

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №3.2 «Создание таблиц базы данных PostgreSQL. Заполнение таблиц рабочими данными»

по дисциплине **«Проектирование и реализация баз данных»**

Автор: Корниенко М.Ю.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

Цель работы.....	3
Практическое задание.....	3
Вариант 12. БД «Прокат автомобилей»	3
Рисунок 1 – Схема логической модели базы данных.	4
Листинг дампа	5
Вывод	4

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.

1. Запросы к БД.

Какой автомобиль находился в прокате максимальное количество часов?

Листинг:

```
WITH TotalHours AS (  
    SELECT  
        r.car_id,  
        m.brand AS model_brand,  
        SUM(EXTRACT(EPOCH FROM (r.end_date - r.start_date))) / 3600 AS total_hours  
    FROM contracts r  
    INNER JOIN cars c ON r.car_id = c.id  
    INNER JOIN models m ON c.model_id = m.id  
    GROUP BY r.car_id, m.brand  
)  
SELECT th.car_id, th.model_brand, th.total_hours  
FROM TotalHours th  
WHERE th.total_hours = (  
    SELECT MAX(total_hours)  
    FROM TotalHours  
);
```

The screenshot shows a database query editor with a 'Query' tab selected. The query is the same as the one in the previous block. Below the query, there is a 'Data Output' tab showing the results of the query. The results are displayed in a table with three columns: car_id, model_brand, and total_hours. The first row shows car_id 1, model_brand Lada, and total_hours 8736.0000000000000000.

car_id	model_brand	total_hours
1	Lada	8736.0000000000000000

Автомобили какой марки чаще всего брались в прокат?

Листинг:

```
WITH RentalCounts AS (  
    SELECT  
        m.brand AS model_brand,  
        COUNT(c.id) AS rental_count  
    FROM contracts ct  
    INNER JOIN cars c ON ct.car_id = c.id  
    INNER JOIN models m ON c.model_id = m.id  
    GROUP BY m.brand  
)  
SELECT rc.model_brand, rc.rental_count
```

```

FROM RentalCounts rc
WHERE rc.rental_count = (
    SELECT MAX(rental_count)
    FROM RentalCounts
);

```

Query










Query History

```
1 WITH RentalCounts AS (
2     SELECT
3         m.brand AS model_brand,
4         COUNT(c.id) AS rental_count
5     FROM contracts ct
6     INNER JOIN cars c ON ct.car_id = c.id
7     INNER JOIN models m ON c.model_id = m.id
8     GROUP BY m.brand
9 )
10 SELECT rc.model_brand, rc.rental_count
11 FROM RentalCounts rc
12 WHERE rc.rental_count = (
13     SELECT MAX(rental_count)
14     FROM RentalCounts
15 );
16
```

Data Output

Messages

Notifications



	model_brand character varying (100) 🔒	rental_count bigint 🔒
1	BMW	1
2	Kia	1
3	Lada	1
4	Mercedes	1
5	Toyota	1

Определить убытки от простоя автомобилей за вчерашний день.
Листинг:

```

WITH DowntimePeriods AS (
    SELECT
        car.id AS car_id,
        models.id AS model_id,
        models.price AS price,
        GREATEST(contracts.end_date, NOW() - INTERVAL '1 day') AS end_date,
        COALESCE(
            LEAD(contracts.start_date) OVER (PARTITION BY car.id ORDER BY
contracts.start_date),
            NOW()
        ) AS next_start_date
    FROM cars car
    LEFT JOIN contracts ON car.id = contracts.car_id
    LEFT JOIN models ON car.model_id = models.id
),
DowntimeCalculations AS (
    SELECT
        dp.car_id,
        dp.model_id,

```

```

        dp.price,
        EXTRACT(EPOCH FROM (dp.next_start_date - dp.end_date)) / 3600 AS downtime_hours
FROM DowntimePeriods dp
WHERE dp.next_start_date > dp.end_date
)
SELECT
    SUM(dp.price * dp.downtime_hours) AS total_downtime_cost
FROM DowntimeCalculations dp;

```

Query Query History

```

1  WITH DowntimePeriods AS (
2      SELECT
3          car.id AS car_id,
4          models.id AS model_id,
5          models.price AS price,
6          GREATEST(contracts.end_date, NOW() - INTERVAL '1 day') AS end_date,
7          COALESCE(
8              LEAD(contracts.start_date) OVER (PARTITION BY car.id ORDER BY contracts.start_date),
9              NOW()
10         ) AS next_start_date
11     FROM cars car
12     LEFT JOIN contracts ON car.id = contracts.car_id
13     LEFT JOIN models ON car.model_id = models.id
14 ),
15 DowntimeCalculations AS (
16     SELECT
17         dp.car_id,
18         dp.model_id,
19         dp.price,
20         EXTRACT(EPOCH FROM (dp.next_start_date - dp.end_date)) / 3600 AS downtime_hours
21     FROM DowntimePeriods dp
22     WHERE dp.next_start_date > dp.end_date
23 )
24 SELECT
25     SUM(dp.price * dp.downtime_hours) AS total_downtime_cost
26 FROM DowntimeCalculations dp;

```

Data Output Messages Notifications

total_downtime_cost numeric

1	[null]
---	--------

Вывести данные автомобиля, имеющего максимальный пробег.

Листинг:

```

select * from cars
where mileage = (
    select max(mileage)
    from cars
);

```

Query Query History

```

1 select * from cars
2 where mileage = (
3     select max(mileage)
4     from cars
5 );
6

```

Data Output Messages Notifications

	id [PK] integer	body_number character varying (17)	register_number character varying (8)	engine_number character varying (17)	last_to_date date	return_remarks character varying (100)	model_id integer	mileage integer
1	5	ТЕЛО5234567890123	РЕГ5234	ДВИГ5234567890123	2023-10-05	[null]	5	800000

Какой автомобиль суммарно находился в прокате дольше всех.

Листинг:

```

select cars.*
from contracts
join cars on cars.id = contracts.car_id
where (end_date - start_date) = (
    select max(end_date - start_date)
    from contracts
);

```

Query Query History

```

1 select cars.*
2 from contracts
3 join cars on cars.id = contracts.car_id
4 where (end_date - start_date) = (
5     select max(end_date - start_date)
6     from contracts
7 );
8

```

Data Output Messages Notifications

	id [PK] integer	body_number character varying (17)	register_number character varying (8)	engine_number character varying (17)	last_to_date date	return_remarks character varying (100)	model_id integer	mileage integer
1	1	ТЕЛО1234567890123	РЕГ1234	ДВИГ1234567890123	2023-10-01	[null]	1	30000

Query
Query History

1
select avg(date_part('year', CURRENT_DATE) - year_of_car) FROM cars;

Data Output
Messages
Notifications

+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	avg double precision
1	9.4

2. Создание представлений:

Какой автомобиль ни разу не был в прокате?

Листинг:

```
CREATE VIEW CarsNotInRent AS
SELECT c.*
FROM cars c
WHERE NOT EXISTS (
    SELECT 1
    FROM contracts contr
    WHERE contr.car_id = c.id
);
```

Query Query History

```

1 CREATE VIEW CarsNotInRent AS
2 SELECT c.*
3 FROM cars c
4 WHERE NOT EXISTS (
5     SELECT 1
6     FROM contracts contr
7     WHERE contr.car_id = c.id
8 );
9
10 select * from CarsNotInRent;

```

Data Output Messages Notifications

id	body_number	register_number	engine_number	last_to_date	return_remarks	model_id	mileage	year_of_car
integer	character varying (17)	character varying (8)	character varying (17)	date	character varying (100)	integer	integer	integer

Вывести данные клиентов, не вернувших автомобиль вовремя.

Листинг:

```

CREATE VIEW ExpiredClients AS
SELECT
    cl.id AS client_id,
    cl.mobile_phone AS phone,
    ct.end_date AS enddate
FROM
    clients cl
INNER JOIN
    (SELECT * FROM contracts WHERE end_date < NOW() AND state_status = 'Активный') AS
ct
ON
    cl.id = ct.client_id;

```

QueryQuery History

```

1 CREATE VIEW ExpiredClients AS
2 SELECT
3     cl.id AS client_id,
4     cl.mobile_phone AS phone,
5     ct.end_date AS enddate
6 FROM
7     clients cl
8 INNER JOIN
9     (SELECT * FROM contracts WHERE end_date < NOW() AND state_status = 'Активный') AS ct
10 ON
11     cl.id = ct.client_id;
12
13 select * from ExpiredClients;

```

Data OutputMessagesNotifications

client_id	phone	enddate
integer	character varying	timestamp with time zone

3. Запросы на модификацию данных:

Добавление работника на должность с наибольшим количеством работников

Листинг:

```

INSERT INTO workers (id, name, passport_data, position_id)
SELECT 6, 'Виталий Витальев', 12345678, (
    SELECT p.id
    FROM positions p
    JOIN workers w ON p.id = w.position_id
    GROUP BY p.id
    ORDER BY COUNT(w.id) DESC
    LIMIT 1
);

```

Query

Query History

```

1  INSERT INTO workers (id, name, passport_data, position_id)
2  SELECT 6, 'Виталий Витальев', 12345678, (
3      SELECT p.id
4      FROM positions p
5      JOIN workers w ON p.id = w.position_id
6      GROUP BY p.id
7      ORDER BY COUNT(w.id) DESC
8      LIMIT 1
9  );
10
11 select * from workers;

```

Data Output

Messages

Notifications

≡

📄

▼

📋

▼

🗑️

🗑️

📥

📥

📈

	id [PK] integer	name character varying (50)	passport_data character varying (12)	position_id integer
1	1	Мария Сергеева	123456789012	1
2	2	Петр Алексеев	234567890123	2
3	3	Светлана Васильева	345678901234	3
4	4	Андрей Петров	456789012345	4
5	5	Екатерина Иванова	567890123456	5
6	6	Виталий Витальев	12345678	5

Повышение зарплаты для должности, на которой суммарно получают меньше всего

Листинг:

```

UPDATE positions
SET salary = salary * 1.05
WHERE id = (

```

```

SELECT p.id
FROM positions p
LEFT JOIN workers w ON p.id = w.position_id
GROUP BY p.id
ORDER BY SUM(p.salary) ASC
LIMIT 1
);

```

Query
Query History

```

1 UPDATE positions
2 SET salary = salary * 1.05
3 WHERE id = (
4     SELECT p.id
5     FROM positions p
6     LEFT JOIN workers w ON p.id = w.position_id
7     GROUP BY p.id
8     ORDER BY SUM(p.salary) ASC
9     LIMIT 1
10 );
11
12 select * from positions;

```

Data Output
Messages
Notifications

	id [PK] integer	salary integer	tasks character varying (100)
1	1	50000	Продавец
2	2	50000	Менеджер по продажам
3	3	50000	Технический специалист
4	5	50000	Руководитель отдела
5	4	52500	Менеджер по логистике

Удаление неактивных контрактов с конца которых прошло более 1 года

```
DELETE FROM contracts
WHERE client_id IN (
    SELECT cl.id
    FROM clients cl
    LEFT JOIN contracts ct ON cl.id = ct.client_id
    GROUP BY cl.id
    HAVING MAX(ct.end_date) < NOW() - INTERVAL '1 year' AND MAX(ct.state_status) !=
'Активный'
);
```

Query Query History

```
1 DELETE FROM contracts
2 WHERE client_id IN (
3     SELECT cl.id
4     FROM clients cl
5     LEFT JOIN contracts ct ON cl.id = ct.client_id
6     GROUP BY cl.id
7     HAVING MAX(ct.end_date) < NOW() - INTERVAL '1 year' AND MAX(ct.state_status) != 'Активный'
8 );
9
10 select * from contracts;
```

Data Output Messages Notifications

	id [PK] integer	start_date timestamp with time zone	end_date timestamp with time zone	state_status character varying (50)	paid_status character varying (50)	car_id integer	client_id integer	worker_id integer	total_cost integer
1	1	2023-01-01 00:00:00+03	2023-12-31 00:00:00+03	Активный	Оплачено		1	1	[null]
2	2	2023-01-02 00:00:00+03	2023-12-30 00:00:00+03	Завершён	Не оплачено	2	2	2	[null]
3	3	2023-01-03 00:00:00+03	2023-12-29 00:00:00+03	Активный	Оплачено	3	3	3	[null]
4	4	2023-01-04 00:00:00+03	2023-12-28 00:00:00+03	Завершён	Оплачено	4	4	4	[null]
5	5	2023-01-05 00:00:00+03	2023-12-27 00:00:00+03	Активный	Не оплачено	5	5	5	[null]

Создание индексов

Листинг:

```
CREATE INDEX idx_cars_model_id ON cars(model_id);
CREATE INDEX idx_contracts_id_end_date ON contracts(id, end_date);
CREATE INDEX idx_workers_name ON workers(name);
CREATE INDEX idx_workers_name_position_id ON workers(name, position_id);
```

Сравнение времени работы с индексами и без:

1. С индексами

Query

Query History

```

2      SELECT
3          car.id AS car_id,
4          models.id AS model_id,
5          models.price AS price,
6          GREATEST(contracts.end_date, NOW() - INTERVAL '1 day') AS end_date,
7          COALESCE(
8              LEAD(contracts.start_date) OVER (PARTITION BY car.id ORDER BY contracts.start_date),
9              NOW()
10         ) AS next_start_date
11     FROM cars car
12     LEFT JOIN contracts ON car.id = contracts.car_id
13     LEFT JOIN models ON car.model_id = models.id
14 ),
15 DowntimeCalculations AS (
16     SELECT
17         dp.car_id,
18         dp.model_id,
19         dp.price,
20         EXTRACT(EPOCH FROM (dp.next_start_date - dp.end_date)) / 3600 AS downtime_hours
21     FROM DowntimePeriods dp
22     WHERE dp.next_start_date > dp.end_date
23 )
24 SELECT
25     SUM(dp.price * dp.downtime_hours) AS total_downtime_cost
26 FROM DowntimeCalculations dp;
27

```

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

📦

⬇️

📈

total_downtime_cost

numeric

🔒

1	[null]
---	--------

Total rows: 1 of 1

Query complete 00:00:00.051

2. Без индексов

Query

Query History

```

2      SELECT
3          car.id AS car_id,
4          models.id AS model_id,
5          models.price AS price,
6          GREATEST(contracts.end_date, NOW() - INTERVAL '1 day') AS end_date,
7          COALESCE(
8              LEAD(contracts.start_date) OVER (PARTITION BY car.id ORDER BY contracts.start_date)
9              NOW()
10         ) AS next_start_date
11     FROM cars car
12     LEFT JOIN contracts ON car.id = contracts.car_id
13     LEFT JOIN models ON car.model_id = models.id
14 ),
15 DowntimeCalculations AS (
16     SELECT
17         dp.car_id,
18         dp.model_id,
19         dp.price,
20         EXTRACT(EPOCH FROM (dp.next_start_date - dp.end_date)) / 3600 AS downtime_hours
21     FROM DowntimePeriods dp
22     WHERE dp.next_start_date > dp.end_date
23 )
24 SELECT
25     SUM(dp.price * dp.downtime_hours) AS total_downtime_cost
26 FROM DowntimeCalculations dp;
27

```

Data Output

Messages

Notifications

≡

📄

▼

📋

▼

🗑️

🔄

⬇️

📈

	total_downtime_cost	
	numeric	🔒
1	[null]	

Total rows: 1 of 1

Query complete 00:00:00.110

Вывод

Во время лабораторной работы я научился использовать разнообразные SQL-запросы для взаимодействия с базой данных, а также узнал, как создавать представления и индексы. Кроме того, я провёл сравнение времени выполнения SELECT-запросов с использованием индексов и без них. Как оказалось, наличие индексов сокращает время выполнения запросов.