Національний технічний університет України
“Київський політехнічний інститутімені Ігоря Сікорського”
Факультет прикладної математики
Кафедра системного програмування і спеціалізованих комп’ютерних систем

Лабораторна робота №2

з дисципліни

“Бази данних і засоби управління ”

Тема: “ **Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL**

“

Виконав: Воєводін Ілля Петрович
Студент групи КВ-84
Перевірив(ла):_____

Київ – 2020

Постановка задачі

Загальне завдання роботи полягає у наступному:

1. Реалізувати функції внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
2. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
3. Забезпечити реалізацію пошуку за декількома атрибутами з двох та більше сутностей одночасно: для числових атрибутів – у рамках діапазону, для рядкових – як шаблон функції LIKE оператора SELECT SQL, для логічного типу – значення True/False, для дат – у рамках діапазону дат.
4. Програмний код виконати згідно шаблону MVC (модель-подання-контролер).

Деталізоване завдання:

1. Забезпечити можливість уведення/редагування/вилучення даних у таблицях бази даних з можливістю контролю відповідності типів даних атрибутів таблиць (рядків, чисел, дати/часу). Для контролю пропонується два варіанти: контроль при введенні (валідація даних) та перехоплення помилок (try..except) від сервера PostgreSQL при виконанні відповідної команди SQL. Особливу увагу варто звернути на дані таблиць, що мають зв’язок 1:N. При цьому з боку батьківської таблиці необхідно контролювати **вилучення** рядків за умови наявності даних у підлеглій таблиці. З точки зору підлеглої таблиці варто контролювати наявність відповідного рядка у батьківській таблиці при виконанні **внесення** нових даних. Унеможливити виведення програмою системних помилок на екрані шляхом їх перехоплення і адекватної обробки. Внесення даних виконується користувачем у консольному вікні програми.
2. Забезпечити можливість автоматичної генерації великої кількості даних у таблицях за допомогою вбудованих у PostgreSQL функцій роботи з псевдовипадковими числами. Дані мають бути згенерованими **не мовою програмування, а відповідним SQL-запитом**!

   Кількість даних для генерування має вводити користувач з клавіатури. Для тесту взяти 100 000 записів для однієї-двох таблиць.

   Особливу увагу слід звернути на відповідність даних вимогам зовнішніх ключів з метою уникнення помилок порушення обмежень цілісності (foreign key).

1. Для реалізації пошуку необхідно підготувати 3 запити, що включають дані з декількох таблиць і фільтрують рядки за 3-4 атрибутами цих таблиць. Забезпечити можливість уведення конкретних значень констант для фільтрації з клавіатури користувачем. Крім того, після виведення даних необхідно вивести час виконання запиту у мілісекундах. Перевірити швидкодію роботи запитів на попередньо згенерованих даних.
1. Програмний код організувати згідно шаблону Model-View-Controller(MVC). При цьому модель, подання та контролер мають бути реалізовані у окремих файлах. Для доступу до бази даних використовувати **лише мову SQL** (без ORM).

Результати

Вимоги до пункту №1 деталізованого завдання:

- ілюстрації обробки виняткових ситуацій (помилок) при уведенні/вилучення даних;

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
2
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
qwerty
Wrong input, try again!
Table name 'qwerty' is incorrect

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
10
Range error
10 out of range (1,7)

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
5
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
email
====== email TABLE ======
ID:
['email_id', 'user_id', 'email_adress', 'email_type']
(1, 6, '545UR', 'YA')
(2, 8, '966TJ', 'LM')
(3, 2, '45HB', 'MD')
(4, 8, '719XR', 'YT')
(5, 8, '978OG', 'KM')
(6, 8, '216JF', 'JS')
(7, 3, '233CR', 'MP')
(8, 6, '536LJ', 'QQ')
(9, 8, '969SG', 'GL')
(10, 3, '557IN', 'MG')
(11, 8, '350RY', 'MV')
Choose key and value to delete:

Input key:
 folders_id
 Input value:
 2
 Key name error
 Key name folders_id doesn't exist

 Process finished with exit code 0
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
1
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
users
====== users TABLE ======
ID:
['user_id', 'username', 'phone_number']
(2, 'UI', 390855)
(3, 'XL', 27298)
(5, 'EL', 600803)
(6, 'AE', 684470)
(7, 'AL', 527326)
(8, 'BI', 751051)
(1, 'GG', 12346)
(10, 'GH', 12346)
(9, 'XT', 621292)
Choose key and value for search:

Input key:
username
Input value:
AA
Table doesn't have username=AA
Can't read "{'username': 'AA'}" because it's not stored

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
3
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
messages
====== messages TABLE ======
ID:
['messages_id', 'read_or_not', 'message_name', 'message_date']
(1, True, 'LFSY', datetime.date(2018, 4, 21))
(2, False, 'FIFF', datetime.date(2018, 7, 8))
(3, False, 'YYUS', datetime.date(2019, 9, 22))
(4, False, 'YVXH', datetime.date(2019, 5, 11))
(5, False, 'UKWV', datetime.date(2019, 10, 23))
(6, True, 'HXPQ', datetime.date(2018, 2, 23))
(7, True, 'ICUR', datetime.date(2019, 4, 11))
(8, True, 'GLMM', datetime.date(2019, 3, 3))
(9, False, 'AAEE', datetime.date(2018, 7, 16))
(10, True, 'ABCD', datetime.date(2020, 7, 7))
Input your data for all keys to create item:
['messages_id', 'read_or_not', 'message_name', 'message_date']
For key 1 Value =1
For key 2 Value =False
For key 3 Value =OOOO
For key 4 Value =2019,5,13
Value error, item already exists
ID: "1" already stored!

Process finished with exit code 0
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
4
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
folders
====== folders TABLE ======
ID:
['folders_id', 'email_id', 'folder_name', 'folder_type']
(1, 3, 'S417H', 'UE')
(2, 8, 'U3N', 'LG')
(3, 3, 'F503M', 'UL')
(4, 4, 'E937B', 'MR')
(5, 2, 'B711G', 'WP')
(6, 2, 'E181H', 'FV')
(7, 4, 'M934M', 'UN')
(8, 4, 'N530A', 'EG')
(9, 5, 'O549H', 'QM')
Choose key and value to update:

Input key:
email_id
Input value:
3
Input new value:
QWE
Value type error
Type of argument QWE is not the same as key type

Process finished with exit code 0
```

- ілюстрації валідації даних при уведенні користувачем.

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
1
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
folders
====== folders TABLE ======
ID:
['folders_id', 'email_id', 'folder_name', 'folder_type']
(1, 3, 'S417H', 'UE')
(2, 8, 'U3N', 'LG')
(3, 3, 'F503M', 'UL')
(4, 4, 'E937B', 'MR')
(5, 2, 'B711G', 'WP')
(6, 2, 'E181H', 'FV')
(7, 4, 'M934M', 'UN')
(8, 4, 'N530A', 'EG')
(9, 5, 'O549H', 'QM')
Choose key and value for search:

Input key:
folder_type
Input value:
MR

Search result:
(4, 4, 'E937B', 'MR')

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
2
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
email
All values from table:
====== email TABLE ======
ID:
['email_id', 'user_id', 'email_adress', 'email_type']
(1, 2, '372BW', 'GA')
(2, 1, '338BV', 'CL')
(3, 7, '131LB', 'GG')
(4, 8, '381YK', 'JI')
(5, 3, '623AR', 'PY')
(6, 8, '75FI', 'KR')
(7, 3, '521DP', 'LP')
(8, 7, '261RL', 'OJ')
(9, 4, '947VA', 'BO')

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
3
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
users
====== users TABLE ======
ID:
['user_id', 'username', 'phone_number']
(1, 'QI', 88055)
(2, 'PP', 113298)
(3, 'VT', 295052)
(4, 'XI', 659761)
(5, 'BP', 613534)
(6, 'GG', 474651)
(7, 'YE', 573121)
(8, 'YF', 549064)
(9, 'VP', 727258)
Input your data for all keys to create item:

['user_id', 'username', 'phone_number']
For key 1 Value =10
For key 2 Value =AB
For key 3 Value =234567
====== users TABLE ======
ID:
['user_id', 'username', 'phone_number']
(10, 'AB', 234567)
(1, 'QI', 88055)
(2, 'PP', 113298)
(3, 'VT', 295052)
(4, 'XI', 659761)
(5, 'BP', 613534)
(6, 'GG', 474651)
(7, 'YE', 573121)
(8, 'YF', 549064)
(9, 'VP', 727258)

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
4
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
messages
====== messages TABLE ======
ID:
['messages_id', 'read_or_not', 'message_name', 'message_date']
(1, True, 'LFSY', datetime.date(2018, 4, 21))
(2, False, 'FIFF', datetime.date(2018, 7, 8))
(3, False, 'YYUS', datetime.date(2019, 9, 22))
(4, False, 'YVXH', datetime.date(2019, 5, 11))
(5, False, 'UKWV', datetime.date(2019, 10, 23))
(6, True, 'HXPQ', datetime.date(2018, 2, 23))
(7, True, 'ICUR', datetime.date(2019, 4, 11))
(8, True, 'GLMM', datetime.date(2019, 3, 3))
(9, False, 'AAEE', datetime.date(2018, 7, 16))
(10, True, 'ABCD', datetime.date(2020, 7, 7))
Choose key and value to update:


Input key:
message_date
Input value:
2020,7,7
Input new value:
2010,10,20
====== messages TABLE ======
ID:
['messages_id', 'read_or_not', 'message_name', 'message_date']
(1, True, 'LFSY', datetime.date(2018, 4, 21))
(2, False, 'FIFF', datetime.date(2018, 7, 8))
(3, False, 'YYUS', datetime.date(2019, 9, 22))
(4, False, 'YVXH', datetime.date(2019, 5, 11))
(5, False, 'UKWV', datetime.date(2019, 10, 23))
(6, True, 'HXPQ', datetime.date(2018, 2, 23))
(7, True, 'ICUR', datetime.date(2019, 4, 11))
(8, True, 'GLMM', datetime.date(2019, 3, 3))
(9, False, 'AAEE', datetime.date(2018, 7, 16))
(10, True, 'ABCD', datetime.date(2010, 10, 20))

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
5
Choose table and input it's name:
< users >, < email >, < folders >, <folders_messages >, < messages >
users
====== users TABLE ======
ID:
['user_id', 'username', 'phone_number']
(10, 'AB', 234567)
(1, 'QI', 88055)
(2, 'PP', 113298)
(3, 'VT', 295052)
(4, 'XI', 659761)
(5, 'BP', 613534)
(6, 'GG', 474651)
(7, 'YE', 573121)
(8, 'YF', 549064)
(9, 'VP', 727258)
Choose key and value to delete:


Input key:
user_id
Input value:
5
====== users TABLE ======
ID:
['user_id', 'username', 'phone_number']
(10, 'AB', 234567)
(1, 'QI', 88055)
(2, 'PP', 113298)
(3, 'VT', 295052)
(4, 'XI', 659761)
(6, 'GG', 474651)
(7, 'YE', 573121)
(8, 'YF', 549064)
(9, 'VP', 727258)

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
6
Choose table or all tables:
1. "users"
2. "email"
3. "folders"
4. "messages"
5. "folders_messages"
6. All tables
Input number to select...
2
Input rows number:
15
========= EMAIL TABLE =========
ID|U_ID|Adress|E_Type
(1, 7, '522PP', 'RD')
(2, 2, '772TY', 'CQ')
(3, 2, '548DX', 'TD')
(4, 3, '186DQ', 'LR')
(5, 2, '702SF', 'DW')
(6, 1, '178BY', 'TM')
(7, 10, '830XA', 'JB')
(8, 6, '684ID', 'RJ')
(9, 9, '626LR', 'AT')
(10, 8, '73IX', 'MB')
(11, 6, '834VI', 'OH')
(12, 10, '667VR', 'BT')
(13, 9, '736CI', 'YR')
(14, 3, '232EY', 'HE')

Process finished with exit code 0
```

Вимоги до пункту №2 деталізованого завдання:

- копії екрану (ілюстрації) з фрагментами згенерованих даних таблиць;

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range

Select option by number...
6
Choose table or all tables:
1. "users"
2. "email"
3. "folders"
4. "messages"
5. "folders_messages"
6. All tables
Input number to select...
6
Input rows number:
10
========= USERS TABLE =========
ID|Username|Phone_Num
(1, 'QI', 88055)
(2, 'PP', 113298)
(3, 'VT', 295052)
(4, 'XI', 659761)
(5, 'BP', 613534)
(6, 'GG', 474651)
(7, 'YE', 573121)
(8, 'YF', 549064)
(9, 'VP', 727258)
========= EMAIL TABLE =========
ID|U_ID|Adress|E_Type
(1, 2, '372BW', 'GA')
(2, 1, '338BV', 'CL')
(3, 7, '131LB', 'GG')
(4, 8, '381YK', 'JI')
(5, 3, '623AR', 'PY')
(6, 8, '75FI', 'KR')
(7, 3, '521DP', 'LP')
(8, 7, '261RL', 'OJ')
(9, 4, '947VA', 'BO')
========= FOLDERS TABLE =========
ID|E_ID|F_Name|F_Type
(1, 3, 'S417H', 'UE')
(2, 8, 'U3N', 'LG')
(3, 3, 'F503M', 'UL')
(4, 4, 'E937B', 'MR')
(5, 2, 'B711G', 'WP')
(6, 2, 'E181H', 'FV')
(7, 4, 'M934M', 'UN')
(8, 4, 'N530A', 'EG')
(9, 5, 'O549H', 'QM')
========= FOLDERS_MESSAGES TABLE =========
FoldersID|Messages_ID
(4, 5)
(3, 4)
(6, 2)
(7, 8)
(1, 2)
(8, 6)
(1, 5)
(4, 5)
(3, 1)
========= MESSAGES TABLE =========
ID|ReadOrNot|Name | Date
(1, True, 'LFSY', datetime.date(2018, 4, 21))
(2, False, 'FIFF', datetime.date(2018, 7, 8))
(3, False, 'YYUS', datetime.date(2019, 9, 22))
(4, False, 'YVXH', datetime.date(2019, 5, 11))
(5, False, 'UKWV', datetime.date(2019, 10, 23))
(6, True, 'HXPQ', datetime.date(2018, 2, 23))
(7, True, 'ICUR', datetime.date(2019, 4, 11))
(8, True, 'GLMM', datetime.date(2019, 3, 3))
(9, False, 'AAEE', datetime.date(2018, 7, 16))

Process finished with exit code 0
```

- копії SQL-запитів, що ілюструють генерацію при визначених вхідних параметрах.

```
def random_users(conn, rows):
    cur = conn.cursor()

    cur.execute("DELETE FROM users;")

    SQL = "INSERT INTO users (user_id, phone_number, username) VALUES (%s, %s, %s);"

    cur.execute("SELECT trunc(random()*(1000000) + 1)::int FROM generate_series(1,1000) ")
    rand_num = cur.fetchall()

    cur.execute("SELECT chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) "
                "FROM generate_series(1,1000)")
    rand_name = cur.fetchall()

    i = 1
    while i < rows:
        data = (i, rand_num[i], rand_name[i])
        cur.execute(SQL, data)
        i = i + 1
```

```
def random_email(conn, rows):
    cur = conn.cursor()

    cur.execute("DELETE FROM email;")

    SQL = "INSERT INTO email (email_id, user_id, email_adress, email_type) VALUES (%s, %s, %s, %s);"

    cur.execute("SELECT max(user_id) FROM users")
    usr_id_max = cur.fetchone()

    if (cur.rowcount == 0) or (usr_id_max is None):
        raise mvc_exc.ParentTabError(f"You can't create random data for email table if parent table doesn't exist")

    cur.execute(f"SELECT user_id FROM users")
    user_ids = cur.fetchall()
    cur.execute(f"SELECT trunc(random()*({usr_id_max[0]}-1) + 1) FROM generate_series(1,1000)")
    rand_id = cur.fetchall()

    i = 0
    for item in rand_id:
        f = False
        for keys in user_ids:
            if item == keys:
                f = True

        if f is False:
            rand_id[i] = usr_id_max[0]
        i = i+1

    cur.execute("SELECT trunc(random()*(1000))::int ||"
                "chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int)FROM generate_series(1,1000) ")
    rand_num = cur.fetchall()

    cur.execute("SELECT chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) "
                "FROM generate_series(1,1000)")
    rand_name = cur.fetchall()

    i = 1
    while i < rows:
        data = (i, rand_id[i], rand_num[i], rand_name[i])
        cur.execute(SQL, data)
        i = i + 1
```

```python
def random_folders(conn, rows):
    cur = conn.cursor()

    cur.execute("DELETE FROM folders;")

    SQL = "INSERT INTO folders (folders_id, email_id, folder_name, folder_type) VALUES (%s, %s, %s, %s);"

    cur.execute("SELECT max(email_id) FROM email")
    eml_id_max = cur.fetchone()

    cur.execute("SELECT email_id FROM email")
    eml_ids = cur.fetchall()

    if (cur.rowcount == 0) or (eml_id_max is None):
        raise mvc_exc.ParentTabError(f"You can't create random data for folders table if parent table doesn't exist")

    cur.execute(f"SELECT trunc(random()*({eml_id_max[0]}-1) + 1) FROM generate_series(1,1000)")
    rand_id = cur.fetchall()

    i = 0
    for item in rand_id:
        f = False
        for keys in eml_ids:
            if item == keys:
                f = True

        if f is False:
            rand_id[i] = eml_id_max[0]
        i = i + 1

    cur.execute("SELECT chr(trunc(65+random()*25)::int) || "
                "trunc(random()*(1000) + 1)::int || chr(trunc(65+random()*25)::int) FROM generate_series(1,1000) ")
    rand_name1 = cur.fetchall()

    cur.execute("SELECT chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) "
                "FROM generate_series(1,1000)")
    rand_name2 = cur.fetchall()

    i = 1
    while i < rows:
        data = (i, rand_id[i], rand_name1[i], rand_name2[i])
        cur.execute(SQL, data)
        i = i + 1
```

```python
def random_fm(conn, rows):
    cur = conn.cursor()

    cur.execute("DELETE FROM folders_messages;")

    SQL = "INSERT INTO folders_messages (folders_id, messages_id) VALUES (%s, %s);"

    cur.execute("SELECT max(messages_id) FROM messages")
    mes_id_max = cur.fetchone()

    cur.execute("SELECT messages_id FROM messages")
    mes_ids = cur.fetchall()

    cur.execute("SELECT max(folders_id) FROM folders")
    fold_id_max = cur.fetchone()

    cur.execute("SELECT folders_id FROM folders")
    fold_ids = cur.fetchall()

    if ((cur.rowcount == 0) or (mes_id_max is None)) or ((cur.rowcount == 0) or (fold_id_max is None)):
        raise mvc_exc.ParentTabError(f"You can't create random data for folders_messages table if parent table "
                                     f"doesn't exist")

    cur.execute(f"SELECT trunc(random()*({fold_id_max[0]}-1) + 1) FROM generate_series(1,1000)")
    rand_f_id = cur.fetchall()

    i = 0
    for item in rand_f_id:
        f = False
        for keys in fold_ids:
            if item == keys:
                f = True

        if f is False:
            rand_f_id[i] = fold_id_max[0]
        i = i + 1

    cur.execute(f"SELECT trunc(random()*({mes_id_max[0]}-1) + 1) FROM generate_series(1,1000)")
    rand_m_id = cur.fetchall()

    i = 0
    for item in rand_m_id:
        f = False
        for keys in mes_ids:
            if item == keys:
                f = True

        if f is False:
            rand_m_id[i] = mes_id_max[0]
        i = i + 1

    i = 1
    while i < rows:
        data = (rand_f_id[i], rand_m_id[i])
        cur.execute(SQL, data)
        i = i + 1
```

```python
def random_messages(conn, rows):
    cur = conn.cursor()

    cur.execute("DELETE FROM messages;")

    SQL = "INSERT INTO messages (messages_id, read_or_not, message_name, message_date) VALUES (%s, %s, %s, %s);"

    cur.execute("SELECT bool(random() > 0.5) FROM generate_series(1,1000)")
    rand_bool = cur.fetchall()

    cur.execute("SELECT chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int)"
                "|| chr(trunc(65+random()*25)::int) || chr(trunc(65+random()*25)::int) FROM generate_series(1,1000) ")
    rand_name = cur.fetchall()

    cur.execute("SELECT timestamp '2018-01-01 00:00:00' + random()*(timestamp '2020-01-01 00:00:00' "
                "- timestamp '2018-01-01 00:00:00') FROM generate_series(1,1000)")
    rand_date = cur.fetchall()

    i = 1
    while i < rows:
        data = (i, rand_bool[i], rand_name[i], rand_date[i])
        cur.execute(SQL, data)
        i = i + 1
```

Вимоги до пункту №3 деталізованого завдання:

- ілюстрації уведення пошукового запиту та результатів виконання запитів;
- копії SQL-запитів, що ілюструють пошук з зазначеними початковими параметрами

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range
8. SQL query

Select option by number...
8
Choose query:
1.Query1(users + email + folders)
2.Query2(folders + folders_messages + messages)
3.Query3(users + email)
Input number to select...
1
SELECT script where phone_number>100000, email_type with 'B' in it and folder_type with 'C' in it
['user_id', 'username', 'phone_number', 'email_id', 'user_id', 'email_adress', 'email_type', 'folders_id', 'email_id', 'folder_name', 'folder_type']
(10, 'YA', 234567, 7, 10, '830XA', 'JB', 1, 7, 'Y622J', 'CT')
(10, 'YA', 234567, 12, 10, '667VR', 'BT', 6, 12, 'D492J', 'CQ')
Input number range:
200000
Input email type:
T
Input folder type:
Q
SELECT script where phone_number>200000, email_type with 'T' in it and folder_type with 'Q' in it
['user_id', 'username', 'phone_number', 'email_id', 'user_id', 'email_adress', 'email_type', 'folders_id', 'email_id', 'folder_name', 'folder_type']
(10, 'YA', 234567, 12, 10, '667VR', 'BT', 6, 12, 'D492J', 'CQ')

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range
8. SQL query

Select option by number...
8
Choose query:
1.Query1(users + email + folders)
2.Query2(folders + folders_messages + messages)
3.Query3(users + email)
Input number to select...
2
SELECT script where email_id < 10, 2 < messages_id < 9 and messages_name with 'U' in it
['folders_id', 'email_id', 'folder_name', 'folder_type', 'folders_id', 'messages_id', 'messages_id', 'read_or_not', 'message_name', 'message_date']
(1, 7, 'Y622J', 'CT', 1, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))
(1, 7, 'Y622J', 'CT', 1, 3, 3, False, 'YYUS', datetime.date(2019, 9, 22))
(3, 7, 'G788M', 'AU', 3, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))
(7, 9, 'U714Q', 'HU', 7, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))
Input email_id range:
12
Input first value for message_id >'m1':
1
Input second value for message_id <'m2':
4
Input message name:
K
SELECT script where email_id < 12, 1 < messages_id < 6 and messages_name with 'K' in it
['folders_id', 'email_id', 'folder_name', 'folder_type', 'folders_id', 'messages_id', 'messages_id', 'read_or_not', 'message_name', 'message_date']
(1, 7, 'Y622J', 'CT', 1, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))
(3, 7, 'G788M', 'AU', 3, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))
(7, 9, 'U714Q', 'HU', 7, 5, 5, False, 'UKWV', datetime.date(2019, 10, 23))

Process finished with exit code 0
```

```
-------| Menu |-------
1. Read 1 item
2. Read table
3. Create item
4. Update item
5. Delete item
6. Input random data in tables
7. Read from all tables with range
8. SQL query

Select option by number...
8
Choose query:
1.Query1(users + email + folders)
2.Query2(folders + folders_messages + messages)
3.Query3(users + email)
Input number to select...
3
SELECT script where username have 'A' in it, user_id in email >5 and email_adress have 'D' in it
['user_id', 'username', 'phone_number', 'email_id', 'user_id', 'email_adress', 'email_type']
(8, 'FA', 549064, 8, 6, '684ID', 'RJ')
Input username:
B
Input range for users_id:
1
Input email_adress:
DQ
SELECT script where username have 'B' in it, user_id in email >1 and email_adress have 'DQ' in it
['user_id', 'username', 'phone_number', 'email_id', 'user_id', 'email_adress', 'email_type']
(4, 'BB', 659761, 4, 3, '186DQ', 'LR')
```

```python
def query1(conn, num, e_type, f_type):
    cur = conn.cursor()

    cur.execute(f"SELECT * FROM users as us INNER JOIN (SELECT * FROM email) as em ON us.user_id = em.user_id "
                f"INNER JOIN (SELECT * FROM folders) as fl ON em.email_id = fl.email_id "
                f"WHERE us.phone_number>{num} AND em.email_type LIKE '%{e_type}%' AND fl.folder_type LIKE '%{f_type}%' "
                f"ORDER BY us.user_id;")
    result = cur.fetchall()

    keys = [description[0] for description in cur.description]
    print(keys)

    if cur.rowcount == 0:
        print("[No such data]")

    for item in result:
        print(item)

    cur.close()
```

```python
def query2(conn, e_num, m_range1, m_range2, mess_name):
    cur = conn.cursor()

    cur.execute(f"SELECT * FROM folders as fl INNER JOIN (SELECT * FROM folders_messages) as fm "
                f"ON fl.folders_id = fm.folders_id "
                f"INNER JOIN (SELECT * FROM messages) as ms ON ms.messages_id = fm.messages_id "
                f"WHERE fl.email_id < {e_num} AND (fm.messages_id > {m_range1} AND fm.messages_id < {m_range2}) AND "
                f"(ms.message_name LIKE '%{mess_name}%') "
                f"ORDER BY fl.folders_id ASC;")
    result = cur.fetchall()

    keys = [description[0] for description in cur.description]
    print(keys)

    if cur.rowcount == 0:
        print("[No such data]")

    for item in result:
        print(item)

    cur.close()
```

```python
def query3(conn, username, usr_range, e_adress):
    cur = conn.cursor()

    cur.execute(f"SELECT * FROM users as us INNER JOIN (SELECT * FROM email) as em ON us.user_id = em.email_id "
                f"WHERE us.username LIKE '%{username}%' AND em.user_id > {usr_range} AND "
                f"em.email_adress LIKE '%{e_adress}%' "
                f"ORDER BY us.user_id;")
    result = cur.fetchall()

    keys = [description[0] for description in cur.description]
    print(keys)

    if cur.rowcount == 0:
        print("[No such data]")

    for item in result:
        print(item)

    cur.close()
```

Вимоги до пункту №4 деталізованого завдання:

- ілюстрації програмного коду з репозиторію Git.