HOW TO

**DESAFIO DE BANCO DE DADOS**

* + Criar um SCRIPT que gere dados para poderem ser inseridos no banco com mais rapidez e assertividade.
  + Esse script poderá ser feito em qualquer linguagem, mas a linguagem Python é uma que possui mais compatibilidade com bibliotecas que possam te ajudar durante o desenvolvimento do código
  + O script terá que possuir a biblioteca da Oracle [ORACLEDB] e uma outra biblioteca que será opcional que é a [FAKER] que consegue gerar dados para cada situação que você precise.
  + O script começará contendo os imports das bibliotecas, seguindo com a criação da conexão com o banco de dados da Oracle colocando esse comando abaixo:

conn = oracledb.connect(user="rm552164", password="fiap23", dsn="oracle.fiap.com.br/orcl")

cur = conn.cursor()

* + **O script será bem adaptado para as determinadas colunas da tabela alvo, no caso da tabela paciente, o código ficará assim:**

from datetime import datetime

import random

from faker import Faker

import oracledb

fake = Faker('pt\_BR')

conn = oracledb.connect(user="rm551763", password="fiap23", dsn="oracle.fiap.com.br/orcl")

cur = conn.cursor()

id\_counter = 5001

cpf\_set = set()

rg\_set = set()

data\_minima = datetime(1900, 1, 1)

data\_maxima = datetime.now()

sql = """INSERT INTO T\_RHSTU\_PACIENTE

             (ID\_PACIENTE, NM\_PACIENTE, NR\_CPF, NM\_RG, DT\_NASCIMENTO, FL\_SEXO\_BIOLOGICO, DS\_ESCOLARIDADE, DS\_ESTADO\_CIVIL, NM\_GRUPO\_SANGUINEO, NR\_ALTURA, NR\_PESO, DT\_CADASTRO, NM\_USUARIO)

             VALUES

             (:ID\_PACIENTE, :NM\_PACIENTE, :NR\_CPF, :NM\_RG, TO\_DATE(:DT\_NASCIMENTO, 'DD-MON-YYYY'), :FL\_SEXO\_BIOLOGICO, :DS\_ESCOLARIDADE, :DS\_ESTADO\_CIVIL, :NM\_GRUPO\_SANGUINEO, :NR\_ALTURA, :NR\_PESO, SYSDATE, USER)"""

for \_ in range(1000):

    id = id\_counter

    id\_counter += 1

    nome = fake.name()

    while True:

        cpf = fake.unique.random\_int(min=100000000, max=999999999)

        if cpf not in cpf\_set:

            cpf\_set.add(cpf)

            break

    while True:

        rg = fake.unique.random\_int(min=10000000, max=99999999)

        if rg not in rg\_set:

            rg\_set.add(rg)

            break

    nasc = fake.date\_between\_dates(data\_minima, data\_maxima).strftime('%d-%b-%Y').upper()

    sexo = random.choice(["M", "F", "I"])

    escolaridade = fake.random\_element(

        elements=(

            "ENSINO FUNDAMENTAL",

            "ENSINO MÉDIO",

            "ENSINO SUPERIOR",

            "PÓS-GRADUAÇÃO",

            "MESTRADO",

            "DOUTORADO",

            "PÓS-DOUTORADO",

            "PHD",

        )

    )

    status\_civil = fake.random\_element(

        elements=(

            "SOLTEIRO",

            "CASADO",

            "DIVORCIADO",

            "VIÚVO",

            "SEPARADO",

            "UNIÃO ESTÁVEL",

        )

    )

    tipo\_sanguineo = fake.random\_element(elements=(

"A+",

"B+",

"AB+",

"O+",

"A-",

"B-",

"AB-",

"O-"))

    altura = random.uniform(0.5, 2.3)

    peso = random.uniform(5, 500)

    cur.execute(

        sql,

        ID\_PACIENTE=id,

        NM\_PACIENTE=nome,

        NR\_CPF=str(cpf),

        NM\_RG=str(rg),

        DT\_NASCIMENTO=nasc,

        FL\_SEXO\_BIOLOGICO=sexo,

        DS\_ESCOLARIDADE=escolaridade,

        DS\_ESTADO\_CIVIL=status\_civil,

        NM\_GRUPO\_SANGUINEO=tipo\_sanguineo,

        NR\_ALTURA=altura,

        NR\_PESO=peso,

    )

    conn.commit()

cur.close()

conn.close()

* + Como visto acima, a biblioteca Faker instruirá muito a criação de dados para você, facilitando a resolução desse desafio
  + No código acima, conseguimos obter uma leve noção sobre a dinamicidade do código, ou seja, as possibilidades de se mexer e adaptar para suas tabelas, apenas possuir bastante atenção às constraints de suas tabelas e relacionamentos, para garantir uma boa usabilidade desse script.

**Boa sorte !**