

Министерство науки и высшего образования Российской Федерации Федеральное
государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО» Факультет
инфокоммуникационных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

по теме: Запросы на выборку и модификацию данных, представления и индексы в
PostgreSQL

Специальность:

09.03.03 Мобильные и сетевые технологии

Проверил:

Говорова М.М. _____

Выполнил:

студент группы К3240 Ковалев В.М.

Санкт-Петербург 2022

ЦЕЛЬ РАБОТЫ

Овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, pgadmin 4.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов и просмотреть историю запросов
4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.

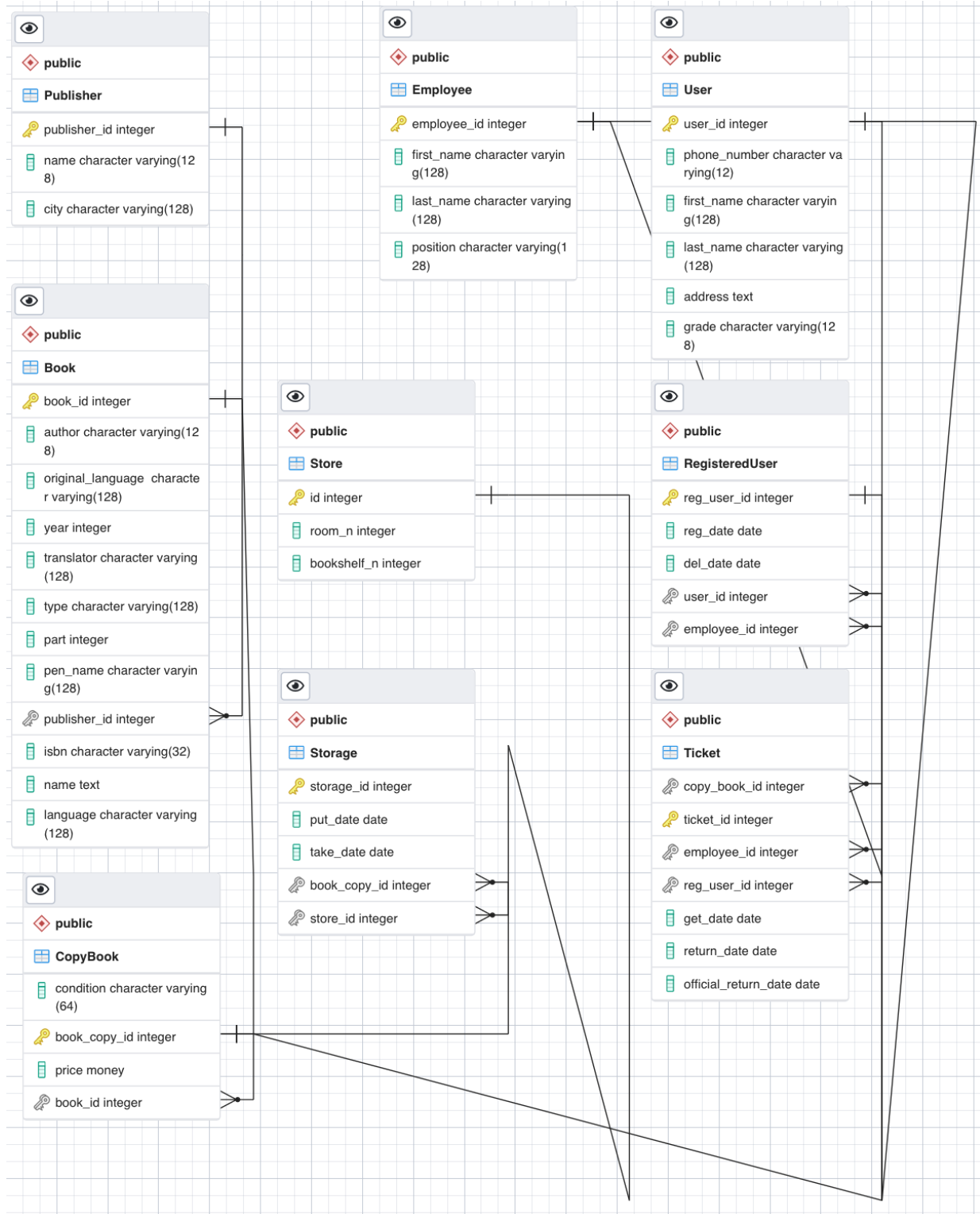
Вариант 3

Библиотека

Описание предметной области: Каждая книга может храниться в нескольких экземплярах. Для каждого экземпляра известно место его хранения (комната, стеллаж, полка). Читателю не может быть выдано более 3-х книг одновременно. Книги выдаются читателям на срок не более 10 дней. БД должна содержать следующий минимальный набор сведений: · Автор (фамилия и имя (инициалы) или псевдоним автора издания). · Название (заглавие) издания. · Номер тома (части, книги, выпуска). · Составитель (фамилия и имена (инициалы) каждого из составителей издания). · Язык, с которого выполнен перевод издания. · Вид издания (сборник, справочник, монография ...). · Область знания. · Переводчик (фамилия и инициалы переводчика). · Место издания (город). · Издательство (название издательства). · Год выпуска издания. · Библиотечный шифр (например, ББК 32.973). · Номер (инвентарный номер) экземпляра. · Номер комнаты (помещения для хранения экземпляров). · Номер стеллажа в комнате. · Номер полки на стеллаже. · Цена конкретного экземпляра. · Дата изъятия экземпляра с установленного места. · Номер читательского билета (формуляра). · Фамилия читателя. · Имя читателя. · Отчество читателя. · Адрес читателя. Телефон читателя.

Дополнить исходные данные информацией о читательском абонементе (выдаче книг).

Рисунок 1. ER-диаграмма



Задание 1.

1. Вывести список читателей, имеющих на руках книги, переведенные с английского языка, изданные позднее 2000 года.

```
SELECT name, year, first_name, last_name, phone_number, return_date,
"original_language ", language FROM "Book" AS book RIGHT JOIN
(SELECT book_id, first_name, last_name, phone_number, return_date FROM "CopyBook"
as copy_book RIGHT JOIN
(SELECT copy_book_id, first_name, last_name, phone_number, return_date FROM
"Ticket" AS ticket LEFT JOIN
(SELECT reg_user_id, first_name, last_name, phone_number FROM "RegisteredUser" AS
rg
LEFT JOIN "User" AS u
ON rg.user_id = u.user_id) AS reg_users
ON reg_users.reg_user_id = ticket.reg_user_id) AS tickets
ON copy_book.book_copy_id = tickets.copy_book_id) AS copy_books
ON book.book_id = copy_books.book_id WHERE "original_language " = 'EN' AND
"language" != 'EN' AND "return_date" IS NULL AND "year" >=2000;
```

Data Output

	name text	year integer	first_name character varying (128)	last_name character varying (128)	phone_number character varying (12)	return_date date	original_language character varying (128)	language character varying (128)
1	Tigers	2001	Vasya	Petrov	+79098674455	[null]	EN	RU
2	Tigers	2001	Anna	Vorobyeva	+79098674456	[null]	EN	RU
3	Tigers	2001	Vladimir	Pubin	+79098674460	[null]	EN	RU

2. Вывести список читателей, не вернувших в срок книги и имеющих на руках более десяти книг.

```
SELECT t1.reg_user_id, COUNT(t1.return_date) FROM "Ticket" AS t1
LEFT JOIN "RegisteredUser" AS t2 ON t1.reg_user_id = t2.reg_user_id
WHERE "official_return_date"<"return_date" GROUP BY t1.reg_user_id;
```

Data Output

	reg_user_id integer	count bigint
1	86	13

3. Найти количество читателей, не вернувших в срок книги и имеющих на руках более десяти книг

```
SELECT COUNT(reg_user_id) FROM (SELECT t1.reg_user_id, COUNT(t1.return_date) FROM "Ticket" AS t1
LEFT JOIN "RegisteredUser" AS t2 ON t1.reg_user_id = t2.reg_user_id
WHERE "official_return_date"<"return_date" GROUP BY t1.reg_user_id) AS t;
```

Data Output

	count bigint
1	1

4. Вывести список книг, которые находятся в библиотеке в единственном экземпляре.

```
SELECT * FROM (SELECT book_id, COUNT(book_id) FROM "CopyBook" GROUP BY book_id) AS t1 WHERE count =1;
```

Data Output

	book_id integer	count bigint
1	21	1
2	22	1

5. Подсчитать количество читателей, которые не обращались в библиотеку в течение года

```
SELECT COUNT(*) FROM (
SELECT DISTINCT reg_user_id FROM "Ticket"
WHERE get_date < '2022-01-01'
OR return_date < '2022-01-01'
GROUP BY reg_user_id) AS T1;
```

	count bigint
1	31

6. Подсчитать количество читателей библиотеки по уровню образования

```
SELECT grade, COUNT(*) FROM "RegisteredUser" AS rg LEFT JOIN "User" AS u ON rg.user_id=u.user_id GROUP BY grade;
```

Data Output

	grade character varying (128)	count bigint
1	Bachelor	16
2	Master	9
3	Schoolchild	11
4	Student	11

7. Вывести список книг по программированию на C#, экземпляры которых отсутствуют в библиотеке, и которые должны быть возвращены не позднее, чем через 3 дня.

Сегодня 2022-01-31

```
SELECT name, official_return_date, return_date FROM "Book" as t4 RIGHT JOIN (
    SELECT * FROM "CopyBook" AS t2 RIGHT JOIN (
        SELECT * FROM "Ticket"
        WHERE return_date IS NULL
        AND official_return_date-'2022-01-31'<=3
        AND official_return_date-'2022-01-31'>0) AS t1 ON
t1.copy_book_id=t2.book_copy_id) AS t3 ON t4.book_id=t3.book_id
WHERE t4.name LIKE '%C#%';
```

	name text	official_return_date date	return_date date
1	C# Guide	2022-02-02	[null]
2	C# Guide	2022-02-03	[null]

Задание 2.

Создать представления для администрации библиотеки, содержащие:

1. сведения о должниках

```
CREATE VIEW CREDITS_USER_LIST AS
SELECT t4.*, t3.credits FROM "User" t4 RIGHT JOIN
(SELECT t2.reg_user_id, t2.user_id, COUNT(t1.return_date) AS credits FROM
"Ticket" AS t1
LEFT JOIN "RegisteredUser" AS t2 ON t1.reg_user_id = t2.reg_user_id
WHERE "official_return_date"<"return_date" GROUP BY t2.reg_user_id) AS t3
ON t3.user_id=t4.user_id;
```

Запрос:

```
SELECT * FROM credits_user_list;
```

	user_id integer	phone_number character varying (12)	first_name character varying (128)	last_name character varying (128)	address text	grade character varying (128)	credits bigint
1	126	77613451233	Anastasia	Zaytseva	Random_street, 844	Schoolchild	13

2. сведения о наиболее популярных книгах (все экземпляры находятся на руках у читателей)

```
CREATE VIEW BOOK_TOP_LIST AS
SELECT t4.name, t4.part, t3.book_id, COUNT(t3.book_copy_id) FROM "Book" AS t4 RIGHT
JOIN (
    SELECT * FROM "CopyBook" AS t2 RIGHT JOIN (
        SELECT * FROM "Ticket" WHERE return_date IS NULL) AS t1
    ON t1.copy_book_id=t2.book_copy_id) AS t3
    ON t3.book_id=t4.book_id GROUP BY (t3.book_id, t4.name, t4.part) ORDER BY
count DESC;
```

Запрос:

```
SELECT * FROM book_top_list;
```

Data Output

	name text	part integer	book_id integer	count bigint
1	Who I Am?	2	16	5
2	Spider-Man	1	4	4
3	C# Guide	1	1	3
4	Monkeys	2	12	3
5	Math	2	17	3
6	Fresh watermelons	1	9	3
7	Along	2	15	2
8	Economics	2	18	2
9	Along	1	5	2
10	Spider-Man	2	14	1
11	Math	1	7	1
12	Fresh watermelons	2	19	1
13	Cockroaches. Bees. Ladybugs.	2	20	1
14	Cockroaches. Bees. Ladybugs.	1	10	1
15	Venom	1	3	1

Задание 3.

1. Запрос на модификацию данных с INSERT (добавление новой книги):

```
INSERT INTO "Book" (book_id, "original_language ", year, type, part, publisher_id,  
isbn, name, language, author)  
SELECT 25, 'EN', 2010, 'Guide', 1, 1, 6434234, 'SQL for kids', 'EN', 'Genry';
```

Проверка:

```
SELECT * FROM "Book" WHERE name = 'SQL for kids';
```

Data Output									
book_id [PK] integer	author character varying (128)	original_language character varying (128)	year integer	transl char	type character varying (128)	part integer	publ char	publis integer	isbn character varying (32)
25	Genry	EN	2010	[null]	Guide	1	[null]	1	6434234

2. Запрос на модификацию данных с UPDATE (продлить всем читателям, у которых на руках книга, дату обязательной ее сдачи на неделю):

```
UPDATE "Ticket"  
SET official_return_date = official_return_date + INTERVAL '7 DAYS'  
WHERE return_date IS NULL;
```

Проверка (изначально дата обязательной сдачи была через два месяца после получения книги):

59	16	68	2	56	2022-01-10	[null]	2022-03-17
60	17	69	2	57	2022-01-15	[null]	2022-03-22
61	149	70	2	58	2022-01-07	[null]	2022-03-14
62	150	71	2	60	2022-01-15	[null]	2022-03-22
63	24	72	2	61	2022-01-14	[null]	2022-03-21
64	25	73	2	62	2022-01-07	[null]	2022-03-14
65	26	74	2	63	2022-01-08	[null]	2022-03-15
66	155	75	2	64	2022-01-12	[null]	2022-03-19

3. Запрос на модификацию данных с DELETE (удалить неиспользуемые пустые книжные полки):

```
DELETE FROM "Store" WHERE id IN (SELECT id FROM "Storage" AS storage
RIGHT JOIN "Store" AS store
ON storage.store_id = store.id
WHERE storage_id IS NULL);
```

Проверка:

Полки, которыми никогда не пользовались:

```
SELECT storage_id, id AS store_id FROM "Storage" AS storage
RIGHT JOIN "Store" AS store
ON storage.store_id = store.id
WHERE storage_id IS NULL;
```

Data Output

	storage_id integer	store_id integer
1	[null]	15
2	[null]	16
3	[null]	14

Теперь их нет:

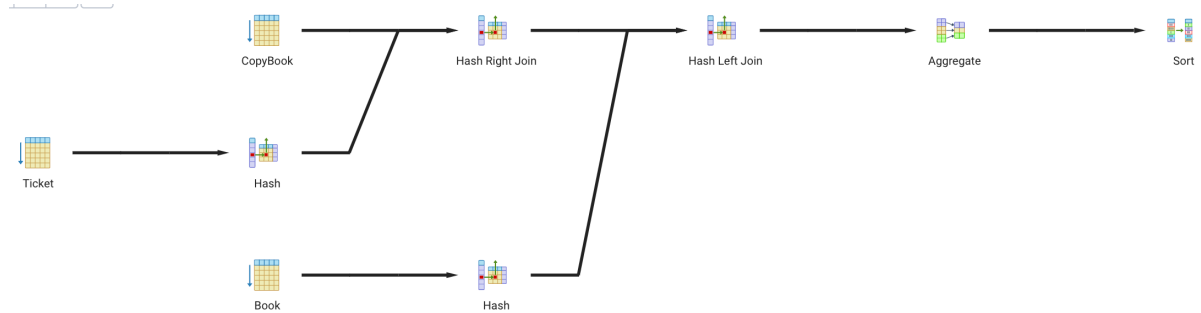
```
SELECT storage_id, id AS store_id FROM "Storage" AS storage
RIGHT JOIN "Store" AS store
ON storage.store_id = store.id
WHERE storage_id IS NULL;
```

Data Output

	storage_id integer	store_id integer

Задание 4.

Графическое представление запроса `SELECT * FROM book_top_list;`



История запросов:

Query History	
Show queries generated internally by pgAdmin? <input type="button" value="No"/>	
Today - 20/04/2022	
	<code>SELECT * FROM book_top_list;</code> 17:14:01
	<code>EXPLAIN SELECT * FROM book_top_list;</code> 17:12:45
	<code>SELECT * FROM book_top_list;</code> 17:09:06
	<code>EXPLAIN SELECT * FROM book_top_list;</code> 17:08:56
	<code>EXPLAIN SELECT * FROM book_top_list;</code> 17:06:49
	<code>EXPLAIN SELECT * FROM book_top_list;</code> 17:06:42
	<code>EXPLAIN SELECT * FROM book_top_list;</code> 17:04:22
	<code>SELECT storage_id, id AS store_id FROM "Storage" AS storage R...</code> 17:00:08
	<code>SELECT storage_id, id AS store_id FROM "Storage" AS storage R...</code> 16:57:26
	<code>SELECT storage_id, id FROM "Storage" AS storage RIGHT JOIN "S...</code> 16:56:58
	<code>SELECT id FROM "Storage" AS storage RIGHT JOIN "Store" AS sto...</code> 16:56:44

Задание 5.

До создания простого индекса:

```
EXPLAIN SELECT * FROM "Ticket" WHERE return_date > '2015-10-19';
```

Successfully run. Total query runtime: 53 msec.

26 rows affected.

QUERY PLAN		
	text	
1	Seq Scan on "Ticket" (cost=0.00..1.98 rows=27 width=28)	
2	[...] Filter: (return_date > '2015-10-19'::date)	

```
CREATE INDEX date_of_return ON "Ticket"(return_date);
```

После:

```
EXPLAIN SELECT * FROM "Ticket" WHERE return_date > '2015-10-19';
```

Successfully run. Total query runtime: 70 msec.


2 rows affected.

QUERY PLAN		
	text	
1	Seq Scan on "Ticket" (cost=0.00..1.98 rows=27 width=28)	
2	[...] Filter: (return_date > '2015-10-19'::date)	

До создания составного индекса:

```
EXPLAIN SELECT * FROM "User" WHERE first_name LIKE 'A%' AND last_name LIKE '%a';
```

Successfully run. Total query runtime: 45 msec. 2 rows affected.

	QUERY PLAN	
	text	
1	Seq Scan on "User" (cost=0.00..5.97 rows=2 width=56)	
2	[...] Filter: (((first_name)::text ~~ 'A% '::text) AND ((last_name)::text ~~ '%a '::text))	

```
CREATE INDEX full_name ON "User"(first_name, last_name);
```

После:

```
EXPLAIN SELECT * FROM "User" WHERE first_name LIKE 'A%' AND last_name LIKE '%a';
```

Successfully run. Total query runtime: 48 msec. 2 rows affected.

QUERY PLAN	
text	
1	Seq Scan on "User" (cost=0.00..5.97 rows=2 width=56)
2	[...] Filter: (((first_name)::text ~~ 'A%':text) AND ((last_name)::text ~~ '%a':text))

Удаление индексов:

Column	Type	Table "public.User"			Default
		Collation	Nullable		
user_id	integer		not null		generated by default as identity
phone_number	character varying(12)		not null		
first_name	character varying(128)		not null		
last_name	character varying(128)		not null		
address	text		not null		
grade	character varying(128)		not null		
Indexes:					
"User_pkey" PRIMARY KEY, btree (user_id)					
"full_name" btree (first_name, last_name)					
Referenced by:					
TABLE ""RegisteredUser"" CONSTRAINT "User_id" FOREIGN KEY (user_id) REFERENCES "User"(user_id)					

```
DROP INDEX full_name;
```

Column	Type	Table "public.User"			Default
		Collation	Nullable		
user_id	integer		not null		generated by default as identity
phone_number	character varying(12)		not null		
first_name	character varying(128)		not null		
last_name	character varying(128)		not null		
address	text		not null		
grade	character varying(128)		not null		
Indexes:					
"User_pkey" PRIMARY KEY, btree (user_id)					
Referenced by:					
TABLE ""RegisteredUser"" CONSTRAINT "User_id" FOREIGN KEY (user_id) REFERENCES "User"(user_id)					

Вывод:

В ходе выполнения лабораторной работы мною были выполнены запросы из своей базы данных. Созданы представления на администрации библиотеки и сделаны запросы на модификацию данных. Также созданы индексы. После создания индекса, время выполнения запроса увеличилось