Практическая работа (вариант 22). Сергушов Павел ПМ22-4

BBO,fĮ [1]:
```
!pip install psycopg2
```

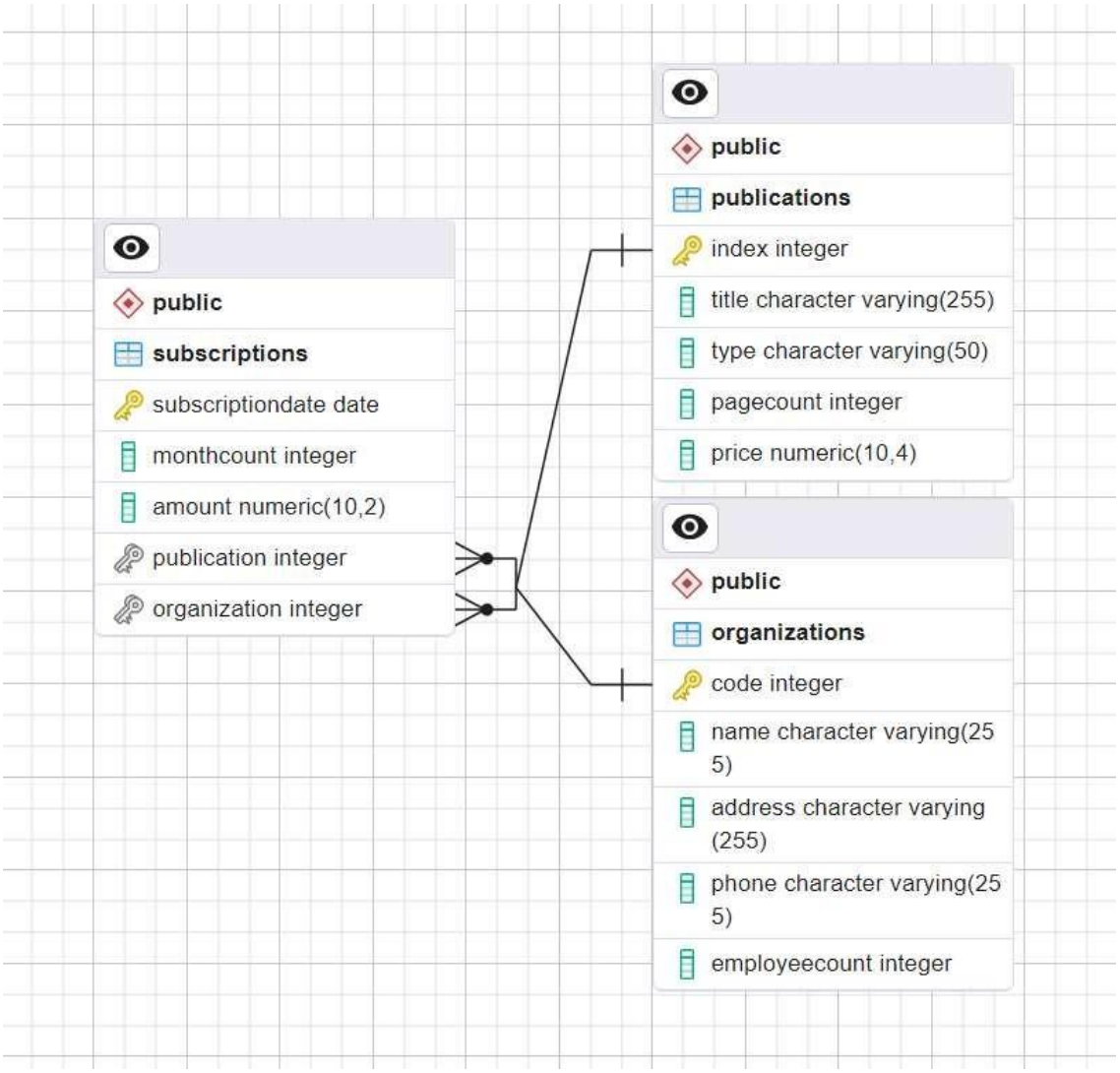
```
Collecting psycopg2
  Obtaining dependency information for psycopg2 from https://files.python
hosted.org/packages/37/2c/5l33dd3l83a3bd8237l569f0dd783e6927672de7e67lb27
  Downloading psycopg2-2.9.9-cp311-cp311-win_amd64.whl.metadata (4.5 kB)
Downloading psycopg2-2.9.9-cp311-cp311-win_amd64.whl (1.2 MB)
   ---------------------------------------- 0.0/1.2 MB ? eta -:---:---
   ---------------------------------------- 0.0/1.2 MB ? eta -:---:---
   ---------------------------------------- 0.0/1.2 MB ? eta -:---:---
   - -------------------------------------- 0.0/1.2 MB 326.8 kB/s eta 0:0
0:04
   ---- ----------------------------------- 0.1/1.2 MB 581.0 kB/s eta 0:0
0:02
   ------- -------------------------------- 0.2/1.2 MB 706.2 kB/s eta 0:0
0:02
   ---------- ----------------------------- 0.3/1.2 MB 1.2 MB/s eta 0:00:
01
                                     -      -- 0.4/1.2 MB 1.4 MB/s eta 0:00:
01
   ------------------------- ---------------- 0.6/1.2 MB 1.8 MB/s eta 0:00:
01
                                     -      -- 0.8/1.2 MB 2.1 MB/s eta 0:00:
01
   ---------------------------------------- ---- 1.1/1.2 MB 2.4 MB/s eta 0:00:
01
   ---------------------------------------- 1.2/1.2 MB 2.5 MB/s eta 0:00:
00
Installing collected packages: psycopg2
Successfully installed psycopg2-2.9.9
```

1. ВНачапе cosдайTe caMy 6aay даННblX На cepBepe PostgreSQL (С nOMOщbю pgAdmin или другого клиеНТа). В cBoeй nporpaMMepeали3уйTe ØyНхцию, KoTopasi co3дaeT cTpyKTypy 6asbl ,0,ElHHbiX — cosдaeT Bce указalH bie Та6лиlJ,bI (HEl3BćlHHR Tćl6nnç и CTOfiбLțOB 3aдайTe лaTHHCKklMи 6yxBaMи (T l6nHLț - BO MHожecTBeH oM qøcne), 6es nробenoB, oCMыcле ыe) c nOMOLL\blÕ KOM lH,Ețbl CREATE, предycMoTpиTe целосTHOCTb ,Eț lHHbIX (cOsдайTe eo6xOдиMыe ключи).

BBOq [6]:

```
import psycopg2
from psycopg2 import sql

def create_tables():
    # TiOdKJñO eHue K 6aae baHHblX
    conn = psycopg2.connect(
        dbname="publications",
        user="postgres",
        password="postgres",
        host="localhost",
        port=5432

    cur = conn.cursor()
    # flOâKJtio eHue ycinaHo8neHo
    commands = (

        CREATE TABLE Organizations (
            Code INTEGER PRIMARY KEY,
            Name VARCHAR(255),
            Address VARCHAR(255),
            Phone VARCHAR(255),
            EmployeeCount INTEGER



        CREATE TABLE Publications (
            Index INTEGER PRIMARY KEY,
            Title VARCHAR(255),
            Type VARCHAR(50),
            PageCount INTEGER,
            Price NUMERIC(10, 4)



        CREATE TABLE Subscriptions (
            SubscriptionDate DATE PRIMARY KEY,
            MonthCount INTEGER,
            Amount NUMERIC(10, 2),
            Publication INTEGER,
            Organization INTEGER,
            FOREIGN KEY (Publication) REFERENCES Publications(Index),
            FOREIGN KEY (Organization) REFERENCES Organizations(Code)




    for command in commands:
        cur.execute(command)

    conn.commit()
    cur.close()
    conn.close()
    print("Done! ")

create_tables()
```

Done!



2. Напишите функции, которые добавляют данные в таблицы. Для каждой таблицы напишите два варианта функций: первый вариант функции добавляет одну строку данных, второй вариант – добавляет все указанные в задании строки. В каждую из функций данные (т.е. одна строка или набор строк) передаются в качестве параметров. Т.е. и в запрос, который выполняется с помощью execute (executemany), эти данные тоже приходят в качестве параметров (нужно использовать параметризованные запросы INSERT).

BBOQ [7]:

```
def add_organization(cur, code, name, address, phone, employee_count):
    cur.execute(
        "INSERT INTO Organizations (Code, Name, Address, Phone, EmployeeCoc
        (code, name, address, phone, employee_count)


def add_organizations(cur, organizations_data):
    cur.executemany(
        "INSERT INTO Organizations (Code, Name, Address, Phone, EmployeeCoc
        organizations_data

def add_publication(cur, index, title, type, page_count, price):
    cur.execute(
        "INSERT INTO Publications (Index, Title, Type, PageCount, Price) V1
        (index, title, type, page_count, price)


def add_publications(cur, publications_data):
    cur.executemany(
        "INSERT INTO Publications (Index, Title, Type, PageCount, Price) V1
        publications_data

def add_subscription(cur, subscription_date, month_count, amount, publicat.
    cur.execute(
        "INSERT INTO Subscriptions (SubscriptionDate, MonthCount, Amount, f
        (subscription_date, month_count, amount, publication, organization'


def add_subscriptions(cur, subscriptions_data):
    cur.executemany(
        "INSERT INTO Subscriptions (SubscriptionDate, MonthCount, Amount, f
        subscriptions_data
```

BBO,£t [9] :

```
conn = psycopg2.connect(
    dbname="publications",
    user="postgres",
    password="postgres",
    host="localhost",
    pont =5432

print("HoQклюueHne  усТаНоВлеНо! ")
cur = conn.cursor()

organizations_data = [
    (1, '000 Мобп Ips', 'Qe абрисТоВ, 12', '745504', 100),
    (2, 'ООО ОпТпс', 'feHnHrpaqc uM np-T, 62', '771619', 50),
    (3, 'ООО РосТагроа спо', 'КуТу3ОВСКпп np-T, 26', None, 30),
    (5,' ООО СТропДОм', ЛеНпНа, 3','283455',40),
    (7,'000 ИНТергеимС','FapapиНа,  5',  None,10)


publications_data = [
    (1, 'НаНОрама',  'raaeTa',  30, 12.0000),
    (2, 'Tezecemb',  'ra3eTa',  40, 15.0000),
    (6, 'ЯрпарКа', None, 100, 10.0000)


subscriptions_data = [
    ('2020-01-06', 5, 0.02, 1, 3),
    ('2020-01-08', 7, 0.12, 1, 7),


cur = conn.cursor()

add_organizations(cur, organizations_data)
add_publications(cur, publications_data)
add_subscriptions(cur, subscriptions_data)


conn.commit()
cur.close()
conn.close()
```

ПОQКпюueHne усТаНоВzeНо!

	subscriptiondate [PK] date	monthcount integer	amount numeric (10,2)	publication integer	organization integer
1	2020-01-06	5	0.02	1	3
2	2020-01-08	7	0.12	1	7

3. НапишиТе фуНКцИи, КоТорые ВыполняfОТ ВыборКи иЗ ТаблиIJ,. Дf я Кахс,0,Ой Та6f vIL\bI ВариантЫ функций должНЫ бblTb TaKrIMH: ВЫбраТЬ Все даННЬie из TEIбпHIJ,Ы; ВыбраТь ,qанные, В КоТорых значение(-я) поля(-ей) раВНО(-ы) За,qаННоМу(-ыМ) — параМеТризоВаННЬie запросЬi, заданНое(-Ыe) значеНие(-я) передае(ю)ТСЯ В фуНКцНЇО В качесТВе параМеТра(-ОВ); ВЬібраТЬ д?tННbie, получеННЬie после соедиНения ТаблиIи„ В запросе должен бьITЬ ХОТЯ бы одиН параМеТр. Все

napaMeTpbi, KoTopbie присуТсТВуюТ В запросах, должНЫ передаВаТЬСя В фуНхцию при ее ВЬізоВе.

ВВОQ [10]:
```
conn = psycopg2.connect(
    dbname="publications”,
    user=”postgres",
    password="postgres”,
    host=”localhost",
    port=5432

print(”Нод зюеНue ycTaHoBzeHo!")
cur = conn.cursor()
```

HOQKnюueHne ycTaHoBzeHo!

QflFl TfiIбппцы Organizations

ВВО,£{ [12]:
```
def get_all_organizations(cur):
    cur.execute("SELECT * FROM Organizations“)
    return cur.fetchall()
def get_organization_by_name(cur, name):
    cur.execute("SELECT * FROM Organizations WHERE Name = %s”, (name,))
    return cur.fetchall()
def get_organizations_with_subscriptions(cur, organization_code):
    cur.execute("""
    SELECT o.Name, o.Address, s.SubscriptionDate, s.Amount
    FROM Organizations o
    JOIN Subscriptions s ON o.Code = s.Organization
    WHERE o.Code = %s
    """, (organization_code,))
    return cur.fetchall()
```

ВВОД [13]:
```
get_all_organizations(cur)
```

Out[13]:
```
[(1, '000 Моби Ips', 'Qe абрпсТОВ, 12', '745504', 100),
 (2, '000 OuTuc', 'feHnHrpaflc uñ np-T, 62', '771619', 50),
 (3, '000 POcTarpoaKCno', 'KyTyaoBCKuñ np-T, 26', None, 30),
 (s,    ООО СТройДОl'»', 'fleHnHd, 3 ,   283455', 40) ,
 (7,   ООО ННТеpейfnC',   rar apnHd, 5 , None, 10) ]
```

ВВО,£{ [14] :
```
get_organization_by_name(cur, 'ООО Моби Ips')
```

Out[14]:
```
[(1, '000 Моби Ips', 'Qe абрпсТОВ, 12', '745504', 100)]
```

ВВО,Д [17]:
```
get_organizations_with_subscriptions(cur, 3)
```

Out[17]:
```
[('ООО РосТагроаКСпо',
  'KyTyaOBCKuи пр-Т, 26',
  datetime.date(2020, 1, 6),
  Decimal('0.02'))]
```

,Цf1R Таблицы Publications

ВВОД [18]:
```
def get_all_publications(cur):
    cur.execute("SELECT * FROM Publications")
    return cur.fetchall()
def get_publications_by_type(cur, type):
    cur.execute("SELECT * FROM Publications WHERE Type = %s", (type,))
    return cur.fetchall()
def get_publications_with_subscriptions(cur, publication_index):
    cur.execute("""
    SELECT p.Title, p.Type, s.SubscriptionDate, s.Amount
    FROM Publications p
    JOIN Subscriptions s ON p.Index = s.Publication
    WHERE p.Index = %s
    """, (publication_index,))
    return cur.fetchall()
```

ВВО,Д [19]:
```
get_all_publications(cur)
```

Out[19]:
```
[(1, 'ПаНОрапа', 'raaeTa', 30, Decimal('12.0000')),
 (2, 'Tenecemb', 'ra3eTa', 40, Decimal('15.0000')),
 (6, 'Ярпарса', None, 100, Decimal('10.0000'))]
```

ВВОQ [20]:
```
get_publications_by_type(cur, 'ra3eTa')
```

Out[20]:
```
[(1, 'НаНорама', 'ra3era', 30, Decimal('12.0000')),
 (2, 'ТелесемЬ', 'raaeTa', 40, Decimal('15.0000'))]
```

ВВОД [23]:
```
get_publications_with_subscriptions(cur, 1)
```

Out[23]:
```
[('НаНорама', 'ra3eTa', datetime.date(2020, 1, 6), Decimal('0.02')),
 ('НаНорама', 'raaeTa', datetime.date(2020, 1, 8), Decimal('0.12'))]
```

Дf1Я Т?Ібгіицы Subscriptions

ВВО,Ц [24]:
```
def get_all_subscriptions(cur):
    cur.execute("SELECT * FROM Subscriptions")
    return cur.fetchall()
def get_subscriptions_by_date(cur, subscription_date):
    cur.execute("SELECT * FROM Subscriptions WHERE SubscriptionDate = %s",
    return cur.fetchall()
def get_detailed_subscriptions(cur, subscription_date):
    cur.execute("""
    SELECT s.SubscriptionDate, s.MonthCount, s.Amount, p.Title, o.Name
    FROM Subscriptions s
    JOIN Publications p ON s.Publication = p.Index
    JOIN Organizations o ON s.Organization = o.Code
    WHERE s.SubscriptionDate = %s
    """, (subscription_date,))
    return cur.fetchall()
```

ВВОД [25]:
```
get_all_subscriptions(cur)
```

Out[25]:
```
[(datetime.date(2020, 1, 6), 5, Decimal('0.02'), 1, 3),
 (datetime.date(2020, 1, 8), 7, Decimal('0.12'), 1, 7)]
```

BBO,£t [26] :

```
get_subscriptions_by_date(cur, '2020-01-06')
```

Out[26]:

```
[(datetime.date(2020, 1, 6), 5, Decimal('0.02'), 1, 3)]
```

BBO,£[[27] :

```
get_detailed_subscriptions(cur, '2020-01-06')
```

Out[27]:

```
[(datetime.date(2020, 1, 6),
  5,
  Decimal('0.02'),
  'HaHO alMa',
  '000 PocTarpoa cno')]
```

BBO,£{ [28] :

```
def update_organization(cur, code, new_phone, new_address):
    cur.execute(
        "UPDATE Organizations SET Phone = %s, Address = %s WHERE Code = %s'
        (new_phone, new_address, code)

def update_publication(cur, index, new_price, new_page_count):
    cur.execute(
        "UPDATE Publications SET Price = %s, PageCount = %s WHERE Index = %
        (new_price, new_page_count, index)

def update_subscription(cur, subscription_date, new_month_count, new_amoun1
    cur.execute(
        "UPDATE Subscriptions SET MonthCount = %s, Amount = %s WHERE Subscr
        (new_month_count, new_amount, subscription_date)
```

BBO,Q [32] :

```
update_organization(cur, 1, '999888777', 'Ko CO OwbCKaa, 123')
get_all_organizations(cur)
```

Out[32]:

```
[(2, '000 0nTuC', 'fleHnHrpaqc nñ np-T, 62', '771619', 50),
 (3, '000 POcTarpoaKCno', 'KyTyaOBCKzH np-T, 26', None, 30),
 (5, ' 000 CTpoñflO ', 'fleHnHa, 3', '283455', 40),
 (7, '000 HHTepreu C', 'rarapnHa, 5', None, 10),
 (1, '000 Mo6u Ips', 'KOmcOmOwbCKan, 123', '999888777', 100)]
```

BBO,Zt [33] :

```
update_publication(cur, 1, 13.00, 32)
get_all_publications(cur)
```

Out[33]:

```
[(2, 'Tenecenb', 'raaeTa', 40, Decimal('15.0000')),
 (6, 'flpnapKa', None, 100, Decimal('10.0000')),
 (1, 'naHopama', 'ra3eTa', 32, Decimal('13.0000'))]
```

BBO,Zt [34] :

```
update_subscription(cur, '2020-01-06', 6, 0.09)
get_all_subscriptions(cur)
```

Out[34]:

```
[(datetime.date(2020, 1, 8), 7, Decimal('0.12'), 1, 7),
 (datetime.date(2020, 1, 6), 6, Decimal('0.09'), 1, 3)]
```

BBO,Zt [35] :

```
conn.commit()
```

HanuLuuTe c|syнКцНи для Kax‹,rtOй Таблицы, КоТорые ВыпогіняІ€іТ а) удалеНие даННых пО HezoToроМу услоВию — ЗНачеНия для услоВий передаВайТе В хачесТВе параМеТроВ В фуНхг\riiO; 6) удапеНие Всех даННых. FIредусМоТриТе ВыпогіНеНие усгіоВий сохраНеНия целосТНоСТи ,qaHHbIX.

ВВОQ [38]:
```
def delete_organization_by_code(cur, code):
    cur.execute(”DELETE FROM Organizations WHERE Code = %s”, (code,))
def delete_all_organizations(cur):
    cur.execute("DELETE FROM Organizations")
def delete_publication_by_index(cur, index):
    cur.execute("DELETE FROM Publications WHERE Index = %s”, (index,))
def delete_all_publications(cur):
    cur.execute("DELETE FROM Publications")
def delete_subscription_by_date(cur, subscription_date):
    cur.execute("DELETE FROM Subscriptions WHERE SubscriptionDate = %s”, ('
def delete_all_subscriptions(cur):
    cur.execute("DELETE FROM Subscriptions")
```

ВВОД [39] :
```
delete_organization_by_code(cur, 1)
conn.commit()
get_all_organizations(cur)
```

Out[39] :
```
[(2, '000 0nTuC', 'fleHnHrpaqc nñ np-T, 62', '771619', 50),
 (3, '000 POcTarpoaKcno', 'KyTyaOBCKzH np-T, 26', None, 30),
 (5, ' 000 CTpoñQom', 'fleHnHa, 3', '283455', 40),
 (7, '000 HHTepreu C', 'rarapnHa, 5', None, 10)]
```

ВВОД [40] :
```
delete_publication_by_index(cur, 2)
conn.commit()
get_all_publications(cur)
```

Out[40]:
```
[(6, 'ЯрпарКа', None, 100, Decimal('10.0000')),
 (1, 'ПаНорама', 'raЗeTa', 32, Decimal('13.0000'))]
```

ВВО,f{ [48] :
```
delete_all_subscriptions(cur)
conn.commit()
get_all_subscriptions(cur)
```

Out[48]:
```
[]
```