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

Отчет

по лабораторной работе №4 «Запросы на выборку и модификацию данных. Представления. Работа с индексами»

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

Автор: Даньшин С. А.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

Оглавление

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 базе данных

- Вывести фамилии водителей и номера автобусов, отправившиеся в рейсы до 12 часов текущего дня.

```
SELECT drivers.last_name, buses.registration_number
FROM trips
JOIN crews ON trips.id = crews.trip_id
JOIN drivers ON crews.driver_id = drivers.id
JOIN buses ON trips.bus_id = buses.id
WHERE cast(trips.start_time as time) < '12:00:00' and cast(trips.start_time as date) = current_date;
```

- Рассчитать выручку от продажи билетов за прошедший день.

```
SELECT SUM(tr.price) AS total_revenue
FROM tickets t
JOIN trips tr ON t.trip_id = tr.id
where cast(t.sold_at as date) = current_date - interval 'day 1'
and status=”paid”;
```

- Вывести список водителей, которые не выполнили ни одного рейса за прошедший день.

```
select * from drivers
where drivers.id not in (
select d.id from trips t
join crews c on c.trip_id = t.id
join drivers d on c.driver_id = d.id
where cast(t.start_time as date) = current_date - interval '1 day';
```

- Вывести сумму убытков из-за непроданных мест в автобусе за прошедшую неделю.

```
select
    tr.id,
  (tr.number_of_seats - COUNT(t.id)) * tr.price as lost_profit
from tickets t
join (
    select
    trips.id as id,
    trips.start_time,
    bm.number_of_seats as number_of_seats,
    trips.price
    from trips
    join buses b on trips.bus_id = b.id
    join bus_models bm on bm.id = b.model_id
) tr on t.trip_id = tr.id
where cast(tr.start_time as date) > current_date - interval '7 day'
group by tr.id, tr.number_of_seats, tr.price;
```

- Сколько рейсов выполнил каждый водитель за последний месяц.

```
    select c.driver_id as driver_id, count(tr.id) from trips tr
    join crews c on tr.id = c.trip_id
    where cast(tr.start_time as date) > current_date - interval '1 month'
    and tr.status=’completed’
    group by c.driver_id
```

- Вывести тип автобуса, который используется на всех рейсах.

```
select
	bus_model_id
from (
	select
		bm.id as bus_model_id
		count(r.id) as rout_count,
	from routes r
	join trips t on r.id=t.route_id
	JOIN buses b ON t.bus_id=b.id
	JOIN bus_models bm ON bm.id=b.model_id
	group bm.id
) as br
where rout_count=(select count(id) from routes r2);
```

- Вывести данные водителя, который провел максимальное время в пути за прошедшую неделю.

```
SELECT driver_id
FROM (
	SELECT ddt.driver_id, sum(ddt.driving_time) as total_driving_time
	FROM (
		SELECT d.id as driver_id, tr.id as trip_id, tr.end_time - tr.start_time as driving_time
		FROM trips tr
		JOIN crews c ON tr.id = c.trip_id
		JOIN drivers d ON c.driver_id = d.id
		WHERE cast(tr.start_time as date) > current_date - interval '7 day'
	) as ddt
	GROUP BY ddt.driver_id
) mt
WHERE total_driving_time = (
	SELECT MAX(total_driving_time)
	FROM (
		SELECT ddt.driver_id, sum(ddt.driving_time) as total_driving_time
		FROM (
			SELECT d.id as driver_id, tr.id as trip_id, tr.end_time - tr.start_time as
driving_time
			FROM trips tr
			JOIN crews c ON tr.id = c.trip_id
			JOIN drivers d ON c.driver_id = d.id
			WHERE cast(tr.start_time as date) > current_date - interval '7 day'
		) as ddt
		GROUP BY ddt.driver_id
	) sub
);
```

2. Представления

- Количество свободных мест на все рейсы на завтра:

```
CREATE VIEW available_seats_tomorrow AS
        SELECT
                t.id AS trip_id,
                r.name AS route_name,
                bm.brand || " ||  bm.model_name AS bus_model,
                b.registration_number AS bus_number,
    COUNT(tickets.id) AS occupied_seats, bm.number_of_seats AS total_seats,
    bm.number_of_seats - COUNT(tickets.id) AS available_seats
        FROM trips t
        JOIN buses b ON t.bus_id = b.id
        JOIN bus_models bm ON b.model_id = bm.id
        JOIN routes r ON t.route_id = r.id
        LEFT JOIN tickets ON t.id = tickets.trip_id
        WHERE t.start_time >= CURRENT_DATE + INTERVAL '1 day'
                AND t.start_time < CURRENT_DATE + INTERVAL '2 days'
        GROUP BY t.id, r.name, bm.brand, bm.model_name, b.registration_number,
bm.number_of_seats;
```

- Самый популярный маршрут этой зимой:

```
CREATE VIEW most_popular_routes_winter AS
SELECT route_id, route_name, total_tickets_sold
FROM (
   SELECT r.id AS route_id, r.name AS route_name,
       COUNT(tickets.id) AS total_tickets_sold
   FROM routes r
   JOIN trips t ON r.id = t.route_id
   JOIN tickets ON t.id = tickets.trip_id
   WHERE EXTRACT(MONTH FROM t.start_time) IN (12, 1, 2)
   GROUP BY r.id, r.name
) as route_sales
WHERE total_tickets_sold = (
   SELECT MAX(total_tickets_sold)
   FROM (
      SELECT COUNT(tickets.id) AS total_tickets_sold
      FROM routes r
      JOIN trips t ON r.id = t.route_id
      JOIN tickets ON t.id = tickets.trip_id
      WHERE EXTRACT(MONTH FROM t.start_time) IN (12, 1, 2)
      GROUP BY r.id, r.name
   ) as max_sales
);
```

3. Запросы на модификацию (с подзапросами)

- Исправить опечатку в названии модели автобуса

```
UPDATE bus_models
SET model_name = 'Lions Coach'
WHERE id = (SELECT id FROM bus_models WHERE brand = ‘MAN’ AND model_name = ‘Lionscroach’’);
```

- Удалить пассажиров которые не ездили у нас больше 2х лет

```
DELETE FROM passengers
WHERE id IN (
   SELECT p.id
   FROM passengers p
   JOIN tickets t ON p.id = t.passenger_id
   JOIN trips tr ON t.trip_id = tr.id
   WHERE tr.end_time < NOW() - INTERVAL '2 years'
   GROUP BY p.id
   HAVING MAX(tr.end_time) < NOW() - INTERVAL '2 years'
);
```

- Добавление билета:

```
INSERT INTO tickets (passenger_id, trip_id, seat_number, status, start_station_id, end_station_id, is_online_sale)
VALUES (
   (SELECT id FROM passengers WHERE email = 'a.a@a.ru'),
   (SELECT id FROM trips WHERE route_id = 1 AND status=”PLANNED” ORDER BY start_time),
   1,
   'RESERVED',
   1,
   2,
   true
);
```

4. Индексы

Создадим индекс на время отправления завершения поездки как отдельно, так и вместе:

```
CREATE INDEX idx_trip_start_time on trips(start_time);

CREATE INDEX idx_trip_end_time on trips(end_time);

CREATE INDEX idx_trip_start_and_end_time on trips(start_date, end_time);
```

Тестовый запрос:

```
select
	t.id as trip_id,
	r.name as route_name,
	bm.brand || ' ' || bm.model_name as bus_model,
	b.registration_number as bus_number,
	bm.number_of_seats - COUNT(t2.id) as available_seats
from
	trips t
join buses b on
	t.bus_id = b.id
join bus_models bm on
	b.model_id = bm.id
join routes r on
	t.route_id = r.id
left join tickets t2 on
	t.id = t2.trip_id
where
	cast(t.start_time as date) = '2021-01-01'
	and cast(t.end_time as date) = '2021-01-01'
group by
	t.id,
	r.name,
	bm.brand,
	bm.model_name,
	b.registration_number,
	bm.number_of_seats;
```

Без индекса:

Planning Time: 4.879 ms
Execution Time: 128.666 ms

Индекс стартового времени:

Planning Time: 3.667 ms
Execution Time: 56.838 ms

Индекс конечного времени:

Planning Time: 1.479 ms
Execution Time: 42.334 ms

Индекс на оба поля:

Planning Time: 0.729 ms
Execution Time: 44.153 ms

Вывод

В ходе лабораторной работы я освоил практические навыки по выполнению select, insert, delete и update запросов. Также, научился делать представления и индексы. В ходе анализа в первом запросе индексы позволили сократить время выполнения запроса на ~40%.