BANCO DE DADOS

IMPORTAR MÓDULOS

In [1]:
```
import sqlite3
import os
import io

from faker import Faker
```

GERAR DADOS ALEATÓRIOS

In [2]:
```
data = Faker()
```

GERADOR DE NOME

In [3]:
```
data.name()
```

Out[3]:
```
'Susan Wilkerson'
```

GERADOR DE TELEFONE

In [4]:
```
data.phone_number()
```

Out[4]:
```
'984-076-0191'
```

GERADOR DE DADOS

In [5]:
```
data.profile()
```

Out[5]:
```
{'job': 'Chartered legal executive (England and Wales)',
 'company': 'Green-Oneal',
 'ssn': '477-54-1491',
 'residence': '52316 Adam Divide\nWongborough, OH 09248',
 'current_location': (Decimal('16.557808'), Decimal('171.185504')),
 'blood_group': 'B-',
 'website': ['http://www.erickson-jones.com/',
  'https://www.rogers.org/',
  'https://miller.biz/',
  'https://www.young.com/'],
 'username': 'lambsarah',
 'name': 'Ann Walker',
 'sex': 'F',
 'address': '713 Lori Forks\nJordanside, OR 82531',
 'mail': 'matthew00@hotmail.com',
 'birthdate': datetime.date(2014, 11, 25)}
```

In [6]:
```
p = data.profile()

p['name']
```

Out[6]:
```
'Adam Martinez'
```

CRIAR UMA LISTA COM 99 PESSOAS

In [7]:
```
lista = []

for _ in range(1, 100):
    linha = [data.name(), data.phone_number()]
    lista.append(linha)
```

IMPRIMIR OS 10 PRIMEIROS

In [8]:
```
lista[0:10]
```

Out[8]:
```
[['Jennifer Rose', '+1-925-387-5486x2356'],
 ['Maureen Martin', '(933)470-5036x533'],
 ['Michael Wallace', '112-868-3272x7086'],
 ['Mr. Charles Wade Jr.', '001-033-345-0282'],
 ['Gina Franco', '996-226-4289'],
 ['Jennifer Johnson', '959-006-9819x62702'],
 ['Christine Winters', '236-278-2103x55206'],
 ['Andrew Ballard', '+1-170-687-9212x53745'],
 ['Bryan Sims', '(924)778-1097x299'],
 ['Matthew Mejia', '3383603748']]
```

CRIANDO O BANCO DE DADOS

In [9]:
```
conexao = sqlite3.connect('banco.db', 100)
cursor = conexao.cursor()
```

In [10]:
```
os.listdir()
```

Out[10]:
```
['.ipynb_checkpoints',
 '2020 alunos.csv',
 '2020 Nomes.txt',
 'BANCO DE DADOS.ipynb',
 'banco.db',
 'enve',
 'ESTATÍSTICA INICIAL.ipynb',
 'MANIPULAÇÃO DE ARQUIVOS.ipynb',
 'requerimentos.txt']
```

CRIANDO A TABELA

In [11]:
```
cursor.execute("""
CREATE TABLE if not exists contatos (
CODIGO INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
NOME TEXT,
TELEFONE VARCHAR(50)
);
""")

conexao.commit()
```

LISTAR OS REGISTROS

In [12]:
```
cursor.execute(""" select * from contatos """)

cursor.fetchall()[-10:]
```

Out[12]:
```
[]
```

INCLUIR UM REGISTRO NA TABELA

In [13]:
```
cursor.execute("""
INSERT INTO contatos (NOME, TELEFONE)
VALUES
('{0}', '{1}')""".format(lista[0][0], lista[0][1]))

conexao.commit()
```

In [14]:
```
lista[0]
```

Out[14]:
```
['Jennifer Rose', '+1-925-387-5486x2356']
```

PESQUISAR O REGISTRO INCLUIDO

In [15]:
```
cursor.execute(""" select * from contatos """)

cursor.fetchall()
```

Out[15]:
```
[(1, 'Jennifer Rose', '+1-925-387-5486x2356')]
```

INCLUIR VÁRIOS REGISTROS NA TABELA

In [16]:
```
cursor.executemany("""
INSERT INTO contatos (NOME, TELEFONE)
VALUES
(?, ?)""", lista[1:])

conexao.commit()
```

PESQUISAR OS 10 PRIMEIROS REGISTROS

INCLUIDOS

In [17]:
```
cursor.execute(""" SELECT * FROM CONTATOS; """)

for linha in cursor.fetchall()[0:10]:
    print(linha)
```

```
(1, 'Jennifer Rose', '+1-925-387-5486x2356')
(2, 'Maureen Martin', '(933)470-5036x533')
(3, 'Michael Wallace', '112-868-3272x7086')
(4, 'Mr. Charles Wade Jr.', '001-033-345-0282')
(5, 'Gina Franco', '996-226-4289')
(6, 'Jennifer Johnson', '959-006-9819x62702')
(7, 'Christine Winters', '236-278-2103x55206')
(8, 'Andrew Ballard', '+1-170-687-9212x53745')
(9, 'Bryan Sims', '(924)778-1097x299')
(10, 'Matthew Mejia', '3383603748')
```

PESQUISAR OS NOMES QUE COMEÇAM COM P

In [18]:
```
cursor.execute("""SELECT * FROM contatos WHERE nome like 'P%'; """)
for linha in cursor.fetchall()[0:10]:
    print(linha)
```

```
(39, 'Paul Snyder', '(591)044-2003')
(40, 'Patrick Patterson', '(207)551-6784x7332')
(55, 'Paul Walker', '882-948-2476x3966')
```

UPDATE

In [19]:
```
cursor.execute("""SELECT codigo from contatos where nome like 'P%'; """)

valores = cursor.fetchall()
```

In [20]:
```
valores
```

Out[20]:
```
[(39,), (40,), (55,)]
```

In [21]:
```
for vl in valores:
    sql = """ update contatos set telefone = null where codigo = {0};""".format(vl

    cursor.execute(sql)
    print(sql)

conexao.commit()
```

```
 update contatos set telefone = null where codigo = 39;
 update contatos set telefone = null where codigo = 40;
 update contatos set telefone = null where codigo = 55;
```

In [23]:
```
cursor.execute(""" select * from contatos where telefone is null""")

cursor.fetchall()
```

Out[23]:
```
[(39, 'Paul Snyder', None),
 (40, 'Patrick Patterson', None),
 (55, 'Paul Walker', None)]
```

DELETE

In [24]:
```
cursor.execute(""" select codigo from contatos where telefone is null; """)

valores = cursor.fetchall()
```

In [25]:
```
valores
```

Out[25]:
```
[(39,), (40,), (55,)]
```

In [26]:
```
for vl in valores:
    sql = """ delete from contatos where codigo = {0}; """.format(vl[0])
    cursor.execute(sql)

conexao.commit()
```

In [27]:
```
cursor.execute(""" select * from contatos where telefone is null; """)

cursor.fetchall()
```

Out[27]:
```
[]
```

CRIANDO UM BACKUP

In [28]:
```
with io.open('banco_backup.sql', 'w') as f:
    for linha in conexao.iterdump():
        f.write('%s\n' % linha)
```

In [29]:
```
os.listdir()
```

Out[29]:
```
['.ipynb_checkpoints',
 '2020 alunos.csv',
 '2020 Nomes.txt',
 'BANCO DE DADOS.ipynb',
 'banco.db',
 'banco_backup.sql',
 'enve',
 'ESTATÍSTICA INICIAL.ipynb',
 'MANIPULAÇÃO DE ARQUIVOS.ipynb',
 'requerimentos.txt']
```

FINALIZANDO A CONEXÃO COM BANCO DE DADOS

In [30]:
```
conexao.close()
```

EXCLUIR O BANCO DE DADOS

In [31]:
```
os.remove('banco.db')
```

In [32]:
```
os.listdir()
```

Out[32]:
```
['.ipynb_checkpoints',
 '2020 alunos.csv',
 '2020 Nomes.txt',
 'BANCO DE DADOS.ipynb',
 'banco_backup.sql',
 'enve',
 'ESTATÍSTICA INICIAL.ipynb',
 'MANIPULAÇÃO DE ARQUIVOS.ipynb',
 'requerimentos.txt']
```

RESTAURANDO UM BACKUP

In [36]:
```
conexao = sqlite3.connect('banco.db')
cursor= conexao.cursor()

f = io.open('banco_backup.sql', 'r')
sql = f.read()

cursor.executescript(sql)
```

Out[36]:
```
<sqlite3.Cursor at 0x2b17d082650>
```

In [37]:
```
cursor.execute(""" select * from contatos; """)

cursor.fetchall()
```

Out[37]:
```
[(1, 'Jennifer Rose', '+1-925-387-5486x2356'),
 (2, 'Maureen Martin', '(933)470-5036x533'),
 (3, 'Michael Wallace', '112-868-3272x7086'),
 (4, 'Mr. Charles Wade Jr.', '001-033-345-0282'),
 (5, 'Gina Franco', '996-226-4289'),
 (6, 'Jennifer Johnson', '959-006-9819x62702'),
 (7, 'Christine Winters', '236-278-2103x55206'),
 (8, 'Andrew Ballard', '+1-170-687-9212x53745'),
 (9, 'Bryan Sims', '(924)778-1097x299'),
 (10, 'Matthew Mejia', '3383603748'),
 (11, 'Mr. Cody Mills', '(089)567-0448x3552'),
 (12, 'Debbie Moore', '(210)500-3376x51494'),
 (13, 'Jessica Bond', '+1-090-789-8219'),
 (14, 'Kenneth Chavez', '720-373-2558'),
 (15, 'Laura Gibson', '001-955-559-9226x33312'),
 (16, 'Andrea Rodriguez', '4001156107'),
 (17, 'Susan Gutierrez', '147.793.4190'),
 (18, 'James Williams', '766.847.9138x690'),
 (19, 'Nicole Johnson', '+1-477-126-9317x383'),
 (20, 'Miguel Bryant', '(938)239-4928x2001'),
 (21, 'Nicole Garcia', '448.694.3621x0799'),
 (22, 'John Ramirez', '+1-135-009-8562x751'),
 (23, 'Kristin Wilson', '+1-087-091-5337x5177'),
 (24, 'Travis Howard', '3705152245'),
 (25, 'Amanda Lang', '+1-715-601-2613x5176'),
 (26, 'Jennifer Ross', '001-992-193-3435x51626'),
 (27, 'Doris Guzman', '929.491.3048x6119'),
 (28, 'Spencer Wells', '300.005.0991x157'),
 (29, 'John Sanders', '9057907061'),
 (30, 'Donald Brown', '(998)529-4476x24635'),
 (31, 'Robert Williams', '708-199-7149x08874'),
 (32, 'Jennifer Faulkner', '(090)526-9893'),
 (33, 'Julie Lane', '799-207-2130x895'),
 (34, 'Lee Dominguez', '+1-537-907-6766'),
 (35, 'Erica Figueroa', '623-365-0885'),
 (36, 'Christina Perry', '001-164-620-2163x80058'),
 (37, 'Megan Martinez', '4941157941'),
 (38, 'Tammy Drake', '444.999.2194x400'),
 (41, 'Julie Roberts', '(254)213-1267x6007'),
 (42, 'Amy Hammond', '001-653-657-1231x5541'),
 (43, 'Jennifer Mccarty', '394.483.1723'),
 (44, 'Michael Scott', '345.044.7958'),
 (45, 'Jacob Sanchez', '700-529-4643'),
 (46, 'Christine Moyer', '001-870-404-1174x129'),
 (47, 'David Scott', '+1-167-263-6134x146'),
 (48, 'Robert Thornton', '337.861.3890x281'),
 (49, 'Craig Braun', '8270808393'),
 (50, 'Lisa Lowe', '418.186.3474'),
 (51, 'Sara Lewis', '364.329.0652'),
 (52, 'Megan Rodriguez', '609.975.1255'),
 (53, 'Rhonda Cohen', '001-525-076-4589x959'),
 (54, 'Jessica Hanson', '(167)222-7901'),
 (56, 'John Chavez', '+1-239-531-9632x143'),
 (57, 'Hunter Mcbride', '300.519.7207'),
 (58, 'Linda Lewis', '410-762-1593x9056'),
 (59, 'Janice Davis', '+1-618-531-0125'),
 (60, 'James Collins', '8328787634'),
 (61, 'Amber Baker', '941-379-4795'),
```

```
(62, 'Kevin Diaz', '068-434-6947'),
(63, 'Darlene Strickland', '850-077-4674x31946'),
(64, 'Travis Hunt', '(505)985-1602'),
(65, 'William Austin', '6292677244'),
(66, 'Cheyenne Short', '(837)350-5949'),
(67, 'Mario Thomas', '690-353-5774x044'),
(68, 'Shannon Miranda', '483.960.7977x6562'),
(69, 'Raymond Maynard', '+1-227-609-2920x834'),
(70, 'Billy Flores', '(720)318-1085x6621'),
(71, 'Tiffany Neal', '763.992.9637x77614'),
(72, 'Susan Richardson', '2410873608'),
(73, 'Deborah Calhoun', '001-157-075-2953x307'),
(74, 'John Fuller', '7676069434'),
(75, 'Diana Fisher', '442.329.8416x758'),
(76, 'Victoria Green', '(184)162-8422'),
(77, 'Cameron Torres', '035.320.5627x074'),
(78, 'Alex Reyes', '441.857.4545'),
(79, 'Timothy Clark', '810.915.4419'),
(80, 'Jeffrey Howe', '346.579.7957'),
(81, 'James Ortiz', '6092837729'),
(82, 'Abigail Rivera', '594.389.4734x7037'),
(83, 'Anthony Flores', '(505)612-8042x612'),
(84, 'Gary Williams', '003.751.3457'),
(85, 'Tiffany Davis', '100-119-0414x603'),
(86, 'Erik Perry', '001-455-370-5870'),
(87, 'Chris Odonnell', '(028)604-3623'),
(88, 'Kathy Nolan', '(940)203-7562x5730'),
(89, 'Kyle Bryant', '698-424-6959x30427'),
(90, 'Chelsea Graham', '0106031244'),
(91, 'Cindy Taylor', '396-035-5398x12133'),
(92, 'Brittany Richards', '6320000926'),
(93, 'Brian Rodriguez', '(484)360-4615x1628'),
(94, 'Jason Lopez', '(223)297-1817x88340'),
(95, 'Anna Rhodes', '2411582839'),
(96, 'Alex Charles', '333.476.5533'),
(97, 'Lisa Morris', '281-846-2163x721'),
(98, 'Craig Logan', '580-527-9725x654'),
(99, 'Thomas Green', '031.793.5224x43632')]
```

FINALIZANDO A CONEXÃO COM BANCO DE DADOS

In [38]:
```
conexao.close()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: