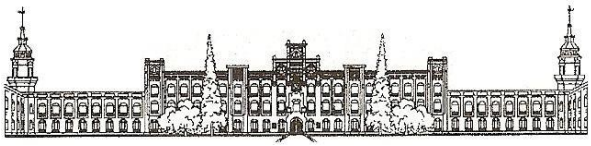


Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут»

Розрахунково-графічна робота.

Створення додатку бази даних, орієнтованого на взаємодію з СУБД PostgreSQL

Виконала студентка групи: КВ-13

ПІБ: Саюн Д.М.

Телеграм: https://t.me/msdsiu

Київ 2023

Метою роботи є здобуття вмінь програмування прикладних додатків баз даних PostgreSQL.
Загальне завдання роботи полягає у наступному:

1. Реалізувати функції перегляду, внесення, редагування та вилучення даних у таблицях бази даних, створених у лабораторній роботі №1, засобами консольного інтерфейсу.
1. Передбачити автоматичне пакетне генерування «рандомізованих» даних у базі.
2. 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.
3. Програмний код виконати згідно шаблону MVC (модель-подання-контролер)

https://github.com/SaiunD/rgr-publishing

Структура бази даних

Програма створювалась для онлайн-платформи зберігання та пошуку наукових публікацій. Розглянемо її більш детально:

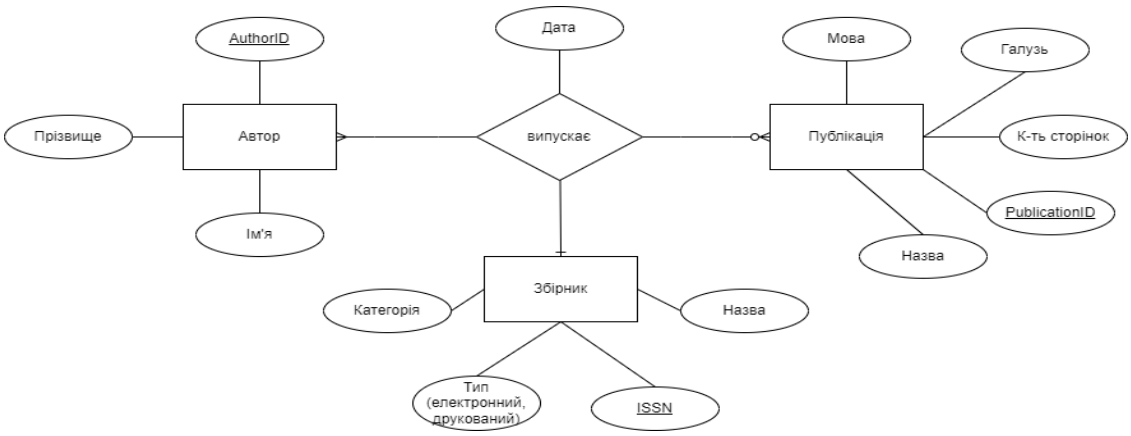


Рис.1 ER-діаграма за нотацією «Crow’s foot»

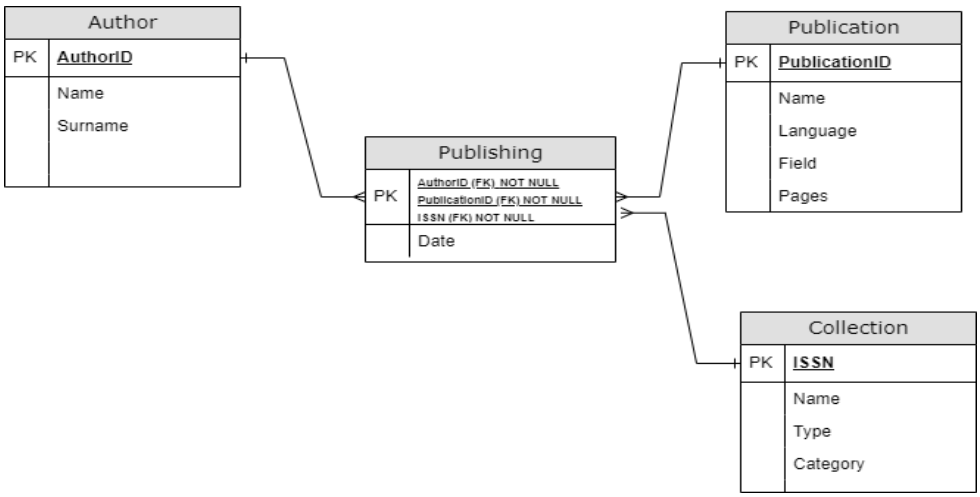


Рис.2 Схема бази даних

Модель має такі сутності: *Автор, Публікація, Збірник.*

1) *Автор* має атрибути:
   - AuthorID (ідентифікатор автора)
   - Ім’я
   - Прізвище

   Це основні дані, які потрібні від автора для публікацій.
2) *Публікація* має атрибути:
   - PublicationID (ідентифікатор публікації)
   - Назва
   - Галузь
   - Мова
   - Кількість сторінок
3) *Збірник* потрібний для випуску публікації і містить атрибути:
   - ISSN (ідентифікатор збірника)
   - Назва
   - Тип (у якому форматі був виданий)
   - Категорія (міжнародний або всеукраїнський)

Ці три сутності реалізовані у тернарний зв’язок багато до багатьох (N:M). Один автор може написати багато публікацій виданих у збірники, а одну публікацію можуть написати багато авторів в один збірник.

Програма

```
Welcome! Which table you want to work with?
1. Author
2. Publication
3. Collection
4. Publishing
5. Queries
6. Quit
Enter your choice: 4
```

Рис.3 Початкове меню

Під час запуску відображається початкове меню в якому пропонується обрати таблицю для подальшої роботи (1-4) або запити (5). Обравши таблицю, відображаються наступне меню:

```
Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice:
```

Рис.4 Меню для таблиць

Для таблиць доступні додавання (1), перегляд (2), оновлення (3), вилучення (4) та генерування даних (5). Опція виходу (6).

Додамо автора та публікацію:

```
Welcome! Which table you want to work with?
1. Author
2. Publication
3. Collection
4. Publishing
5. Queries
6. Quit
Enter your choice: 1

Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 1
Enter AuthorID: AU1
Enter author's Name: Jane
Enter author's Surname: Simpson
Added successfully!

Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 6

Welcome! Which table you want to work with?
1. Author
2. Publication
3. Collection
4. Publishing
5. Queries
6. Quit
Enter your choice: 2
Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 1
Enter PublicationID: Pub1
Enter publication's name: Agroeconomics
Enter language: english
Enter field: economics
Enter pages: 45
Added successfully!

Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 6
```

Рис.5 Додавання автора та публікації

Тепер спробуємо додати в таблицю Publishing видання щойно створеним автором AU1 публікації Pub1 в недоданий збірни 43300:

```
Welcome! Which table you want to work with?
1. Author
2. Publication
3. Collection
4. Publishing
5. Queries
6. Quit
Enter your choice: 4

Menu:
1. Add Task
2. View Tasks
4. Delete Task
5. Generate data
6. Back
Enter your choice: 1
Enter AuthorID: AU1
Enter PublicationID: Pub1
Enter ISSN: 43300
Enter Date: 2008-07-02
ПОМИЛКА:  insert або update в таблиці "Publishing" порушує обмеження зовнішнього ключа "Publishing_ISSN_fkey"
DETAIL:  Ключ (ISSN)=(43300) не присутній в таблиці "Collection".
```

Рис.6 Результат вставки в дочірню таблицю даних яких не існує в батьківських

Перейдемо до генерації даних:

```
Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 5
Enter number of data to generate: 3
Generated successfully!

Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 2
Collections:
+----------+--------+------------+----------+
|   ISSN   |  Name  |    Type    | Category |
+----------+--------+------------+----------+
| 36697458 | CjdNsc |   e-book   |    B     |
| 63167020 | CrsDen | paper book |    B     |
| 85414701 | YtyPno |   e-book   |    C     |
+----------+--------+------------+----------+
```

Рис.7 Генерація 3 випадкових збірників

Додамо запис в Publishing:

```
Menu:
1. Add Task
2. View Tasks
4. Delete Task
5. Generate data
6. Back
Enter your choice: 1
Enter AuthorID: AU1
Enter PublicationID: Pub1
Enter ISSN: 85414701
Enter Date: 2006-08-09
Added successfully!

Menu:
1. Add Task
2. View Tasks
4. Delete Task
5. Generate data
6. Back
Enter your choice: 2
Publishings:
+----------+---------------+----------+------------+
| AuthorID | PublicationID |   ISSN   |    Date    |
+----------+---------------+----------+------------+
|   AU1    |     Pub1      | 85414701 | 2006-08-09 |
+----------+---------------+----------+------------+
```

Рис.8 Додавання даних в Publishing

Тепер перевіримо програму на вилучення з батьківської таблиці даних:

```
Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 4
Enter AuthorID: AU1
ПОМИЛКА:  update або delete в таблиці "Author" порушує обмеження зовнішнього ключа "Publishing_AuthorID_fkey" таблиці "Publishing"
DETAIL:  На ключ (AuthorID)=(AU1) все ще є посилання в таблиці "Publishing".
```

Рис.9 Результат видалення з батьківської таблиці, даних які є в дочірній

Програма видає помилку та не видаляє дані, поки дані не будуть вилучені з дочірньої таблиці.

Згенеруємо для таблиці Author:

```
Menu:
1. Add Task
2. View Tasks
3. Update Task
4. Delete Task
5. Generate data
6. Back
Enter your choice: 5
Enter number of data to generate: 100000
Generated successfully!
```

Рис.10 Генерація 100 000 даних у таблицю Author

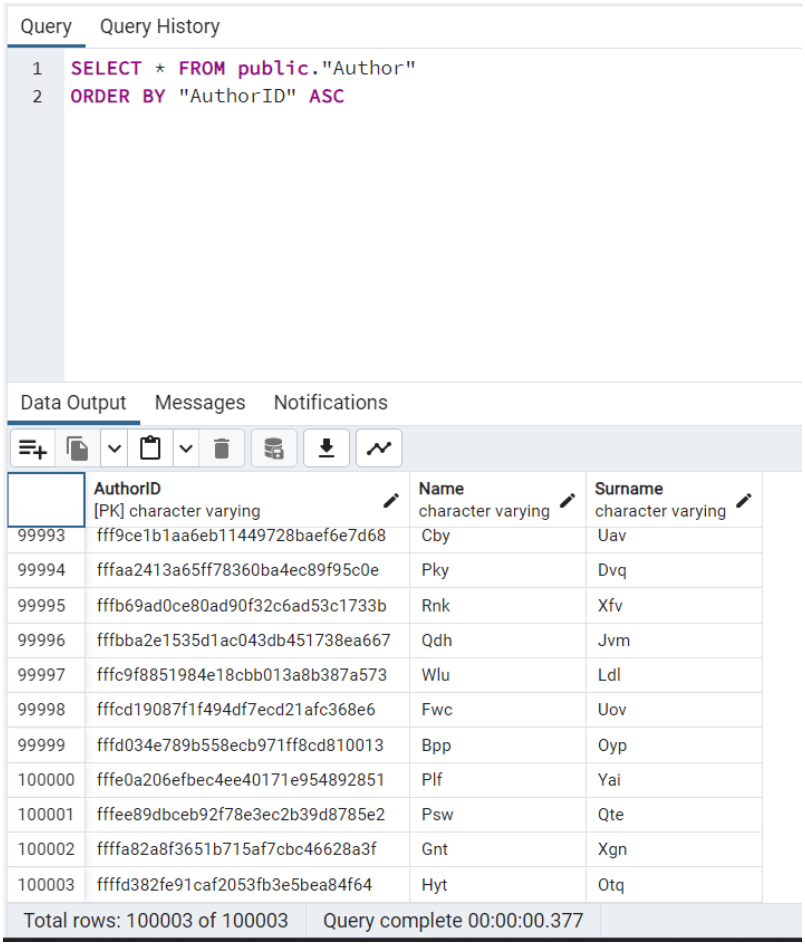
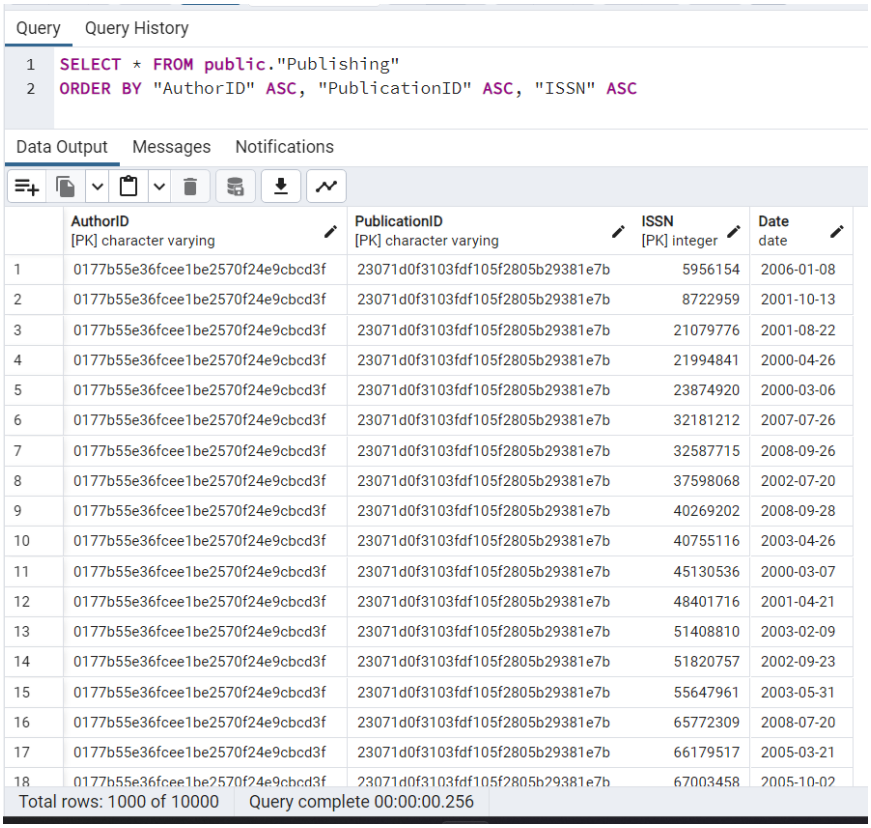


Рис.11 Результат генерації 100 000 даних (3 даних були додані вручну)

Згенеруємо дані і для таблиці Publishing:

```
Menu:
1. Add Task
2. View Tasks
4. Delete Task
5. Generate data
6. Back
Enter your choice: 5
Enter number of data to generate: 10000
Generated successfully!
```

Рис.12 Генерація 10 000 даних для Publishing



Query Query History

```
1  SELECT * FROM public."Publishing"
2  ORDER BY "AuthorID" ASC, "PublicationID" ASC, "ISSN" ASC
```

Data Output Messages Notifications

	AuthorID [PK] character varying	PublicationID [PK] character varying	ISSN [PK] integer	Date date
1	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	5956154	2006-01-08
2	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	8722959	2001-10-13
3	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	21079776	2001-08-22
4	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	21994841	2000-04-26
5	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	23874920	2000-03-06
6	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	32181212	2007-07-26
7	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	32587715	2008-09-26
8	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	37598068	2002-07-20
9	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	40269202	2008-09-28
10	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	40755116	2003-04-26
11	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	45130536	2000-03-07
12	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	48401716	2001-04-21
13	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	51408810	2003-02-09
14	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	51820757	2002-09-23
15	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	55647961	2003-05-31
16	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	65772309	2008-07-20
17	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	66179517	2005-03-21
18	0177b55e36fcee1be2570f24e9cbcd3f	23071d0f3103fdf105f2805b29381e7b	67003458	2005-10-02

Total rows: 1000 of 10000 Query complete 00:00:00.256

Рис.13 Результат генерації Publishing

SQL запити, які використовувались для генерації:

Для Author:

```
INSERT INTO "Author"
SELECT DISTINCT
md5(random()::text),
chr(trunc(65+random()*25)::int) || chr(trunc(97+random()*26)::int) ||
chr(trunc(97+random()*26)::int),
chr(trunc(65+random()*25)::int) || chr(trunc(97+random()*26)::int) ||
chr(trunc(97+random()*26)::int)
FROM generate_series(1, num)
```

Для Publication:

```
INSERT INTO "Publication"
SELECT DISTINCT
md5(random()::text),
chr(trunc(65+random()*25)::int) || chr(trunc(97+random()*26)::int) ||
chr(trunc(97+random()*26)::int) || chr(trunc(65+random()*25)::int) ||
chr(trunc(97+random()*26)::int) || chr(trunc(97+random()*26)::int),
('{"ukrainian", "english", "spanish", "croatian", "portuguese",
"french"}'::text[])[floor(random()*6)+1],
('{"medicine", "biology", "engineering", "economics", "physics", "chemistry", "history",
"philosophy"}'::text[])[floor(random()*8+1)],
floor(random() * 100)
FROM generate_series(1, num)
```

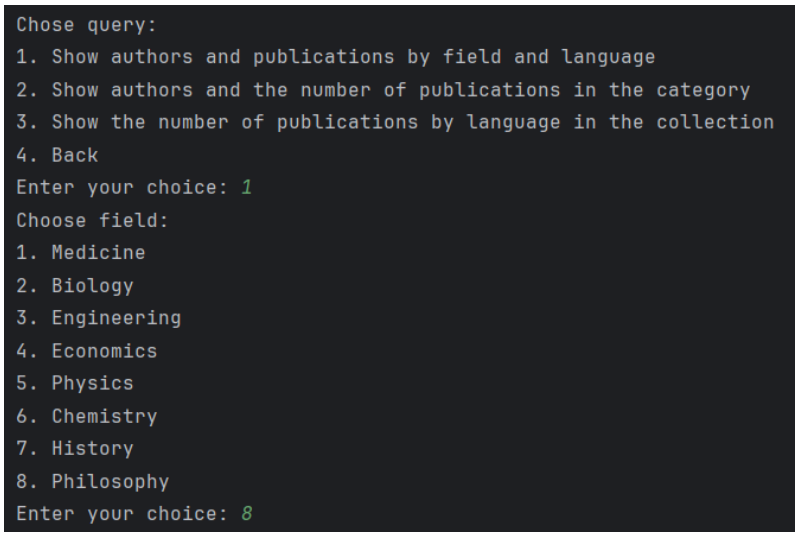
Для Collection:

```
INSERT INTO "Collection" SELECT pr."ISSN", pr."Name", pr."Type", pr."Category"
FROM "Collection"
RIGHT JOIN (SELECT DISTINCT floor(random() * 100000000) + 1,
chr(trunc(65+random()*25)::int) || chr(trunc(97+random()*26)::int) ||
chr(trunc(97+random()*26)::int) || chr(trunc(65+random()*25)::int) ||
chr(trunc(97+random()*26)::int) || chr(trunc(97+random()*26)::int),
('{"e-book", "paper book"}'::text[])[floor(random()*2)+1],
('{"A", "B", "C"}'::text[])[floor(random()*3+1)]
FROM generate_series(1, num)) pr ON "Collection"."ISSN" = sb."ISSN" and
"Collection"."Name"=sb."Name" and "Collection"."Type" = sb."Type" and
"Collection"."Category"=sb."Category"
WHERE "Collection"."ISSN" is null and "Collection"."Name" is null and
"Collection"."Type" is null and "Collection"."Category" is null
```

Для Publishing:

```
INSERT INTO "Publishing"
SELECT pr."AuthorID", pr."PublicationID", pr."ISSN", pr."Date"
FROM "Publishing"
RIGHT JOIN (SELECT DISTINCT t1."AuthorID", t2."PublicationID", t3."ISSN",
'2000-01-01'::date + trunc(random() * 366 * 10)::int as "Date"
FROM (SELECT "AuthorID", row_number() OVER (ORDER BY random()) as rn
FROM "Author" order by random() LIMIT num3) t1,
(SELECT "PublicationID", row_number() OVER (ORDER BY random()) as rn FROM
"Publication" order by random() LIMIT num3) t2,
(SELECT "ISSN", row_number() OVER (ORDER BY random()) as rn FROM
"Collection" order by random() LIMIT num3) t3 LIMIT num) pr
ON "Publishing"."AuthorID" = pr."AuthorID" AND "Publishing"."PublicationID" =
pr."PublicationID" AND "Publishing"."ISSN" = pr."ISSN"
WHERE "Publishing"."AuthorID" IS NULL and "Publishing"."PublicationID" is null
and "Publishing"."ISSN" is null
```

Перейдемо до SQL запитів:



```
Enter your choice: 8
Choose language:
1. Ukrainian
2. English
3. Spanish
4. Croatian
5. Portuguese
6. French
Enter your choice: 1
```

Author's Name	Author's Surname	Publication	Date
Nju	Ylx	AzeHpt	2003-02-17
Jom	Uyd	AzeHpt	2004-12-24
Qhw	Ilk	AzeHpt	2007-03-09
Xni	Xit	AzeHpt	2009-12-03
Fzy	Fpo	AzeHpt	2005-09-22
Kvz	Sxu	AzeHpt	2006-08-20
Vth	Hxh	AzeHpt	2005-10-22
Yhw	Vlb	AzeHpt	2008-08-26
Kvz	Sxu	AzeHpt	2000-06-08
Vth	Hxh	AzeHpt	2000-02-29
Eac	Tdg	AzeHpt	2004-08-11
Nju	Ylx	AzeHpt	2009-03-14
Eac	Tdg	AzeHpt	2002-02-15
Esx	Avp	AzeHpt	2007-02-05
Tkm	Rrg	AzeHpt	2001-06-24

Рис.14 Результат запиту 1

Перший запит показує авторів та публікації, які належать заданій галузі та мові

```
Chose query:
1. Show authors and publications by field and language
2. Show authors and the number of publications in the category
3. Show the number of publications by language in the collection
4. Back
Enter your choice: 2
Choose category:
1. A
2. B
3. C
Enter your choice: 1
Result:
```

Author's Name	Author's Surname	PublicationCount
Rvp	Kio	185
Vql	Jqt	180
Jom	Uyd	180
Njq	Wbq	186
Bri	Omj	188
Xhl	Bti	183
Qvx	Btw	188

Рис.15 Результат запиту 2

Другий запит відображає скільки публікацій в певній категорії всіх наявних авторів

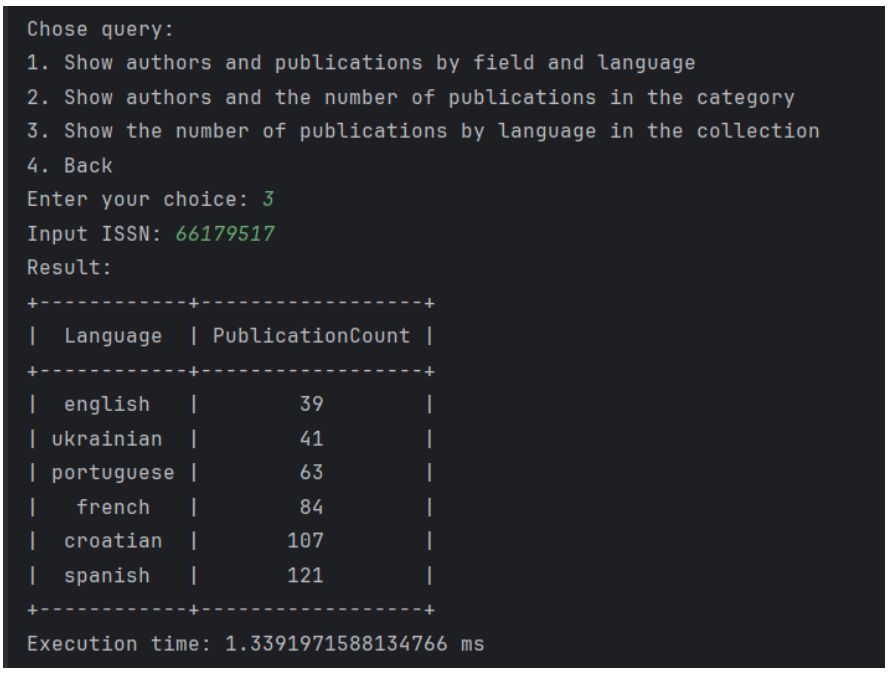


Рис.16 Результат запиту 3

Третій запит показує скільки публікацій та якою мовою було видано в певному збірнику

Перший запит:

```
SELECT *
FROM "Author"
JOIN "Publishing" ON "Author"."AuthorID" = "Publishing"."AuthorID"
JOIN "Publication" ON "Publishing"."PublicationID" = "Publication"."PublicationID"
WHERE "Publication"."Field" = 'philosophy' AND "Publication"."Language" = 'english'
```

Другий запит:

```
SELECT a."Name", a."Surname", COUNT(p."PublicationID") AS PublicationCount
FROM "Author" a
INNER JOIN "Publishing" pb ON a."AuthorID" = pb."AuthorID"
INNER JOIN "Publication" p ON pb."PublicationID" = p."PublicationID"
INNER JOIN "Collection" c ON pb."ISSN" = c."ISSN"
WHERE c."Category" = 'A'
GROUP BY a."Name", a."Surname", c."Category";
```

Третій запит:

```
SELECT P."Language", COUNT(*) AS PublicationCount
FROM "Publication" P
INNER JOIN "Publishing" PA ON PA."PublicationID" = P."PublicationID"
INNER JOIN "Author" A ON A."AuthorID" = PA."AuthorID"
INNER JOIN "Collection" C ON PA."ISSN" = C."ISSN"
WHERE C."ISSN" = 66179517
GROUP BY P."Language"
ORDER BY PublicationCount;
```

Модуль Model та короткий опис

```
import psycopg2                 #імпорт бібліотеки для роботи з базами даних
from view import View           #імпорт модуля View


class Model:
    def __init__(self):        #підключення до postgreSQL
        self.conn = psycopg2.connect(
            dbname='postgres',
            user='postgres',
            password='230520',
            host='localhost',
            port=5432
        )
        self.view = View()
        self.create_table_author()
        self.create_table_publication()
        self.create_table_collection()
        self.create_table_publishing()

    def create_table_author(self):    #створення таблиці Автор, або отримання
                                                                    її з серверу
        c = self.conn.cursor()

        c.execute("SELECT EXISTS (SELECT 1 FROM information_schema.tables
WHERE table_name = 'Author')")
        table_exists = c.fetchone()[0]

        if not table_exists:
            c.execute('''
                CREATE TABLE "Author" (
                    "AuthorID" character varying NOT NULL,
                    "Name" character varying NOT NULL,
                    "Surname" character varying NOT NULL,
                    CONSTRAINT "Author_pkey" PRIMARY KEY ("AuthorID")
                )
            ''')

        self.conn.commit()

    def create_table_publication(self):       #створення таблиці Публікація,
                                              або отримання її з серверу
        c = self.conn.cursor()

        c.execute("SELECT EXISTS (SELECT 1 FROM information_schema.tables
WHERE table_name = 'Publicaion')")
        table_exists = c.fetchone()[0]

        if not table_exists:
            c.execute('''
                CREATE TABLE IF NOT EXISTS "Publication" (
                    "PublicationID" character varying NOT NULL,
                    "Name" character varying NOT NULL,
                    "Language" character varying NOT NULL,
                    "Field" character varying NOT NULL,
                    "Pages" integer,
                    CONSTRAINT "Publication_pkey" PRIMARY KEY
("PublicationID")
                )
            ''')

        self.conn.commit()
```

```
    def create_table_collection(self):        #створення таблиці Збірник,
                                              або отримання її з серверу
        c = self.conn.cursor()

        c.execute("SELECT EXISTS (SELECT 1 FROM information_schema.tables
WHERE table_name = 'Collection')")
        table_exists = c.fetchone()[0]

        if not table_exists:
            c.execute('''
                CREATE TABLE "Collection" (
                    "ISSN" integer NOT NULL,
                    "Name" character varying NOT NULL,
                    "Type" character varying NOT NULL,
                    "Category" character varying NOT NULL,
                    CONSTRAINT "Collection_pkey" PRIMARY KEY ("ISSN")
                )
            ''')

        self.conn.commit()

    def create_table_publishing(self):        #створення таблиці Випуск,
                                              або отримання її з серверу
        c = self.conn.cursor()

        c.execute("SELECT EXISTS (SELECT 1 FROM information_schema.tables
WHERE table_name = 'Publishing')")
        table_exists = c.fetchone()[0]

        if not table_exists:
            c.execute('''
                CREATE TABLE "Publishing" (
                    "AuthorID" character varying COLLATE pg_catalog."default"
NOT NULL,
                    "PublicationID" character varying COLLATE
pg_catalog."default" NOT NULL,
                    "ISSN" integer NOT NULL,
                    "Date" date NOT NULL,
                    CONSTRAINT "Publishing_pkey" PRIMARY KEY ("AuthorID",
"PublicationID", "ISSN")
                )
            ''')

        self.conn.commit()

    def get_all(self, table_name):            #отримати всі дані з таблиці
        c = self.conn.cursor()
        c.execute(f'SELECT * FROM "{table_name}"')
        return c.fetchall()

    def add_data(self, table_name, data): #додати дані до таблиці
        try:
            c = self.conn.cursor()
            placeholders = ", ".join(["%s"] * len(data))
            sql = f"INSERT INTO \"{table_name}\" VALUES ({placeholders});"
            c.execute(sql, list(data.values()))
            self.conn.commit()
            self.view.show_message("Added successfully!")
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)
```

```
def update_data(self, table_name, data, condition_column, condition_value):
                                                #оновити дані в таблиці
        try:
            if not table_name or not data or not condition_column or
condition_value is None:
                print("Insufficient information to update data.")
                return

            c = self.conn.cursor()
            set_clause = ", ".join([f"{key} = %s" for key in data.keys()])
            sql = f"UPDATE \"{table_name}\" SET {set_clause} WHERE
\"{condition_column}\" = %s;"

            values = list(data.values())
            values.append(condition_value)
            c.execute(sql, values)
            self.conn.commit()
            self.view.show_message("Updated successfully!")
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)

    def delete_data(self, table_name, cond, val):     #вилучити дані з
                                            таблиці(для батьківських таблиць)
        try:
            if not table_name or cond is None:
                print("Insufficient information to update data.")
                return

            c = self.conn.cursor()
            sql = f"DELETE FROM \"{table_name}\" WHERE \"{cond}\" = '{val}';"
            c.execute(sql)
            self.conn.commit()
            self.view.show_message("Deleted successfully!")
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)

    def delete_data_publishing(self, table_name, cond1, val1, cond2, val2,
cond3, val3):
                                                #вилучити дані з таблиці Випуск
        try:
            if not table_name or cond1 or cond2 or cond3 is None:
                print("Insufficient information to update data.")
                return

            c = self.conn.cursor()
            sql = f"DELETE FROM \"{table_name}\" WHERE \"{cond1}\" = '{val1}'
AND \"{cond2}\" = '{val2}'AND \"{cond3}\" = {val3}"
            c.execute(sql)
            self.conn.commit()
            self.view.show_message("Deleted successfully!")
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)
```

```
def generate_data(self, table_name, num):     #генерація даних для таблиці
        try:
            c = self.conn.cursor()
            if table_name == "Author":
                sql = (f"insert into \"Author\" select distinct
md5(random()::text), chr(trunc(65+random()*25)::int) ||
chr(trunc(97+random()*26)::int) || chr(trunc(97+random()*26)::int),
chr(trunc(65+random()*25)::int) || chr(trunc(97+random()*26)::int) ||
chr(trunc(97+random()*26)::int)"
                       f"from generate_series(1,{num})")
            elif table_name == "Publication":
                sql = (f"INSERT INTO \"Publication\""
                       f"SELECT DISTINCT "
                       f"md5(random()::text), "
                       f"chr(trunc(65 + random() * 25)::int) || chr(trunc(97
+ random() * 26)::int) || chr(trunc(97 + random() * 26)::int) || chr(trunc(65
+ random() * 25)::int) || chr(trunc(97 + random() * 26)::int) || chr(trunc(97
+ random() * 26)::int), "
                       f"('{{\"ukrainian\", \"english\", \"spanish\",
\"croatian\", \"portuguese\", \"french\"}}'::text[])[floor(random()*6)+1], "
                       f"('{{\"medicine\", \"biology\", \"engineering\",
\"economics\", \"physics\", \"chemistry\", \"history\",
\"philosophy\"}}'::text[])[floor(random()*8+1)], floor(random() * 100) "
                       f"FROM generate_series(1, {num})"
                       )
            elif table_name == "Collection":
                sql = (f"INSERT INTO \"Collection\""
                       f"SELECT pr.\"ISSN\", pr.\"Name\", pr.\"Type\",
pr.\"Category\""
                       f"FROM \"Collection\""
                       f"RIGHT JOIN"
                       f"(SELECT DISTINCT "
                       f"floor(random() * 100000000) + 1, "
                       f"chr(trunc(65 + random() * 25)::int) || chr(trunc(97
+ random() * 26)::int) || chr(trunc(97 + random() * 26)::int) || chr(trunc(65
+ random() * 25)::int) || chr(trunc(97 + random() * 26)::int) || chr(trunc(97
+ random() * 26)::int), "
                       f"('{{\"e-book\", \"paper
book\"}}'::text[])[floor(random()*2)+1], "
                       f"('{{\"A\", \"B\",
\"C\"}}'::text[])[floor(random()*3+1)] "
                       f"FROM generate_series(1, {num})) pr"
                       f"ON \"Collection\".\"ISSN\" = sb.\"ISSN\" and
\"Collection\".\"Name\"=sb.\"Name\" and \"Collection\".\"Type\" = sb.\"Type\"
and \"Collection\".\"Category\"=sb.\"Category\""
                       f"WHERE \"Collection\".\"ISSN\" is null and
\"Collection\".\"Name\" is null and \"Collection\".\"Type\" is null and
\"Collection\".\"Category\" is null"
                       )
            elif table_name == "Publishing":
                num3 = round(num**(1/3), 0)
                sql = (f"INSERT INTO \"Publishing\" "
                       f"SELECT pr.\"AuthorID\", pr.\"PublicationID\",
pr.\"ISSN\", pr.\"Date\" "
                       f"FROM \"Publishing\" RIGHT JOIN"
                       f"(SELECT DISTINCT "
                       f"t1.\"AuthorID\", "
                       f"t2.\"PublicationID\", "
                       f"t3.\"ISSN\", "
                       f"'2000-01-01'::date + trunc(random() * 366 * 10)::int
as \"Date\" "
                       f"FROM "
                       f"(SELECT \"AuthorID\", row_number() OVER (ORDER BY
random()) as rn FROM \"Author\" order by random() LIMIT {num3}) t1, "
```

```
                        f"(SELECT \"PublicationID\", row_number() OVER (ORDER
BY random()) as rn FROM \"Publication\" order by random() LIMIT {num3}) t2, "


                        f"(SELECT \"ISSN\", row_number() OVER (ORDER BY
random()) as rn FROM \"Collection\" order by random() LIMIT {num3}) t3 "
                        f"LIMIT {num}) pr "
                        f"ON \"Publishing\".\"AuthorID\" = pr.\"AuthorID\" "
                        f"AND \"Publishing\".\"PublicationID\" =
pr.\"PublicationID\""
                        f"AND \"Publishing\".\"ISSN\" = pr.\"ISSN\""
                        f"WHERE \"Publishing\".\"AuthorID\" IS NULL and
\"Publishing\".\"PublicationID\" is null and \"Publishing\".\"ISSN\" is null"
                        )
            c.execute(sql, num)
            self.conn.commit()
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)

    def show_by_field_language(self, field, language):       #запит 1.
        показує авторів та публікації, які належать заданій галузі та мові
        try:
            c = self.conn.cursor()
            sql = (f"SELECT \"Author\".\"Name\" AS author_name,
\"Author\".\"Surname\" AS author_surname, \"Publication\".\"Name\" AS
pub_name, \"Publishing\".\"Date\" "
                   f"FROM \"Author\" "
                   f"JOIN \"Publishing\" ON \"Author\".\"AuthorID\" =
\"Publishing\".\"AuthorID\""
                   f"JOIN \"Publication\" ON \"Publishing\".\"PublicationID\"
= \"Publication\".\"PublicationID\" "
                   f"WHERE \"Publication\".\"Field\" = '{field}' AND
\"Publication\".\"Language\" = '{language}'"
                   )
            c.execute(sql)
            self.conn.commit()
            return c.fetchall()
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)

    def show_by_category(self, category):                    #запит 2.
        відображає скільки публікацій в певній категорії всіх наявних авторів
        try:
            c = self.conn.cursor()
            sql = (f"SELECT a.\"Name\", a.\"Surname\",
COUNT(p.\"PublicationID\") AS PublicationCount "
                   f"FROM \"Author\" a "
                   f"INNER JOIN \"Publishing\" pb ON a.\"AuthorID\" =
pb.\"AuthorID\" "
                   f"INNER JOIN \"Publication\" p ON pb.\"PublicationID\" =
p.\"PublicationID\" "
                   f"INNER JOIN \"Collection\" c ON pb.\"ISSN\" = c.\"ISSN\"
"
                   f"WHERE c.\"Category\" = '{category}' "
                   f"GROUP BY a.\"Name\", a.\"Surname\", c.\"Category\""
                   )
            c.execute(sql)
            self.conn.commit()
            return c.fetchall()
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)
```

```
    def show_collection(self, issn):                            #запит 3.
        показує скільки публікацій та якою мовою було видано в певному
                                                                        збірнику
        try:
            c = self.conn.cursor()
            sql = (f"SELECT P.\"Language\", COUNT(*) AS PublicationCount "
                   f"FROM \"Publication\" P "
                   f"INNER JOIN \"Publishing\" PA ON PA.\"PublicationID\" =
P.\"PublicationID\" "
                   f"INNER JOIN \"Author\" as A ON A.\"AuthorID\" =
PA.\"AuthorID\" "
                   f"INNER JOIN \"Collection\" C ON PA.\"ISSN\" = C.\"ISSN\"
"
                   f"WHERE c.\"ISSN\" = {issn} "
                   f"GROUP BY P.\"Language\" "
                   f"ORDER BY PublicationCount"
                   )
            c.execute(sql)
            self.conn.commit()
            return c.fetchall()
        except psycopg2.Error as e:
            self.conn.rollback()
            print(e)
```