



НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря
Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

**Кафедра системного програмування та спеціалізованих
комп'ютерних систем**

Лабораторна робота №3

з дисципліни
«Бази даних і засоби управління»

Тема: «Засоби оптимізації роботи СУБД PostgreSQL»

Виконав: студент III курсу

ФПМ групи КВ-84

Голуб Володимир Володимирович

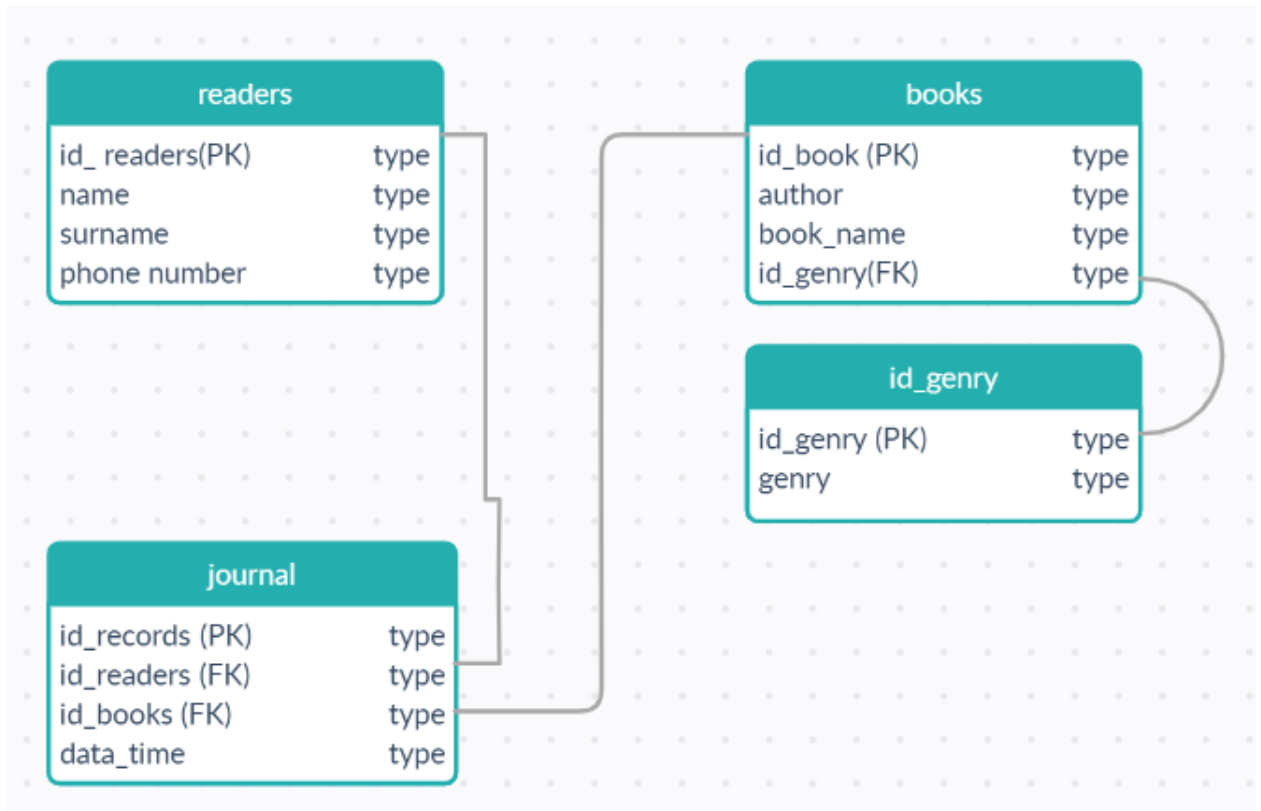
Перевірів:

Київ – 2020

Завдання роботи полягає у наступному:

1. Перетворити модуль “Модель” з шаблону MVC лабораторної роботи №2 у вигляд об’єктно-реляційної проекції (ORM).
2. Створити та проаналізувати різні типи індексів у PostgreSQL.
3. Розробити тригер бази даних PostgreSQL.

Структура БД



Завдання 1

Було змінено з модуля model лабораторної роботи №2, функції delete, insert, update, та записані в файл під назвою MODEL.py інші функції такі як запити та генерація даних не були змінені, та використовуються з минулого модуля.

За посилання на Github:

https://github.com/holubvova/DATA_BASE/blob/master/3lab_BD/MODEL.py

Можна побачити саму реалізацію класів, та відношень між ними у вигляді ORM.

Тут буде показано тестування вище вказаних функцій які змінено, спочатку показно що при введенні хибних даних ми отримаємо повідомлення типу "error", "Error" "input correct number or 4- exit". Вхідні та вихідні дані зміненого модуля лишилися без змін.

Реалізація класів

```
class genry(Base):
    __tablename__ = "id_genry"

    id_genry = Column(Integer, primary_key=True)
    genry = Column(String)
    books = relationship("book")

    def __init__(self, genry):

        self.genry = [genry]
```

```
class book(Base):
    __tablename__ = 'books'

    id_boooks = Column(Integer, primary_key=True)
    author = Column(String)
    name_book = Column(String)
    genry = Column(Integer, ForeignKey('id_genry.id_genry'))
    journal = relationship("journal")

    def __init__(self, author, name_book, genry):
        self.author = author
        self.name_book = name_book
        self.genry = genry
```

```
class readers(Base):  
    __tablename__ = "readers"  
  
    id_readers = Column(Integer, primary_key=True)  
    name = Column(String)  
    surname = Column(String)  
    phone_number = Column(Integer)  
    journal = relationship("journal")  
  
    def __init__(self, name, surname, phone_number):  
        self.name = name  
        self.surname = surname  
        self.phone_number = phone_number
```

```
class journal(Base):  
    __tablename__ = "journal"  
  
    id_records = Column(Integer, primary_key=True)  
  
    id_readers = Column(Integer, ForeignKey('readers.id_readers'))  
  
    id_books = Column(Integer, ForeignKey('books.id_books'))  
  
    data_time = Column(DateTime)  
  
    def __init__(self, id_readers, id_books, data_time):  
        self.id_books = id_books  
        self.id_readers = id_readers  
        self.data_time = data_time
```

Тестування на введення не коректних даних

```
select a menu item and enter the number:"
```

```
1 - Task_1
```

```
2 - Tast_2
```

```
3 - Task_3
```

```
4 - exit
```

```
Input:
```

```
1
```

```
Select number
```

```
1 - Insert
```

```
2 - DELL
```

```
3 - Update
```

```
4 - back
```

```
1
```

```
'name_table' : [coloms]
```

```
"books" : [ 'author','name_book','genry'],
```

```
"journal" :['id_readers','id_books','data_time'],
```

```
"id_genry" :[ 'genry'],
```

```
"readers" : ['name','surname','phone_number']
```

```
please, input for example:
```

```
[name_table new_value new_value new_value]
```

```
books Mark_tven 15_capitan 12
```

```
123
```

```
123
```

```
input correct data
```

```
error
```

```
please, input for example:
```

```
[name_table new_value new_value new_value]
```

```
books Mark_tven 15_capitan 12
```

```
books 123 123 123
```

```
error
```

```
"Hello,
```

```
books test 123 test 123
```

```
error
```

```
"Hello,
```

```

please, input for example:
[name_table new_value new_value new_value]
books    Mark_tven 15_capitan 12

books test etst test
error

```

```

'name_table' : [coloms]
"books" : [ 'author','name_book','genry'],
"journal" : ['id_readers','id_books','data_time'],
"id_genry" : [ 'genry'],
"readers" : ['name','surname','phone_number']

please, input for example:
[name_table new_value new_value new_value]
books    Mark_tven 15_capitan 12

journal tes tets ets
error

```

```

"books" : 'id_books',
            "journal" : 'id_records' ,
            "id_genry" : 'id_genry',
            "readers" : 'id_readers'
please, input for example:
books 99

teste 123
error
"Hello

```

```

2
"books" : 'id_books',
            "journal" : 'id_records' ,
            "id_genry" : 'id_genry',
            "readers" : 'id_readers'

please, input for example:
books 99

books test
input correct data or 4-exit

```

```

please, input for example:
name_table  set_columns  data          id
id_genry    genry        літопис        87

>? id genry test
input correct data or 4-exit

```

```

>? id_genry genry test test
Error
"Hello,
select a menu item and enter the number:"

```

```

please, input for example:
name_table  set_columns  data          id
id_genry    genry        літопис        87

>? readers test test test test teste teste
error
"Hello,

```

```

>? 3

'name_table' : [coloms]
"books" : [ 'author','name_book','genry'],
"journal" :['id_readers','id_books','data_time'],
"id_genry" :[ 'genry'],
"readers" : ['name','surname','phone_number']

please, input for example:
name_table  set_columns  data          id
id_genry    genry        літопис        87

>? readers teste tes te te te te 6
Error

```

```

>? 3

'name_table' : [coloms]
"books" : [ 'author','name_book','genry'],
"journal" :['id_readers','id_books','data_time'],
"id_genry" :[ 'genry'],
"readers" : ['name','surname','phone_number']

please, input for example:
name_table  set_columns  data          id
id_genry    genry        літопис        87

>? id_genry test 20046
input correct data or 4-exit
>? id_genry test 20046
input correct data or 4-exit

```

Введения коректних даних

```
'name_table' : [columns]
"books" : [ 'author','name_book','genry'],
"journal" :['id_readers','id_books','data_time'],
"id_genry" :[ 'genry'],
"readers" : ['name','surname','phone_number']
```

please, input for example:

```
[name_table new_value new_value new_value]
books    Mark_tven 15_capitan 12
```

```
journal 1 328 2020-03-12 06:00:00+02
```

```
"Hello,
```

```
select a menu item and enter the number:"
```

Результат План выполнения Сообщения

	id_records [PK] integer	id_readers integer	id_books integer	data_time timestamp with time zone
24	68	1	317	2020-07-09 07:13:14.70665+03
25	69	1	317	2021-10-19 00:28:25.089856+03
26	70	4	317	2020-12-12 10:49:37.294171+02
27	71	11	316	2020-03-12 06:00:00+02
28	73	11	318	2020-03-12 09:40:00+02
29	74	11	316	2020-03-12 06:00:00+02
30	78	2	328	2020-03-12 06:00:00+02
31	79	1	328	2020-03-12 06:00:00+02

```
1 SELECT * FROM public.books
2 ORDER BY id_boooks ASC
```

Результат План выполнения Сообщения

	id_boooks [PK] integer	author character varying	name_book character varying	genry integer
1	316	OX	OH	20028
2	317	YB	MS	20027
3	318	XG	JM	20028
4	320	NI	NH	20027
5	322	RJ	YF	20028
6	325	MO	OV	20028
7	327	DU	GW	20027
8	328	ON	NX	20027
9	334	WB	OP	20027
10	335	YC	QA	20027


```

"books" : 'id_books',
        "journal" : 'id_records' ,
        "id_genry" : 'id_genry',
        "readers" : 'id_readers'
please, input for example:
books 99

books 328
"Hello,
select a menu item and enter the number:"

        1 - Task_1
        2 - Tast_2
        3 - Task_3
        4 - exit

```

1	316	OX	OH	20028
2	317	YB	MS	20027
3	318	XG	JM	20028
4	322	RJ	YF	20028
5	325	MO	OV	20028
6	327	DU	GW	20027
7	328	ON	NX	20027
8	334	WB	OP	20027
9	336	VQ	VJ	20027
10	338	VS	VS	20028

	id_records [PK] integer	id_readers integer	id_books integer	data_time timestamp with time zone
1	41	5	318	2021-10-05 20:12:33.024766+03
2	42	1	317	2022-09-15 06:41:07.396061+03
3	43	6	317	2021-09-29 10:59:45.577792+03
4	44	1	316	2021-07-18 07:53:09.442359+03
5	45	6	317	2020-07-10 19:44:09.665991+03
6	46	6	317	2021-07-02 17:32:02.704773+03

Select number

- 1 - Insert
- 2 - DELL
- 3 - Update
- 4 - back

3

```
'name_table' : [coloms]
"books" : [ 'author','name_book','genry'],
"journal" :['id_readers','id_books','data_time'],
"id_genry" :[ 'genry'],
"readers" : ['name','surname','phone_number']
```

please, input for example:

```
name_table  set_columns  data          id
id_genry    genry        літопис      87
```

journal id_readers 6 id_books 317 data_time 2004-05-05 10:00 46

"Hello,

	id_records [PK] integer	id_readers integer	id_books integer	data_time timestamp with time zone
1	41	5	318	2021-10-05 20:12:33.024766+03
2	42	1	317	2022-09-15 06:41:07.396061+03
3	43	6	317	2021-09-29 10:59:45.577792+03
4	44	1	316	2021-07-18 07:53:09.442359+03
5	45	6	317	2020-07-10 19:44:09.665991+03
6	46	6	317	2021-07-02 17:32:02.704773+03

	id_records [PK] integer	id_readers integer	id_books integer	data_time timestamp with time zone
1	41	5	318	2021-10-05 20:12:33.024766+03
2	42	1	317	2022-09-15 06:41:07.396061+03
3	43	6	317	2021-09-29 10:59:45.577792+03
4	44	1	316	2021-07-18 07:53:09.442359+03
5	45	6	317	2020-07-10 19:44:09.665991+03
6	46	6	317	2004-05-05 10:00:00+03

	id_genry [PK] integer	genry character varying
1	20027	літопис
2	20028	VKCNL
3	20029	SVEBP
4	20030	LGIGV
5	20031	EBGGB
6	20032	HTGBS

```

'name_table' : [coloms]
"books" : [ 'author','name_book','genry'],
"journal" :['id_readers','id_books','data_time'],
"id_genry" :[ 'genry'],
"readers" : ['name','surname','phone_number']

```

please, input for example:

```

name_table  set_columns  data      id
id_genry    genry        літопис  87

```

```

> id_genry genry test 20028

```

```

"Hello,

```

	id_genry [PK] integer	genry character varying
1	20027	litopyys
2	20028	test

Отже можна прослідкувати те що всі функції delete, update, insert працюють вірно, що показано вище.

2. Индекси

Для створення Gin індексу було створено таблицю `readers_1`, з колонкою тип якої вимагає для роботи з даним тригером., та заповнено було за допомогою тригера дану таблицю

Створення таблиці.

```
CREATE TABLE public.readers_1
(
    id_readers integer NOT NULL GENERATED ALWAYS AS IDENTITY ( INCREMENT 1 START 1
MINVALUE 1 MAXVALUE 2147483647 CACHE 1 ),
    name character varying COLLATE pg_catalog."default",
    surname text COLLATE pg_catalog."default",
    phone_number integer,
    ts_vector tsvector,
    CONSTRAINT readers_pkey PRIMARY KEY (id_readers)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE public.readers
    OWNER to postgres;
```

Тригер заповнення

Имя	tsvectorupdate
Триггер включён?	Enable
Построчный триггер?	<input checked="" type="checkbox"/> Да
Триггер-ограничение?	<input type="checkbox"/> Нет
Откладываемое?	<input type="checkbox"/> Нет
Отложенное?	<input type="checkbox"/> Нет
Триггерная функция	
Аргументы	'ts_vector', 'pg_catalog.english', 'name', 'surname'

Срабатывает

BEFORE

События

INSERT

Да

UPDATE

Да

DELETE

Нет

TRUNCATE

Нет

Общие

Определение

События

Transition

Код

SQL

1 tsvector_update_trigger_byid

public.readers_1/library/postgres@PostgreSQL 11

Query Editor

История запросов

1 SELECT * FROM public.readers_1

2 ORDER BY id_readers ASC

Результат

План выполнения

Сообщения

	id_readers [PK] integer	name character varying	surname text	phone_number integer	ts_vector tsvector
267	268	RFDFU	IMCDX	20442102	'imcdx':2 'rfd...
268	269	RJSGT	LYVYB	68233383	'lyvyb':2 'rjs...
269	270	UGLPJ	OJJCU	91777956	'ojjcu':2 'ugl...
270	271	KYNNNS	CSOQN	57054846	'csoqn':2 'kyn...
271	272	YQKMH	ODAJH	94351861	'odajh':2 'yqk...
272	273	PYHNF	DCERE	72703387	'dcere':2 'pyh...
273	274	SGRNR	ONPSM	95488541	'onpsm':2 'sgr...

gin_idx

Общие

Определение

SQL

Имя

gin_idx

Табличное пространство

pg_default

Комментарий

Столбцы

Столбец	Класс операторов	Порядок сортировки	NULL	Правило сортировки
ts_vector		ASC	LAST	

```
1 explain analyze select name , ts_vector from readers_1 where ts_vector @@ to_tsquery('ter:*')
```

Notifications Результат План выполнения Сообщения

QUERY PLAN
text

```
1 Bitmap Heap Scan on readers_1 (cost=35.75..1599.00 rows=2000 width=38) (actual time=0.054....
2   Recheck Cond: (ts_vector @@ to_tsquery('ter:*':text))
3   Heap Blocks: exact=17
4   -> Bitmap Index Scan on gin_idx (cost=0.00..35.25 rows=2000 width=0) (actual time=0.044..0.04...
5       Index Cond: (ts_vector @@ to_tsquery('ter:*':text))
6 Planning Time: 0.166 ms
7 Execution Time: 0.142 ms
```

```
1 explain analyze select count (ts_vector) from readers_1 where ts_vector @@ to_tsquery('u:*')
```

Notifications Результат План выполнения Сообщения

QUERY PLAN
text

```
1 Aggregate (cost=1604.00..1604.01 rows=1 width=8) (actual time=8.358..8.360 rows=1 loops=1)
2   -> Bitmap Heap Scan on readers_1 (cost=35.75..1599.00 rows=2000 width=32) (actual time=3.4...
3       Recheck Cond: (ts_vector @@ to_tsquery('u:*':text))
4       Heap Blocks: exact=1031
5       -> Bitmap Index Scan on gin_idx (cost=0.00..35.25 rows=2000 width=0) (actual time=3.319..3...
6           Index Cond: (ts_vector @@ to_tsquery('u:*':text))
7 Planning Time: 0.161 ms
8 Execution Time: 8.524 ms
```

```
1 explain analyze select count (ts_vector) from readers_1 where ts_vector @@ to_tsquery('A:*')
```

Notifications Результат План выполнения Сообщения

QUERY PLAN
text

```
1 Aggregate (cost=1604.00..1604.01 rows=1 width=8) (actual time=10.724..10.725 rows=1 loops=1)
2   -> Bitmap Heap Scan on readers_1 (cost=35.75..1599.00 rows=2000 width=32) (actual time=5.2...
3       Recheck Cond: (ts_vector @@ to_tsquery('A:*':text))
4       Heap Blocks: exact=1031
5       -> Bitmap Index Scan on gin_idx (cost=0.00..35.25 rows=2000 width=0) (actual time=4.977..4...
6           Index Cond: (ts_vector @@ to_tsquery('A:*':text))
7 Planning Time: 0.164 ms
8 Execution Time: 10.869 ms
```

1	<code>explain analyze select * from readers_1 where ts_vector @@ to_tsquery('yrs:*') order by name;</code>
2	

Notifications	Результат	План выполнения	Сообщения
---------------	-----------	-----------------	-----------

	<div>QUERY PLAN</div> <div>text</div> <div></div>	
1	Sort (cost=1708.66..1713.66 rows=2000 width=52) (actual time=0.110..0.111 rows=16 loops=1)	
2	Sort Key: name	
3	Sort Method: quicksort Memory: 27kB	
4	-> Bitmap Heap Scan on readers_1 (cost=35.75..1599.00 rows=2000 width=52) (actual time=0.0...	
5	Recheck Cond: (ts_vector @@ to_tsquery('yrs:*'.text))	
6	Heap Blocks: exact=16	
7	-> Bitmap Index Scan on gin_idx (cost=0.00..35.25 rows=2000 width=0) (actual time=0.040..0...	
8	Index Cond: (ts_vector @@ to_tsquery('yrs:*'.text))	
9	Planning Time: 0.326 ms	
10	Execution Time: 0.155 ms	

1	<code>explain analyze select name from readers_1 where ts_vector @@ to_tsquery('ae:*') group by name;</code>
2	

Notifications	Результат	План выполнения	Сообщения
---------------	-----------	-----------------	-----------

	<div>QUERY PLAN</div> <div>text</div> <div></div>	
1	HashAggregate (cost=1604.00..1624.00 rows=2000 width=6) (actual time=1.159..1.256 rows=31...	
2	Group Key: name	
3	-> Bitmap Heap Scan on readers_1 (cost=35.75..1599.00 rows=2000 width=6) (actual time=0.31...	
4	Recheck Cond: (ts_vector @@ to_tsquery('ae:*'.text))	
5	Heap Blocks: exact=273	
6	-> Bitmap Index Scan on gin_idx (cost=0.00..35.25 rows=2000 width=0) (actual time=0.272..0...	
7	Index Cond: (ts_vector @@ to_tsquery('ae:*'.text))	
8	Planning Time: 0.253 ms	
9	Execution Time: 1.472 ms	

Даний тригер теж використовувався всюди оскільки вважається що він більш ефективніше. Також можна прослідкувати що випадки що запланований час в відрізняється від реального часу. це спричинено тим що не усюди ми використовували ту колонку по якій був створений індекс.

Також даний тригер зручно використовувати при пошуку слів чи частин слів у тексті.

Для індекса hash ми використовували також дану таблицю

Query Editor	История запросов
--------------	------------------

1	<code>create index hash_1 on readers using hash(name)</code>
---	--

Notifications	Результат	План выполнения	Сообщения
---------------	-----------	-----------------	-----------

CREATE INDEX
Запрос завершён успешно, время выполнения: 963 msec.

Query Editor История запросов

```
1 create index hash_2 on readers_1 using hash(name)
```

Notifications Результат План выполнения Сообщения

CREATE INDEX

Запрос завершён успешно, время выполнения: 666 msec.

library/postgres@PostgreSQL 11

Query Editor История запросов

```
1 explain analyze select * from readers_1 where name = 'TNLNK';
2
```

Notifications Результат План выполнения Сообщения

QUERY PLAN	
▲	text
1	Index Scan using hash_2 on readers_1 (cost=0.00..8.02 rows=1 width=52) (actual time=0.049..0.050 rows=1 loops=1)
2	Index Cond: ((name)::text = 'TNLNK'::text)
3	Planning Time: 1.262 ms
4	Execution Time: 0.084 ms

Query Editor История запросов

```
1 explain analyze select name from readers_1 where name = 'TNLNK' group by name;
2
```

Notifications Результат План выполнения Сообщения

QUERY PLAN	
▲	text
1	Group (cost=0.00..8.02 rows=1 width=6) (actual time=0.085..0.087 rows=1 loops=1)
2	Group Key: name
3	-> Index Scan using hash_2 on readers_1 (cost=0.00..8.02 rows=1 width=6) (actual time=0.082..0.083 rows=1 loops=1)
4	Index Cond: ((name)::text = 'TNLNK'::text)
5	Planning Time: 0.163 ms
6	Execution Time: 0.115 ms

library/postgres@PostgreSQL 11

Query Editor История запросов

1 explain analyze select name from readers_1 where name = 'TNLNK' order by name;

2

Notifications Результат План выполнения Сообщения

QUERY PLAN
text

1 Index Scan using hash_2 on readers_1 (cost=0.00..8.02 rows=1 width=6) (actual time=0.022..0.024 rows=1 loops=1)

2 Index Cond: ((name)::text = 'TNLNK'::text)

3 Planning Time: 0.176 ms

4 Execution Time: 0.049 ms

Даний індекс чітко показує що у всіх випадках він працює швидше тому що він використовується. Це спричинено тим що він індексував стовбець name по якому і відбувалась фільтрація, групування.

3 Завдання

Тригер який працює до функції delete and update.

На кожен функцію delete/update в тілі тригера відбуватимуться різні дії

При виконанні delete в таблицю LOOP записується та кількість одного і того ж запису але тої кількості скільки є в таблиці books елементів.

При виконанні update старі значення ми записуємо в таблицю trigger та йде перевірка на значення нового поля author чи не містить значення NULL якщо це так то вилітає виключення та виведення помилки 'author cannot null'.

Тригер

Код тіла тригерної функції:

DECLARE

stop integer := 0;

r cursor for select * from books;

Begin

IF TG_OP = 'DELETE' then

for books in r loop

insert into public."LOOP"(column3,genry) values('test', 20027);

stop := stop+1;

end loop;

return old;

END IF;

IF TG_OP = 'UPDATE' THEN

IF NEW.author is NULL then

```

        raise exception 'author cannot null';
    end IF;

    insert into trigger(id,colum1,colum2,colum3,colum4)
    values (trunc(random()*100)::int+step,
            (OLD.id_boooks), OLD.author,
            OLD.name_book,OLD.genry);

END IF;

return NEW;
END;
```

Результат работы тригера

Query Editor	История запросов
<pre> 1 UPDATE public.books 2 SET author = NULL, name_book=NULL, genry=20033 3 WHERE id_boooks = 324;</pre>	

Результат	План выполнения	Сообщения	Notifications
<pre> ERROR: author cannot null CONTEXT: PL/pgSQL function func_1() line 22 at RAISE SQL-состояние: P0001</pre>			

Таблица books до update

	id_boooks [PK] integer	author character varying	name_book character varying	genry integer
1	316	OX	OH	20028
2	317	YB	MS	20027
3	318	XG	JM	20028
4	319	KI	EH	20033
5	320	NI	NH	20027
6	322	RJ	YF	20028
7	323	TB	TQ	20027
8	324	VF	FL	20027
9	325	MO	OV	20028
10	327	DU	GW	20027
11	328	ON	NX	20027
12	329	LE	UX	20028

library/postgres@PostgreSQL 11

Query Editor История запросов

```

1 UPDATE public.books
2     SET  author = 'test', name_book='test', genry=20033
3     WHERE id_boooks = 324;
```

Результат План выполнения Сообщения Notifications







UPDATE 1

Запрос завершён успешно, время выполнения: 357 msec.

Таблица books після update

	id_boooks [PK] integer	author character varying	name_book character varying	genry integer
1	316	OX	OH	20028
2	317	YB	MS	20027
3	318	XG	JM	20028
4	319	KI	EH	20033
5	320	NI	NH	20027
6	322	RJ	YF	20028
7	324	test	test	20033
8	325	MO	OV	20028
9	327	DU	GW	20027

Таблиця trigger

	 id [PK] integer 	column2 character varying 	column3 character varying 	column1 integer 	column4 integer 	
1	1	AZ	CU	340	20028	
2	25	DR	UT	321	20027	
3	32	AZ	Update	340	20028	
4	55	ДЮМА	три_мушкетери	342	[null]	
5	82	ДЮМА	три_мушкетери	341	[null]	
6	88	AZ	Update	340	20028	
7	407	VF	FL	324	20027	

Таким чином , тригер виконує свою функція і працює стабільно.