr guarda a consulta, com as entradas tendo seu valor reservado por '%s'.

A execução da consulta se dá pelo comando do cursor `cur.execute(sqlStr,(medico,data,data))`, onde o valor `medico` e `data` foram fornecidos pelo usuário. O valor de retorno da consulta é obtido pelo comando `rows = cur.fetchall()`.

Os comandos de inserção ocorrem de forma análoga, com a exceção de que é necessário que se execute a função de commit, para que a operação seja efetivada no banco de dados :

```
def Insert2(conn, cur):
    os.system('clear')
    print('2 - Cadastrar um visitante no sistema')
    nome= str(input('informe o nome do visitante: '))
    rg= str(input('informe o rg do visitante: '))
    sqlStr="INSERT INTO Visitante VALUES( %s, %s);"
    cur.execute(sqlStr,(rg, nome))
    conn.commit()
```