

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: Хурс П.И

Факультет: ИКТ

Группа: K3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

1. Запросы к базе данных	3
2. Представления.....	6
3. Кастом запросы	9
4. Индексы	10
Вывод.....	11

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

- Свободные места на все поезда, отправляющиеся с вокзала в течение следующих суток.

```
SELECT
train.train_id,
  schedule.departure_time,
  carriage.carriage_number,
  seat.seat_number
FROM
  train
JOIN schedule ON train.schedule_id = schedule.schedule_id
JOIN carriage ON train.train_id = carriage.train_id
JOIN seat ON carriage.carriage_id = seat.carriage_id
LEFT JOIN ticket ON seat.seat_id = ticket.seat_id
WHERE
  schedule.station_name_from = 'StationA' AND
  schedule.departure_time BETWEEN CURRENT_DATE + INTERVAL '1 day' AND
CURRENT_DATE + INTERVAL '2 days' AND
  seat.locked_status = false
ORDER BY
  train.train_id, carriage.carriage_number, seat.seat_number;
```

	train_id integer	departure_time date	carriage_number integer	seat_number character varying (6)
1	4	2023-12-08	1	1A
2	4	2023-12-08	2	2A
3	5	2023-12-08	1	1A

- Список пассажиров, отправившихся в Москву всеми рейсами за прошедшие сутки.

```
SELECT
```

```

p.passenger_id,
  p.first_name,
  p.last_name,
  s.departure_time,
  s.station_name_to
FROM
  passenger p
JOIN ticket t ON p.passenger_id = t.passenger_id
JOIN seat se ON t.seat_id = se.seat_id
JOIN carriage c ON se.carriage_id = c.carriage_id
JOIN train tr ON c.train_id = tr.train_id
JOIN schedule s ON tr.schedule_id = s.schedule_id
WHERE
  t.station_name_to = 'Moscow' AND
  s.departure_time BETWEEN CURRENT_DATE - INTERVAL '1 day' AND
CURRENT_DATE
ORDER BY
  s.departure_time DESC;

```

	passenger_id integer	first_name character varying (100)	last_name character varying (100)	departure_time date	station_name_to character varying (100)
1	3	Ivan	Ivanov	2023-12-06	Moscow
2	4	Maria	Petrova	2023-12-06	Moscow

- Номера поездов, на которые проданы все билеты на следующие сутки.

```

SELECT
  t.train_id
FROM
  train t
JOIN schedule sch ON t.schedule_id = sch.schedule_id
JOIN carriage c ON t.train_id = c.train_id
JOIN seat s ON c.carriage_id = s.carriage_id
LEFT JOIN ticket ti ON s.seat_id = ti.seat_id
WHERE
  sch.departure_time BETWEEN CURRENT_DATE + INTERVAL '1 day' AND
CURRENT_DATE + INTERVAL '2 days'
GROUP BY
  t.train_id
HAVING
  COUNT(s.seat_id) = COUNT(ti.ticket_id);

```

	train_id [PK] integer
1	8
2	9

- Свободные места в купейные вагоны всех рейсов до Москвы на текущие сутки.

```

SELECT
  tr.train_id,

```

```

    sch.departure_time,
    c.carriage_number,
    s.seat_number
FROM
    train tr
JOIN schedule sch ON tr.schedule_id = sch.schedule_id
JOIN carriage c ON tr.train_id = c.train_id
JOIN seat s ON c.carriage_id = s.carriage_id
LEFT JOIN ticket ti ON s.seat_id = ti.seat_id
WHERE
    sch.station_name_to = 'Москва' AND
    sch.departure_time BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '1
day' AND
    c.carriage_type = 'купе' AND
    s.locked_status = false AND
    ti.ticket_id IS NULL
ORDER BY
    tr.train_id, c.carriage_number, s.seat_number;

```

	train_id integer	departure_time date	carriage_number integer	seat_number character varying (6)
1	10	2023-12-07	1	2A
2	10	2023-12-07	1	2B
3	11	2023-12-07	1	1B

- Выручка от продажи билетов на все поезда за прошедшие сутки.

```

SELECT
SUM(t.price) AS total_revenue
FROM
    ticket t
JOIN seat s ON t.seat_id = s.seat_id
JOIN carriage c ON s.carriage_id = c.carriage_id
JOIN train tr ON c.train_id = tr.train_id
JOIN schedule sch ON tr.schedule_id = sch.schedule_id
WHERE
    sch.departure_time BETWEEN CURRENT_DATE - INTERVAL '1 day' AND
CURRENT_DATE;

```

	total_revenue bigint
1	450

- Общее количество билетов, проданных по всем направлениям в вагоны типа “СВ”.

```


SELECT
COUNT(t.ticket_id) AS total_tickets

```

```

FROM
    ticket t
JOIN seat s ON t.seat_id = s.seat_id
JOIN carriage c ON s.carriage_id = c.carriage_id
WHERE
    c.carriage_type = 'CB';

```



	total_tickets bigint 
1	0

- Номера и названия поездов, все вагоны которых были заполнены менее чем наполовину за прошедшие сутки.

```

SELECT
    tr.train_id,
    sch.train_name
FROM
    train tr
JOIN schedule sch ON tr.schedule_id = sch.schedule_id
JOIN carriage c ON tr.train_id = c.train_id
LEFT JOIN seat s ON c.carriage_id = s.carriage_id
LEFT JOIN ticket t ON s.seat_id = t.seat_id
WHERE
    sch.departure_time BETWEEN CURRENT_DATE - INTERVAL '1 day' AND
CURRENT_DATE
GROUP BY
    c.carriage_id, tr.train_id, sch.train_name
HAVING
    AVG(CASE WHEN t.ticket_id IS NOT NULL THEN 1 ELSE 0 END) < 0.5

```

	train_id integer 	train_name character varying (100) 
1	10	Capital Rocket

2. Представления

- для пассажиров о наличии свободных мест на заданный рейс

```
CREATE VIEW view_available_seats AS
SELECT
    s.seat_id,
    tr.train_id,
    c.carriage_number,
    s.seat_number
FROM
    train tr
JOIN carriage c ON tr.train_id = c.train_id
JOIN seat s ON c.carriage_id = s.carriage_id
LEFT JOIN ticket t ON s.seat_id = t.seat_id
WHERE
    s.locked_status = false AND
    t.ticket_id IS NULL;
```

	seat_id integer	train_id integer	carriage_number integer	seat_number character varying (6)
1	6	4	1	1A
2	8	4	2	2A
3	9	5	1	1A
4	16	10	1	2A
5	17	10	1	2B
6	19	11	1	1B

- количество непроданных билетов на все поезда, формирующиеся за прошедшие сутки (номер поезда, тип вагона, количество).

```
CREATE VIEW view_unsold_tickets AS
SELECT
    tr.train_id,
    c.carriage_type,
    COUNT(*) - COUNT(t.ticket_id) AS unsold_tickets
FROM
    train tr
JOIN schedule sch ON tr.schedule_id = sch.schedule_id
JOIN carriage c ON tr.train_id = c.train_id
JOIN seat s ON c.carriage_id = s.carriage_id
LEFT JOIN ticket t ON s.seat_id = t.seat_id
WHERE
    sch.departure_time BETWEEN CURRENT_DATE - INTERVAL '1 day' AND
CURRENT_DATE
GROUP BY
    tr.train_id,
    c.carriage_type;
```

	train_id integer	carriage_type character varying (50)	unsold_tickets bigint
1	6	Business	0
2	7	Economy	0
3	10	купе	2
4	11	купе	1

3. Кастом запросы

- Измените статус всех поездов, отправленные которых запланировано на следующую неделю, на "Сезонная корректировка", если по-прежнему доступно более 50% мест.

```
UPDATE train
SET train_status = 'Under Maintenance'
WHERE train_id IN (
    SELECT tr.train_id
    FROM train tr
    JOIN schedule sch ON tr.schedule_id = sch.schedule_id
    WHERE tr.train_status = 'Delayed'
    AND sch.departure_time BETWEEN CURRENT_DATE - INTERVAL '1 month' AND
CURRENT_DATE
    GROUP BY tr.train_id
    HAVING COUNT(tr.train_id) > 5
);
```

```
UPDATE 3
```

```
Query returned successfully in 100 msec.
```

- Удалите все будущие бронирования (билеты) на поезда, которые были помечены как "Находящиеся на обслуживании" в связи с сезонными изменениями.

```
DELETE FROM ticket
WHERE seat_id IN (
    SELECT s.seat_id
    FROM seat s
    JOIN carriage c ON s.carriage_id = c.carriage_id
    JOIN train tr ON c.train_id = tr.train_id
    WHERE tr.train_status = 'Under Maintenance'
    AND tr.train_id IN (
        SELECT c.train_id
        FROM schedule sch
        WHERE sch.departure_time > CURRENT_DATE
    )
);
```

DELETE 0

Query returned successfully in 37 msec.

- Добавление специального поезда для мероприятия:

```
INSERT INTO public.train (train_id, shedule_id, departure_date, arrival_date,
train_status)
SELECT
    (SELECT MAX(train_id) + 1 FROM public.train),
    schedule_id,
    CURRENT_DATE,
    CURRENT_DATE + INTERVAL '2 days',
    'Special Event'
FROM public.schedule
WHERE station_name_from = 'Event City Start' AND station_name_to = 'Event City
End';
```

INSERT 0 1

Query returned successfully in 83 msec.

4. Индексы

Создадим индекс на даты в контракте и на айди машины:

```
CREATE INDEX idx_passport_date_of_birth on
passenger(passport_date_of_birth);
```

```
SELECT *
FROM passenger
WHERE EXTRACT(YEAR FROM age(passport_date_of_birth)) BETWEEN 70 AND 80;
```

Без индекса:

```
Successfully run. Total query runtime: 470 msec.  
157092 rows affected.
```

С индексом:

```
Successfully run. Total query runtime: 307 msec.  
157092 rows affected.
```

```
CREATE INDEX idx_passenger_passport_number  
ON public.passenger (passport_number);
```

```
SELECT passport_number  
FROM passenger  
WHERE passport_number ~ '^[0-9]';
```

Без индекса:

```
Successfully run. Total query runtime: 324 msec.  
624885 rows affected.
```

С индексом:

```
Successfully run. Total query runtime: 229 msec.  
624885 rows affected.
```

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 дали буст в ~40%, во втором запросе практически не дали.