

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

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 4
по теме: Запросы на выборку и модификацию данных,
представления и индексы в PostgreSQL
по дисциплине: Проектирование и реализация баз данных

Специальность:
09.03.03 Мобильные и сетевые технологии

Проверил:
Говорова М.М. _____
Дата: «__» _____ 20__ г.
Оценка _____

Выполнил:
студент группы К3240
Чернов Е. К.

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.

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 посмотреть 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 1. БД «Отель»

Описание предметной области: Отели находятся в разных городах. Цены на номера одного типа во всех отелях одинаковы и зависят от типа номера и количества мест. Номер может быть забронирован, занят или свободен. При заезде в отель постояльцы проходят регистрацию. Информация о регистрации постояльцев отеля (выехавших из отеля) хранится в течение года и 1 января удаляется в архив.

БД должна содержать следующий минимальный набор сведений: Адрес отеля. Название отеля. Номер комнаты. Тип комнаты. Количество мест. Цена комнаты за сутки проживания. Имя постояльца. Фамилия постояльца. Отчество постояльца. Адрес постоянного проживания. Дата заезда. Дата отъезда.

The ER diagram illustrates the following tables and their attributes:

- public.room_type**: id_room_type integer (PK), rm_type character varying(20), facilities character varying(255), room_amount integer.
- public.hotel**: id_hotel integer (PK), city character varying(30), name_hotel character varying(100), address character varying(255).
- public.registration**: id_reg integer (PK), personnel_number integer, id_room integer, passport character varying(10), status_reg character varying(20), status_pay character varying(20), booking_date timestamp without time zone, check_in timestamp without time zone, check_out timestamp without time zone.
- public.price**: id_price integer (PK), end_date timestamp without time zone, seats_number integer, start_date timestamp without time zone, price_per_day double precision, id_room_type integer (FK).
- public.promotions**: id_promotions integer (PK), id_room_type integer, end_d timestamp without time zone, description character varying(255), start_d timestamp without time zone.
- public.room**: id_room integer (PK), id_room_type integer (FK), id_hotel integer (FK), room_number integer, status character varying(20).
- public.roomer**: passport character varying(10) (PK), phone_num character varying(12), home_address character varying(255), f_name character varying(20), surname_roomer character varying(30), l_name character varying(30).
- public.worker**: personnel_number integer (PK), phone character varying(12), first_name character varying(20), surname character varying(30), last_name character varying(30).

Relationships are defined as follows:

- room_type** to **hotel**: One-to-many relationship on the `id_hotel` foreign key.
- registration** to **room_type**: Many-to-one relationship on the `id_room` foreign key.
- registration** to **worker**: Many-to-one relationship on the `personnel_number` foreign key.
- price** to **room_type**: Many-to-one relationship on the `id_room_type` foreign key.
- promotions** to **room_type**: Many-to-one relationship on the `id_room_type` foreign key.
- room** to **hotel**: Many-to-one relationship on the `id_hotel` foreign key.
- room** to **room_type**: Many-to-one relationship on the `id_room_type` foreign key.
- roomer** to **room**: Many-to-one relationship on the `id_room` foreign key.
- roomer** to **registration**: Many-to-one relationship on the `passport` foreign key.

3

ВЫПОЛНЕНИЕ

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3);

Запросы на выборку данных:

- Составить список всех 2-местных номеров отелей, с ценой менее 200 т.р., упорядочив данные в порядке уменьшения стоимости:

```
1 select distinct r.id_hotel, r.id_room, r.room_number, rt.rm_type, rt.facilities, adt.price
2 from room r, room_type rt, hotel h,
3 (select distinct id_room_type, (price_per_day / (date_part('day', end_date - start_date) + 1))
4 as price from price order by id_room_type) adt
5 where
6     h.id_hotel = r.id_hotel and
7     r.id_room_type = rt.id_room_type and
8     adt.id_room_type = r.id_room_type and
9     rt.rm_type in('standart', 'vip')
10 order by price;
```








Data Output		Explain	Messages	Notifications		
 id_hotel integer	 id_room integer	 room_number integer	 rm_type character varying (20)	 facilities character varying (255)	 price double precision	
1	2	12	11	standart	TV, Wi-Fi	2000
2	2	13	22	standart	TV, Wi-Fi	2000
3	4	4	11	standart	Телевизор, интернет	3000
4	4	5	22	standart	Телевизор, интернет	3000
5	4	6	33	standart	Телевизор, интернет	3000
6	4	7	111	vip	Телевизор, интернет, джакузи	5000
7	4	8	222	vip	Телевизор, интернет, джакузи	5000
8	4	9	333	vip	Телевизор, интернет, джакузи	5000
9	2	14	111	vip	TV, Wi-Fi, swimming pool	5500
10	2	15	222	vip	TV, Wi-Fi, swimming pool	5500

Рисунок 2 - Задание 1.1

- Выбрать все записи регистрации постояльцев, которые выехали из отелей в течение последней недели:

```
select * from registration where check_out >= '20.03.2022';
```

	id_reg [PK] integer	personnel_number integer	id_room integer	passport character varying (10)	status_reg character varying (20)	status_pay character varying (20)	booking_date timestamp without time zone	check_in timestamp without time zone	check_out timestamp without time zone
1	32	2	3	4432342379	зарегистрирован	оплачено	2022-03-18 00:00:00	2022-03-18 00:00:00	2022-03-20 00:00:00
2	33	2	5	4439772379	зарегистрирован	оплачено	2022-03-13 00:00:00	2022-03-18 00:00:00	2022-03-20 00:00:00
3	35	1	1	4400342379	зарегистрирован	оплачено	2022-03-18 00:00:00	2022-03-19 00:00:00	2022-03-20 00:00:00
4	36	1	2	4465772379	зарегистрирован	оплачено	2022-03-13 00:00:00	2022-03-19 00:00:00	2022-03-20 00:00:00
5	37	1	6	4498812379	зарегистрирован	оплачено	2022-03-16 00:00:00	2022-03-19 00:00:00	2022-03-20 00:00:00
6	38	3	2	4948342379	зарегистрирован	оплачено	2022-03-20 00:00:00	2022-03-21 00:00:00	2022-03-23 00:00:00
7	39	3	4	4432342301	зарегистрирован	оплачено	2022-03-11 00:00:00	2022-03-21 00:00:00	2022-03-23 00:00:00
8	40	3	7	4499942379	зарегистрирован	оплачено	2022-03-19 00:00:00	2022-03-21 00:00:00	2022-03-22 00:00:00
9	41	2	1	5948342379	зарегистрирован	оплачено	2022-03-20 00:00:00	2022-03-22 00:00:00	2022-03-26 00:00:00
10	42	2	3	6432342301	зарегистрирован	оплачено	2022-03-11 00:00:00	2022-03-22 00:00:00	2022-03-25 00:00:00
11	43	2	8	7499942379	зарегистрирован	оплачено	2022-03-19 00:00:00	2022-03-22 00:00:00	2022-03-25 00:00:00
12	44	1	6	5948345379	зарегистрирован	оплачено	2022-03-21 00:00:00	2022-03-23 00:00:00	2022-03-24 00:00:00
13	45	4	5	5948315379	зарегистрирован	оплачено	2022-03-22 00:00:00	2022-03-24 00:00:00	2022-03-26 00:00:00
14	46	3	4	5948315349	зарегистрирован	оплачено	2022-03-23 00:00:00	2022-03-25 00:00:00	2022-03-26 00:00:00
15	47	3	7	5948315329	зарегистрирован	оплачено	2022-03-24 00:00:00	2022-03-25 00:00:00	2022-03-27 00:00:00
16	48	2	2	1028763678	зарегистрирован	оплачено	2022-03-25 00:00:00	2022-03-26 00:00:00	2022-03-27 00:00:00
17	49	2	3	1023963678	зарегистрирован	оплачено	2022-03-22 00:00:00	2022-03-26 00:00:00	2022-03-27 00:00:00
18	50	2	6	1024012678	зарегистрирован	оплачено	2022-03-23 00:00:00	2022-03-26 00:00:00	2022-03-27 00:00:00
19	51	2	8	4039872378	зарегистрирован	оплачено	2022-03-18 00:00:00	2022-03-26 00:00:00	2022-03-27 00:00:00
20	52	6	12	5928315349	зарегистрирован	оплачено	2022-03-17 00:00:00	2022-03-20 00:00:00	2022-03-22 00:00:00

Рисунок 3.1 - Задание 1.2.1

Без магических дат:

1

select * from registration where check_out >= (now() - interval '7 day');

Data Output

Explain

Messages

Notifications

	id_reg [PK] integer	 personnel_number integer	 id_room integer	 passport character varying (10)	 status_reg character varying (20)	 status_pay character varying (20)	booking_date timestamp without time zone	 check_in timestamp without time zone	 check_out timestamp without time zone	check_time time

Рисунок 3.2 - Задание 1.2.2

- Чему равен общий суточный доход каждого отеля за последнюю неделю?

```

1 select adt.id_hotel, (sum(adt.price_per_day) / 7) as total from
2 (select DISTINCT pr.id_price, pr.price_per_day, pr.id_room_type, r.id_hotel
3  from price pr inner join room r on pr.id_room_type = r.id_room_type
4  where pr.start_date > '20.03.2022' order by pr.id_price) adt
5 group by adt.id_hotel;

```

Data Output		Explain	Messages	Notifications
	id_hotel integer	total double precision		
1	4	16428.5714285714		
2	2	9214.28571428571		

Рисунок 4.1 - Задание 1.3.1

Без магических дат:

```

1 select adt.id_hotel, (sum(adt.price_per_day) / 7) as total from
2 (select DISTINCT pr.id_price, pr.price_per_day, pr.id_room_type, r.id_hotel
3  from price pr inner join room r on pr.id_room_type = r.id_room_type
4  where pr.start_date > (now() - interval '7 day') order by pr.id_price) adt
5 group by adt.id_hotel;

```

Data Output		Explain	Messages	Notifications
	id_hotel integer	total double precision		

Рисунок 4.2 - Задание 1.3.2

- Составить список свободных номеров одного из отелей на текущий день:

```
1 select * from room where status = 'свободен';
```







Data Output		Explain	Messages	Notifications	
	id_room [PK] integer 	id_room_type integer 	id_hotel integer 	room_number integer 	status character varying (20) 
1	2	1	4	2	свободен
2	3	1	4	3	свободен
3	6	2	4	33	свободен
4	7	3	4	111	свободен
5	8	3	4	222	свободен
6	10	4	2	1	свободен
7	12	5	2	11	свободен
8	13	5	2	22	свободен
9	15	6	2	222	свободен

Рисунок 5 - Задание 1.4

- Найти общие потери от незанятых номеров за текущий день по всей сети:

Расценки для отеля с id_hotel = 4 и id_hotel = 2:

4	Цена-сутки	Обслуживание-сутки	2	Цена-сутки	Обслуживание-сутки
econom	1500	1000	econom	1200	1000
standart	3000	2000	standart	2000	2000
vip	5000	3500	vip	5500	3500

Рисунок 6 - Цены на номера

```

1 select sum(case
2 when id_room_type in(1, 4) then 1000
3 when id_room_type in(2, 5) then 1000
4 else 3500
5 end) from room where status='занят';

```

Data Output Explain Messages Notifications

	sum bigint
1	14500

Рисунок 7 - Задание 1.5

- Определить, в каком отеле имеется наибольшее количество незанятых номеров на текущие сутки:

```

1 select id_hotel from
2 (select id_hotel, count(id_hotel) from room where status = 'свободен' group by id_hotel) adt
3 where adt.count = (select max(adadt.count) from
4 (select id_hotel, count(id_hotel) from
5 room where status = 'свободен' group by id_hotel) adadt);

```

Data Output Explain Messages Notifications

	id_hotel integer
1	4

Рисунок 8 - Задание 1.6

- Определить самый популярный тип номеров за последний месяц:

```

1 select rm_type from room_type rt, (select id_room_type from
2 (select id_room_type, count(start_date) from price
3 where start_date >= (now() - '1 month'::interval) group by id_room_type order by id_room_type) adt
4 where adt.count = (select max(adadt.count) from
5 (select id_room_type, count(start_date) from
6 price where start_date >= (now() - '1 month'::interval) group by id_room_type order by id_room_type) adadt)) adt
7 where rt.id_room_type = adt.id_room_type;

```

Data Output Explain Messages Notifications

	rm_type character varying (20)
1	econom

Рисунок 9 - Задание 1.7

Создание представлений:

- Для турагентов (поиск свободных номеров в отелях):

1

```
select * from tour_agent;
```

Data Output

Explain

Messages

Notifications

	<div>id_hotel</div> <div>integer</div>	<div>id_room</div> <div>integer</div>	<div>room_number</div> <div>integer</div>	<div>rm_type</div> <div>character varying (20)</div>
1	4	2	2	econom
2	4	3	3	econom
3	4	6	33	standart
4	4	7	111	vip
5	4	8	222	vip
6	2	10	1	econom
7	2	12	11	standart
8	2	13	22	standart
9	2	15	222	vip

create view tour_agent as

select r.id_hotel, r.id_room, r.room_number, rt.rm_type from room r, room_type rt

where status = 'свободен' and r.id_room_type = rt.id_room_type order by id_room;

Рисунок 10 - Задание 1.8

- Для владельца компании (информация о доходах каждого отеля в сети за прошедшую неделю):

Query Editor	Query History	Scratch Pad	×
1	<code>select * from profit_week;</code>	<code>create view profit_week as select adt.id_hotel, sum(adts.price_per_day) as total from (select DISTINCT pr.id_price, pr.price_per_day, pr.id_room_type, r.id_hotel from price pr inner join room r on pr.id_room_type = r.id_room_type where pr.start_date > '20.03.2022' order by pr.id_price) adt group by adt.id_hotel;</code>	
Data Output	Explain	Messages	Notifications
	<code>id_hotel</code> integer	<code>total</code> double precision	
1	4	115000	
2	2	64500	

Рисунок 11.1 - Задание 1.9.1

Без магических дат:

1

```
select * from profit_week;
```

Data Output

Explain

Messages

Notifications

id_hotel

integer

total

double precision

create view profit_week as

select adt.id_hotel, sum(adts.price_per_day) as total from

(select DISTINCT pr.id_price, pr.price_per_day, pr.id_room_type, r.id_hotel from

price pr inner join room r on pr.id_room_type = r.id_room_type

where pr.start_date > (now() - interval '7 day') order by pr.id_price) adt group by adt.id_hotel;

Рисунок 11.2 - Задание 1.9.2

2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов;

INSERT запрос, который добавляет строку, основываясь на только что добавленной регистрации:

```
1 insert into price (id_room_type, start_date, end_date, seats_number, price_per_day)
2 select r.id_room_type, rg.check_in, rg.check_out, 2 as seats_number, 11000 as price_per_day from registration rg, room r where
3 (select max(id_reg) from registration) = rg.id_reg and rg.id_room = r.id_room;
```

Data Output Explain Messages Notifications

INSERT 0 1

Query returned successfully in 47 msec.

Рисунок 12 - Задание 2.1

UPDATE запрос, который обновляет статус комнаты, после регистрации:

```
1 update room set status='занят'
2 where id_room=(select id_room from registration r where id_reg=(select max(id_reg) from registration));
```

Data Output Explain Messages Notifications

UPDATE 1

Query returned successfully in 45 msec.

Рисунок 13 - Задание 2.2

DELETE запрос удаляет строку, которая несет информацию о цене регистрации номера, за который не заплатили:

```
1 delete from price
2 where start_date=(select check_in from registration where check_in='27.03.2022' and status_pay!='оплачено')
3 and end_date=(select check_out from registration where check_in='27.03.2022' and status_pay!='оплачено');
```

Data Output Explain Messages Notifications

DELETE 1

Query returned successfully in 27 msec.

Рисунок 14 - Задание 2.3

3. Изучить графическое представление запросов и посмотреть историю запросов;



Рисунок 15 - Просмотр графического представления запроса

hotel/postgres@PostgreSQL 10

Query Editor Query History

Show queries generated internally by pgAdmin? ☒ Yes

Today - 30.03.2022

20:27:46
select rm_type from room_type rt, (select id_room_type from (select...

20:25:58
select rm_type from room_type rt, (select id_room_type from (select...

20:25:58
select rm_type from room_type rt, (select id_room_type from (select...

20:24:25
explain select rm_type from room_type rt, (select id_room_type from...

20:24:05
select rm_type from room_type rt, (select id_room_type from (select...

20:23:58
explain select rm_type from room_type rt, (select id_room_type from...

20:23:54
explain query select rm_type from room_type rt, (select id_room_t...

20:23:49
explain [query] select rm_type from room_type rt, (select id_room_t...

20:23:41
explain (format json) [query] select rm_type from room_type rt, (se...

20:23:07
explain select rm_type from room_type rt, (select id_room_type from...

20:15:51
select * from registration where id_reg = 45;

select * from registration;

rm_type
character varying (20)

1 econom

30.03.2022 20:27:46 1 33 msec
Date Rows Affected Duration

Copy Copy to Query Editor

```
select rm_type from room_type rt, (select id_room_type from
(select id_room_type, count(start_date) from price group by id_room_type
where adt.count = (select max(adadt.count) from
(select id_room_type, count(start_date) from
price group by id_room_type order by id_room_type) adadt)) adt
where rt.id_room_type = adt.id_room_type;
```

Messages
Successfully run. Total query runtime: 33 msec.
1 rows affected.

Рисунок 16 - Просмотр истории 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.

До и после создания индекса CREATE INDEX id_room_type_idx ON room_type (id_room_type);:

1

select id_room_type from room_type;

Data Output

Explain

Messages

Notifications

Graphical

Analysis

Statistics

#	Node	Timings		Rows	Loops
		Exclusive	Inclusive	Actual	
1.	→ Seq Scan on room_type as room_type (actual=0.011..0.012 rows=6 loops=1)	0.012 ms	0.012 ms	6	1

1

select id_room_type from room_type;

Data Output

Explain

Messages

Notifications

Graphical

Analysis

Statistics

#	Node	Timings		Rows	Loops
		Exclusive	Inclusive	Actual	
1.	→ Seq Scan on room_type as room_type (actual=0.013..0.014 rows=6 loops=1)	0.014 ms	0.014 ms	6	

До и после создания составного индекса create index roomer_name_idx on roomer(surname_roomer, f_name);:

1

select surname_roomer, f_name from roomer;

Data Output

Explain

Messages

Notifications

Graphical

Analysis

Statistics

#	Node	Timings		Rows	Loops
		Exclusive	Inclusive	Actual	
1.	→ Seq Scan on roomer as roomer (actual=0.013..0.017 rows=40 loops=1)	0.017 ms	0.017 ms	40	1

1

select surname_roomer, f_name from roomer;

Data Output

Explain

Messages

Notifications

Graphical

Analysis

Statistics

#	Node	Timings		Rows	Loops
		Exclusive	Inclusive	Actual	
1.	→ Seq Scan on roomer as roomer (actual=0.012..0.015 rows=40 loops=1)	0.015 ms	0.015 ms	40	1

Рисунок 17 - Время выполнения запросов без/с индексом

ВЫВОДЫ

В ходе выполнения работы, были освоены практические навыки создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.