МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота №2

з дисципліни “Бази даних”

тема “Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL”

Виконав
студент ІІ курсу
групи КП-93
Шевляков Андрій Олексійович

Перевірив
“” “вересня” 2020р.
викладач
Петрашенко Андрій Васильович

Київ 2020

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).

URL репозиторія: https://github.com/OXOTNIKPROGER/DB

1) Валідація даних та обробка виключних ситуацій

   Приклад валідації деяких даних при додаванні нового запису до таблиці Author

```
1)Add
2)Update
3)Delete
4)Generate
5)Find/Filter
6)Get
7)Set links
8)Delete links
Quit-for exit
Input number
1
Input name:
name_
Input date of first publication(yyyy-mm-dd):
1908432-129401-12490
Incorrect date format
Input date of first publication(yyyy-mm-dd):
1000-10-10
Input year of birth:
rui1
Not a number!
Input year of birth:
83192
Input year of death:
12
```

   Приклад помилок які виникають one-to-many при порушенні foreign

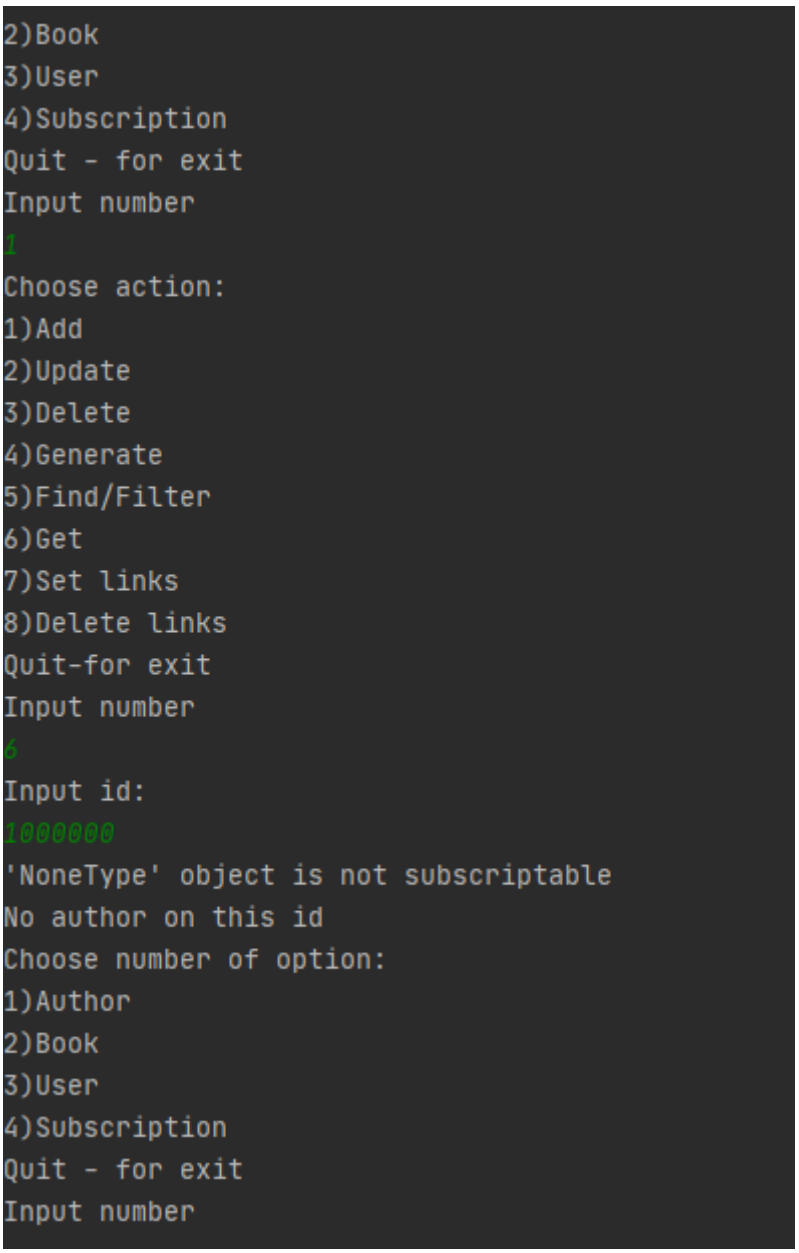
key у таблиці subscription

```
1)Add
2)Update
3)Delete
4)Generate
5)Find/Filter
6)Get
7)Set links
8)Delete links
Quit-for exit
Input number
1
Input number:
1289
Input price:
132
Input expire_date(yyyy-mm-dd):
1000-10-10
Input number of books:
190322
Input user_id:
1000000
No user on this id
```

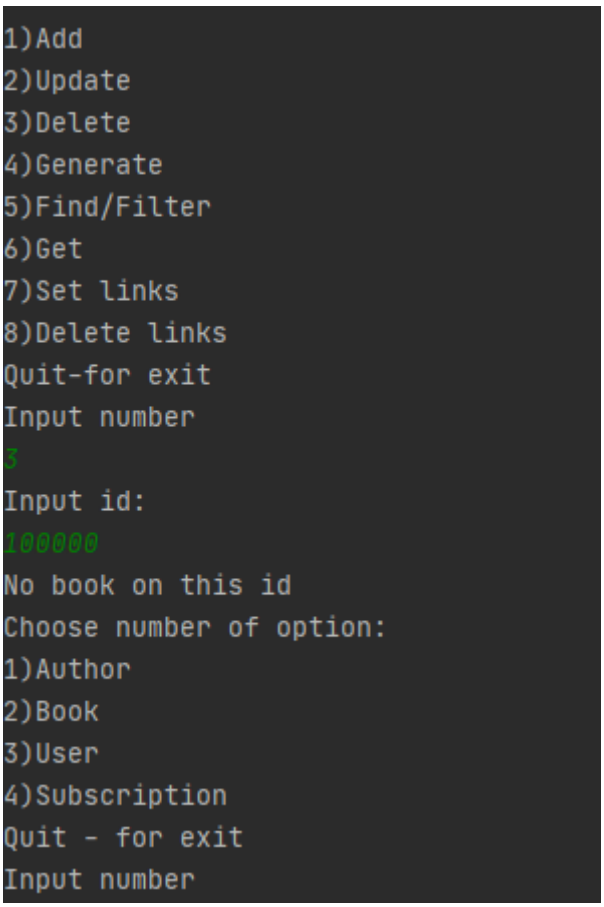
Приклад помилки перехопленою із БД

```
Choose action:
1)Add
2)Update
3)Delete
4)Generate
5)Find/Filter
6)Get
7)Set links
8)Delete links
Quit-for exit
Input number
7
Input main_entity than second
Input id:
1000000
Input id:
1000000
ОШИБКА:  INSERT или UPDATE в таблице "books_authors" нарушает ограничение внешнего ключа "books_id"
DETAIL:  Ключ (book_id)=(1000000) отсутствует в таблице "book".
```

Приклад обробки виключної ситуації при перегляді данних із таблиці Author



Приклад обробки помилок при вилученні даних у таблиці Book



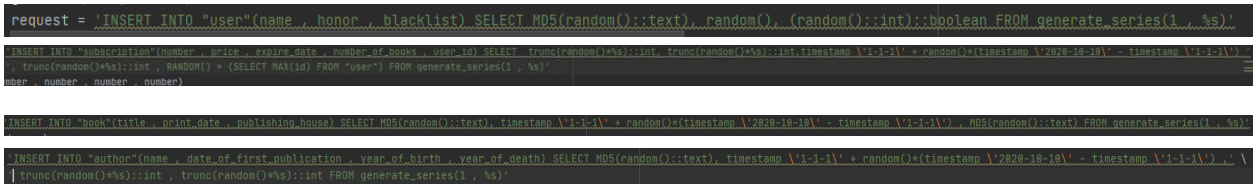
2) Приклад згенерованих даних у таблиці Book та у таблиці Subscription

Результат | План выполнения | Сообщения | Notifications

	id [PK] integer	title text	print_date date	publishing_house text
2079	2081	MJU	2020-06-03	SKSD
2080	2082	XJQ	2009-06-10	UVAS
2081	2083	FUL	2009-10-05	KOJK
2082	2084	FJM	2006-07-11	LBWW
2083	2085	NAT	2018-06-04	EQKS
2084	2086	ING	2013-01-02	HTNP
2085	2087	KML	2019-02-12	RGDK
2086	2088	AKS	2017-06-26	SFQW
2087	2089	FBT	2003-06-17	JGUW
2088	2090	JXC	2004-09-14	VIRB
2089	2091	QIH	2010-08-01	UQSQ
2090	2092	LGQ	2014-01-08	TTYW
2091	2093	IPU	2013-05-01	XUQR
2092	2094	AJM	2010-01-12	PFKV
2093	2095	NOC	2003-05-07	GEAE
2094	2096	FKE	2003-06-28	HMUF
2095	2097	PLW	2006-03-09	FRJT
2096	2098	BFG	2018-08-01	TNNJ
2097	2099	WXS	2012-05-01	YJDL
2098	2100	ITI	2019-07-29	STIN

	id [PK] integer	number integer	price integer	expire_date date	number_of_books integer	user_id integer
773	778	10312	3386	2003-03-29	8889	39646
774	779	1284	372	2009-01-17	10920	12115
775	780	3255	13358	2012-03-09	17079	51956
776	781	8623	18533	2018-01-02	1096	53200
777	782	7250	2580	2009-07-30	10141	5413
778	783	2482	9675	2008-08-23	9699	1290
779	784	14142	1879	2006-11-19	13161	16358
780	785	8762	10091	2019-04-18	189	75242
781	786	4012	17349	2014-01-22	13352	62047
782	787	11885	10299	2020-08-22	14130	92819
783	788	8646	9131	2010-07-17	4949	11356
784	789	10662	17419	2014-10-03	15607	68632
785	790	5101	14670	2018-10-08	18765	88403
786	791	102	16658	2008-03-15	3036	71510
787	792	2892	13466	2003-11-03	823	97680
788	793	6848	11828	2007-07-24	11442	36486
789	794	873	8031	2010-10-06	16894	7905
790	795	15383	12366	2015-03-30	924	68791
791	796	15605	6929	2018-07-16	107	25888
792	797	18481	7589	2004-05-29	8062	32378

Запити, які генерують рандомізовані дані



3)

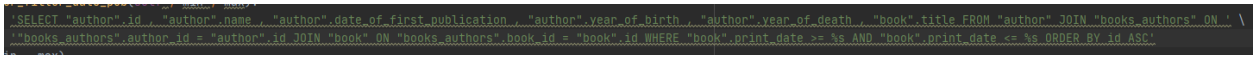
Приклад пошукового запиту у Author

Приклад пошукового запиту у Author

```
Choose option:
2
Input min print_date(yyyy-mm-dd):
1000-10-10
Input max print_date(yyyy-mm-dd):
2000-10-10
///Execution time:
0.004987955093383789
Author id-> 1
Author name-> Jack London
Author date_of_first_publication-> 1895-01-02
Author year_of_birth-> 1876
Author year_of_death-> 1916
Author book-> The Call of the Wild


Author id-> 111
Author name-> YF
Author date_of_first_publication-> 2004-01-06
Author year_of_birth-> 25376
Author year_of_death-> 7339
Author book-> The Call of the Wild


Choose number of option:
```



Приклад пошукового запиту у Book

```
4)Generate
5)Find/Filter
6)Get
7)Set links
8)Delete links
Quit-for exit
Input number
5
///Execution time:
0.02296161651611328
title-> War and peace
user name-> Jack London


title-> The Call of the Wild
user name-> Jack London


title-> OSN
user name-> d32764fac192c4d6a1f0b77d8bff45b6


title-> OKC
user name-> fbbd7afffdcdd810d055528c911b69d2
```

```
= 'SELECT b.title , "user".name FROM "book" b JOIN "books_users" ON b.id = "books_users".book_id JOIN "user" ON "user".id = "books_users".user_id ORDER BY title DESC, name ASC'
```

Приклад пошукового запиту у User

```
Choose option:
1
Input id10
Input id20
///Execution time:
0.019955158233642578
User id-> 20
User name-> 9ae4118ba8bc5762076be493e7586ee9
User honor-> 0.6095542423202609
User blacklist-> False


User id-> 12
User name-> new_na
User honor-> 0.1
User blacklist-> True


User id-> 13
User name-> 0eb8144109f4b5661f407b4fcb07f6b7
User honor-> 0.7291174122091562
User blacklist-> False


User id-> 14
User name-> 3a5126b99e66bd3d79dde00fd2e31552
User honor-> 0.857023074928005
User blacklist-> True
```

```
User id-> 16
User name-> 05bcad233f8a78ec896771f46b28440a
User honor-> 0.3006695366598997
User blacklist-> True


User id-> 18
User name-> a26a322db7853aa74dfa94481cb2c2e5
User honor-> 0.16987905905652312
User blacklist-> True


User id-> 11
User name-> new_name
User honor-> 0.9
User blacklist-> True


User id-> 15
User name-> 802c50c8815faa859433a49e39d33d8f
User honor-> 0.4687090630249884
User blacklist-> False


User id-> 17
User name-> 9ef0e53666da695a1a680149a04cdcc5
User honor-> 0.2723076352100051
User blacklist-> True
```

```
'SELECT * FROM "user" WHERE "user".id >= %s AND "user".id <= %s ORDER BY(SELECT COUNT(*) FROM "subscription" WHERE "subscription".user_id = "user".id)'
```

Приклад пошукового запиту у User

```
2)Find users sorted by amount subscription desc
3)Find users with false blacklist order by subscription(amount)
4)Exit
Choose option:
2
///Execution time:
143.45428085327148
User id-> 11
User name-> new_name
User honor-> 0.9
User blacklist-> True


User id-> 12
```

```
'SELECT * FROM "user" ORDER BY(SELECT COUNT("subscription".id) FROM "subscription" WHERE "subscription".user_id = "user".id), "user".name DESC LIMIT 35'
```

Приклад пошукового запиту у User

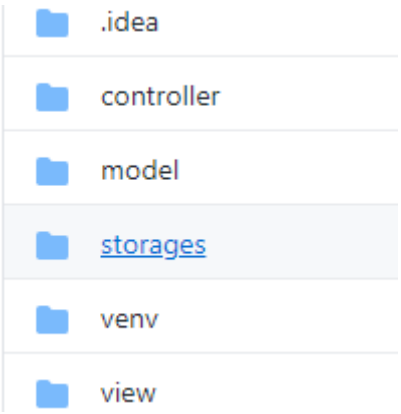
```
User id-> 68898
User name-> f2d62eb5a1798ba32cf58f2c4fd93f2d
User honor-> 0.6820568422974524
User blacklist-> False


User id-> 82453
User name-> 06a1b04f7bdf4fb90b2b54f0aa098805
User honor-> 0.7132282258676312
User blacklist-> False


User id-> 87133
User name-> dbfc3ba8f192ba5fff92a6c947a31d14
```

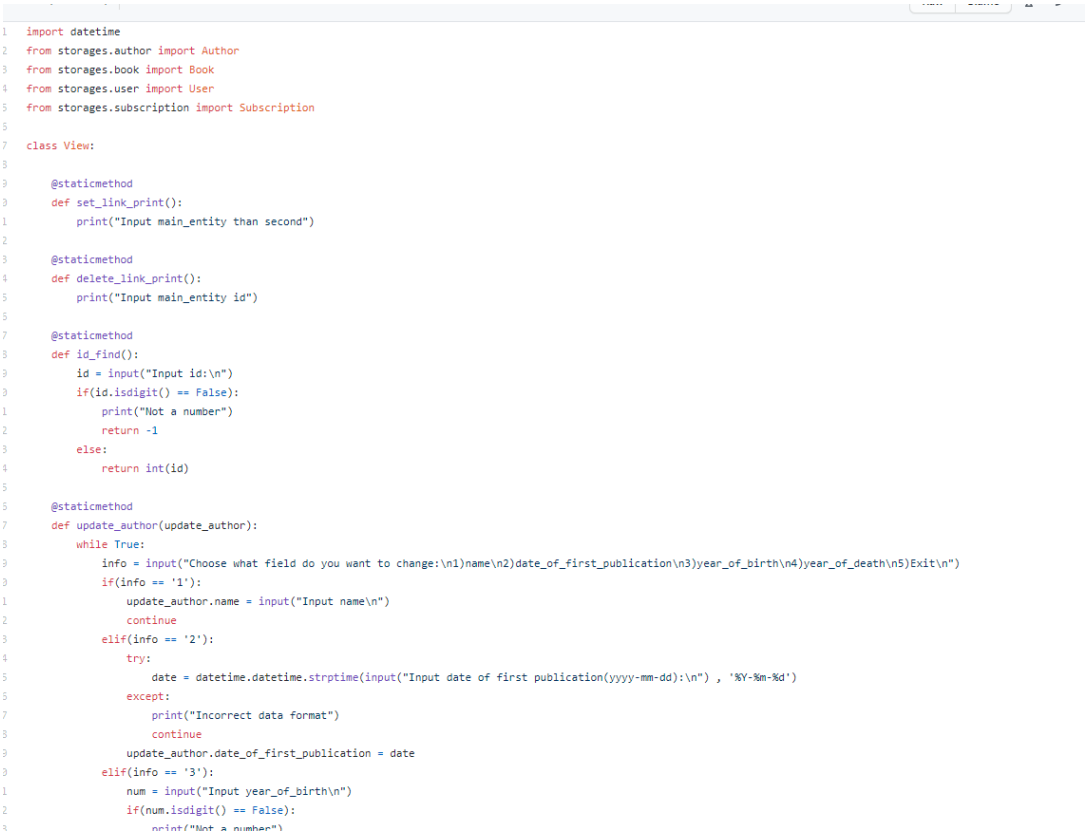
```
'SELECT * FROM "user" WHERE "user".blacklist = false ORDER BY(SELECT COUNT(id) FROM "subscription" WHERE "subscription".user_id = "user".id)'
```

4) Приклад файлової структури коду MVC



Storages містить обьекти таблиці БД

Файл view.py містить статичні методик ласу, які відповідають за обробку запитів клієнта та збір данних



```python
import datetime
from storages.author import Author
from storages.book import Book
from storages.user import User
from storages.subscription import Subscription

class View:

    @staticmethod
    def set_link_print():
        print("Input main_entity than second")

    @staticmethod
    def delete_link_print():
        print("Input main_entity id")

    @staticmethod
    def id_find():
        id = input("Input id:\n")
        if(id.isdigit() == False):
            print("Not a number")
            return -1
        else:
            return int(id)

    @staticmethod
    def update_author(update_author):
        while True:
            info = input("Choose what field do you want to change:\n1)name\n2)date_of_first_publication\n3)year_of_birth\n4)year_of_death\n5)Exit\n")
            if(info == '1'):
                update_author.name = input("Input name\n")
                continue
            elif(info == '2'):
                try:
                    date = datetime.datetime.strptime(input("Input date of first publication(yyyy-mm-dd):\n") , '%Y-%m-%d')
                except:
                    print("Incorrect data format")
                    continue
                update_author.date_of_first_publication = date
            elif(info == '3'):
                num = input("Input year_of_birth\n")
                if(num.isdigit() == False):
                    print("Not a number")
```

Файл контроллер відповідає за виклик статичних функцій у необхідний момент, прийняття рішень на основі запитів клієнта та виклик методів із класів моделі

```
from view.view import View
from model.AuthorModel import AuthorModel
from model.BookModel import BookModel
from model.UserModel import UserModel
from model.SubscriptionModel import SubscriptionModel

in_menu = True
authorModel = AuthorModel('lab' , 'postgres' , 'Scorpions' , 'localhost')
bookModel = BookModel('lab' , 'postgres' , 'Scorpions' , 'localhost')
userModel = UserModel('lab' , 'postgres' , 'Scorpions' , 'localhost')
subscriptionModel = SubscriptionModel('lab' , 'postgres' , 'Scorpions' , 'localhost')

while(in_menu):
    info = View.main_menu()
    if(info == 'error'):
        print("Incorrect input")
        continue
    if(info == '1'):
        while True:
            sub_info = View.sub_menu()
            if(sub_info == -1):
                continue
            else:
                break
        if(sub_info == '1'):
            author = View.add_author()
            authorModel.add_entity(author)
            continue
        elif(sub_info == '2'):
            while True:
                id = View.id_find()
                if(id == -1):
                    continue
                else:
                    break
            if(authorModel.get_entity(id) == None):
                print("No author on this id!")
            else:
                authorModel.update_entity(View.update_author(authorModel.get_entity(id)))
        elif(sub_info == '3'):
            while True:
                id = View.id_find()
```

Файли моделі(один абстрактний та 4 наслідники) містять методи для спілкування із СУБД та створення запитів

```python
from model.DBmodel import DBModel
from storages.author import Author
import psycopg2
import time

class AuthorModel(DBModel):
    def __init__(self , dbname , user , password , host ):
        super(AuthorModel, self).__init__(dbname , user, password , host)
        try:
            self.cursor = self.conn.cursor()
        except (Exception , psycopg2.DatabaseError) as error:
            print(error)

    def __del__(self):
        try:
            self.cursor.close()
            self.conn.close()
        except (Exception , psycopg2.DatabaseError) as error:
            print(error)

    def add_entity(self , new_entity):
       request = 'INSERT INTO author(name , date_of_first_publication , year_of_birth , year_of_death) VALUES (%s, %s, %s, %s)'
       data = (new_entity.name, new_entity.date_of_first_publication, new_entity.year_of_birth, new_entity.year_of_death)
       try:
           self.cursor.execute(request, data)
           self.conn.commit()
       except(Exception,psycopg2.DatabaseError) as error:
           print(error)

    def get_entities(self):
        request = 'SELECT * FROM author'
        authors = list()
        try:
            self.cursor.execute(request)
            records = self.cursor.fetchall()
            if (records != None):
                for record in records:
                    authors.append(Author(record[0] , record[1] , record[2] , record[3] , record[4]))
        except (Exception , psycopg2.DatabaseError) as error:
            print(error)
        finally:
            return authors
```