

**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»

Обучающийся Федоров Даниил Михайлович

Факультет прикладной информатики

Группа К3240

Направление подготовки 09.03.03 Прикладная информатика

Образовательная программа Мобильные и сетевые технологии 2023

Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2024/2025

СОДЕРЖАНИЕ

Стр.

1 Цель работы.....	3
2 Практическое задание.....	4
3 Схема базы данных (ЛР 3).	5
4 Выполнение.....	6
4.1 Запросы к базе данных.....	6
Выводы	15

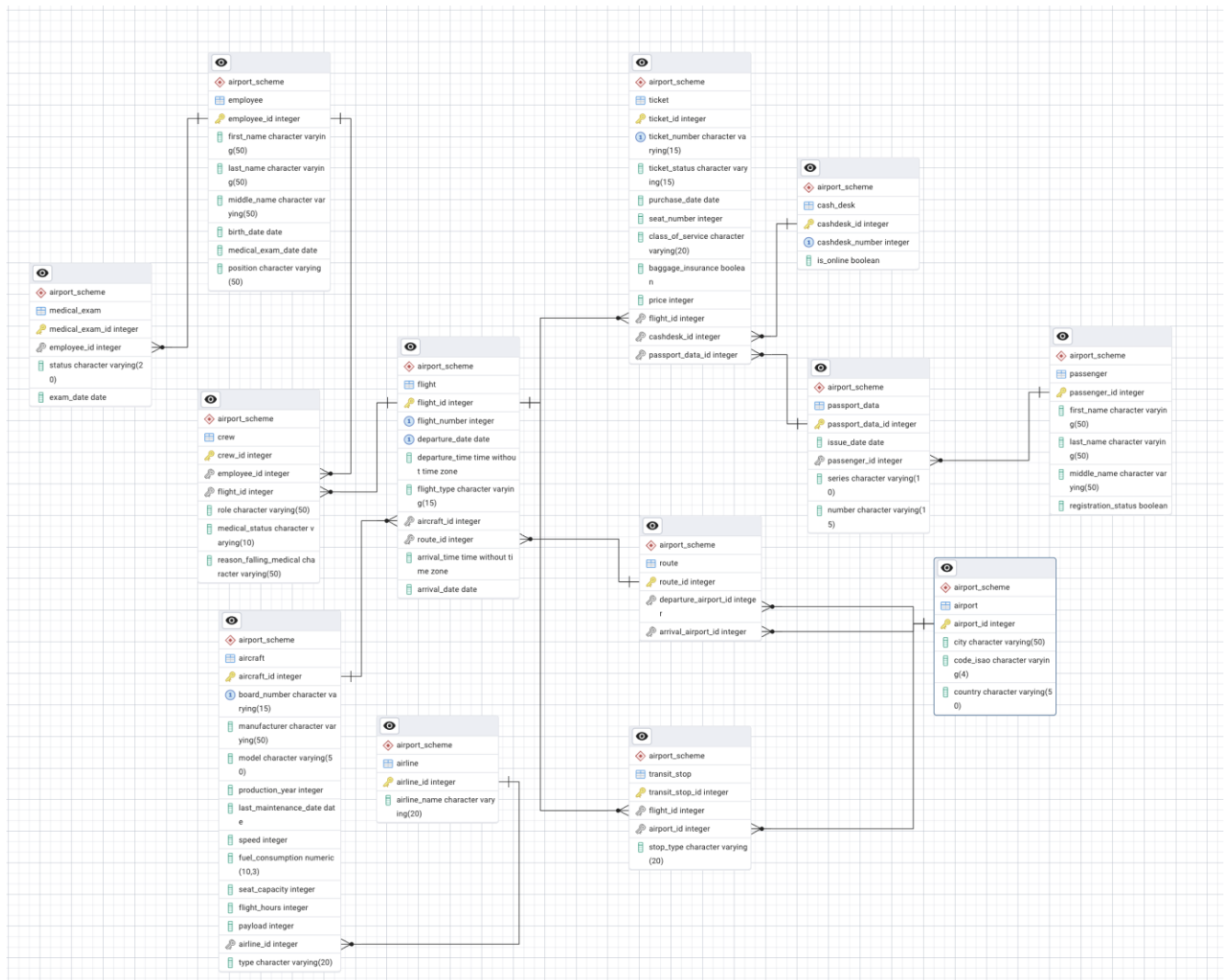
1 Цель работы

Цель работы: овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

2 Практическое задание

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию лабораторной работы №2, 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.

3 Схема базы данных (ЛР 3).



4 Выполнение

4.1 Запросы к базе данных

- Определить расчетное время полета по всем маршрутам.
SELECT
- route_id,
- EXTRACT(EPOCH FROM (
• (CAST(arrival_date AS timestamp) + arrival_time) -
• (CAST(departure_date AS timestamp) + departure_time)
•)) / 3600 AS flight_duration_hours,
• (EXTRACT(EPOCH FROM (
• (CAST(arrival_date AS timestamp) + arrival_time) - (CAST(departure_date
AS timestamp) + departure_time)
•)) % 3600) / 60 AS flight_duration_minutes
- FROM
- airport_scheme.flight
- ORDER BY
- route_id, flight_number;

	route_id integer	flight_duration_hours numeric	flight_duration_minutes numeric
1	1	1.9166666666666667	55.0000000000000000
2	2	2.0000000000000000	0.0000000000000000
3	3	2.0000000000000000	0.0000000000000000
4	4	2.0000000000000000	0.0000000000000000
5	5	17.0000000000000000	0.0000000000000000
6	6	2.0000000000000000	0.0000000000000000
7	7	2.0000000000000000	0.0000000000000000
8	8	2.0000000000000000	0.0000000000000000
9	9	2.0000000000000000	0.0000000000000000
10	10	2.0000000000000000	0.0000000000000000
11	11	2.0000000000000000	0.0000000000000000
12	12	2.0000000000000000	0.0000000000000000
13	13	2.0000000000000000	0.0000000000000000
14	14	2.0000000000000000	0.0000000000000000
15	15	2.0000000000000000	0.0000000000000000
16	16	2.0000000000000000	0.0000000000000000
17	17	2.0000000000000000	0.0000000000000000
18	18	2.0000000000000000	0.0000000000000000
19	19	2.0000000000000000	0.0000000000000000
20	20	2.0000000000000000	0.0000000000000000
21	21	2.0000000000000000	0.0000000000000000
22	22	2.0000000000000000	0.0000000000000000
23	23	2.0000000000000000	0.0000000000000000

- Определить расход топлива по всем маршрутам.

```

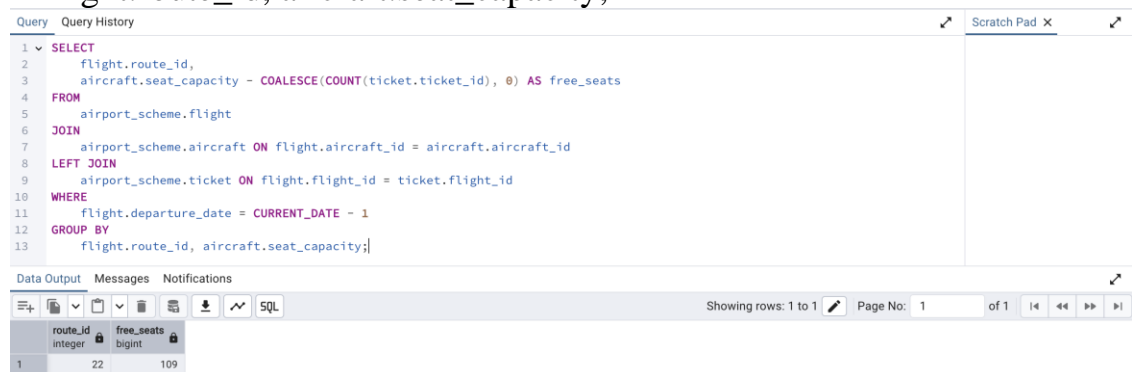
SELECT
    route.route_id,
    SUM(aircraft.fuel_consumption * (EXTRACT(EPOCH FROM
(CAST(flight.arrival_date AS timestamp) + flight.arrival_time) -
CAST(flight.departure_date AS timestamp) - flight.departure_time) / 3600)) AS
fuel_consumed_tonnes
FROM
    airport_scheme.flight
JOIN
    airport_scheme.route ON flight.route_id = route.route_id
JOIN
    airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id
GROUP BY
    route.route_id

```

	route_id [PK] integer	fuel_consumed_tonnes numeric
1	29	10.600000000000000000
2	4	14.000000000000000000
3	34	9.200000000000000000
4	40	14.600000000000000000
5	32	10.200000000000000000
6	10	9.200000000000000000
7	9	10.000000000000000000
8	7	10.600000000000000000
9	35	10.000000000000000000
10	38	8.800000000000000000
11	15	10.200000000000000000
12	6	8.400000000000000000
13	26	9.400000000000000000
14	12	14.200000000000000000
15	39	13.600000000000000000
16	24	9.200000000000000000
17	19	13.200000000000000000
18	36	9.600000000000000000
19	25	10.000000000000000000
20	31	10.800000000000000000
21	30	9.800000000000000000
22	21	10.200000000000000000

- Вывести данные о том, сколько свободных мест оставалось в самолетах, совершавших полет по заданному рейсу за вчерашний день.

```
SELECT
    flight.route_id,
    aircraft.seat_capacity - COALESCE(COUNT(ticket.ticket_id), 0) AS
free_seats
FROM
    airport_scheme.flight
JOIN
    airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id
LEFT JOIN
    airport_scheme.ticket ON flight.flight_id = ticket.flight_id
WHERE
    flight.departure_date = CURRENT_DATE - 1
GROUP BY
    flight.route_id, aircraft.seat_capacity;
```



The screenshot shows a SQL query editor with the following query:

```
1 SELECT
2     flight.route_id,
3     aircraft.seat_capacity - COALESCE(COUNT(ticket.ticket_id), 0) AS free_seats
4 FROM
5     airport_scheme.flight
6 JOIN
7     airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id
8 LEFT JOIN
9     airport_scheme.ticket ON flight.flight_id = ticket.flight_id
10 WHERE
11     flight.departure_date = CURRENT_DATE - 1
12 GROUP BY
13     flight.route_id, aircraft.seat_capacity;
```

The results are displayed in a table with the following columns: route_id (integer), free_seats (bigint). The table shows one row with route_id 22 and free_seats 109.

route_id integer	free_seats bigint
22	109

- Рассчитать убытки компании за счет непроданных билетов за вчерашний день.

```
SELECT
    flight.route_id,
    ticket.price * (aircraft.seat_capacity - COALESCE(COUNT(ticket.ticket_id),
0)) AS fail_sum
FROM
    airport_scheme.flight flight
JOIN
    airport_scheme.aircraft aircraft ON flight.aircraft_id = aircraft.aircraft_id
LEFT JOIN
    airport_scheme.ticket ticket ON flight.flight_id = ticket.flight_id
WHERE
    flight.departure_date = CURRENT_DATE - 1
GROUP BY
```


flight.route_id, ticket.price, aircraft.seat_capacity;

Query Query History		Scratch Pad X
<pre> 1 SELECT 2 flight.route_id, 3 ticket.price * (aircraft.seat_capacity - COALESCE(COUNT(ticket.ticket_id), 0)) AS fail_sum 4 FROM 5 airport_scheme.flight flight 6 JOIN 7 airport_scheme.aircraft aircraft ON flight.aircraft_id = aircraft.aircraft_id 8 LEFT JOIN 9 airport_scheme.ticket ticket ON flight.flight_id = ticket.flight_id 10 WHERE 11 flight.departure_date = CURRENT_DATE - 1 12 GROUP BY 13 flight.route_id, ticket.price, aircraft.seat_capacity; </pre>		
Data Output Messages Notifications		
Showing rows: 1 to 1 Page No: 1 of 1		
route_id integer	fail_sum bigint	
22	3052000	

- Определить, какой тип самолетов чаще всего летал в заданный аэропорт назначения.

```

SELECT
    airport.code_isao, aircraft.type
FROM
    airport_scheme.aircraft
JOIN
    airport_scheme.flight ON flight.aircraft_id = aircraft.aircraft_id
JOIN
    airport_scheme.route ON route.route_id = flight.route_id
JOIN
    airport_scheme.airport ON route.departure_airport_id = airport.airport_id
GROUP BY
    airport.code_isao, aircraft.type
HAVING
    COUNT(flight.flight_id) = (
        SELECT MAX(frequency) FROM (SELECT COUNT(flight.flight_id) AS
frequency
        FROM airport_scheme.flight
        JOIN airport_scheme.route ON route.route_id = flight.route_id
        JOIN airport_scheme.airport ON route.departure_airport_id =
airport.airport_id
        WHERE route.departure_airport_id = airport.airport_id
        GROUP BY flight.aircraft_id
    )

```

Query Query History

```

1 SELECT
2   airport.code_isao, aircraft.type
3 FROM
4   airport_scheme.aircraft
5 JOIN
6   airport_scheme.flight ON flight.aircraft_id = aircraft.aircraft_id
7 JOIN
8   airport_scheme.route ON route.route_id = flight.route_id
9 JOIN
10  airport_scheme.airport ON route.departure_airport_id = airport.airport_id
11 GROUP BY
12   airport.code_isao, aircraft.type
13 HAVING
14   COUNT(flight.flight_id) = (
15     SELECT MAX(frequency) FROM (SELECT COUNT(flight.flight_id) AS frequency
16     FROM airport_scheme.flight
17     JOIN airport_scheme.route ON route.route_id = flight.route_id
18     JOIN airport_scheme.airport ON route.departure_airport_id = airport.airport_id
19     WHERE route.departure_airport_id = airport.airport_id
20     GROUP BY flight.aircraft_id
21   )
22 );

```

Data Output Messages Notifications

Showing rows: 1 to 40 Page No: 1

	code_isao character varying (4)	type character varying (20)
1	OREN	грузовой
2	VABB	грузовой
3	KSFO	пассажирский
4	LEMD	пассажирский
5	RJTT	грузовой
6	NOVG	пассажирский
7	ZSPD	пассажирский
8	EDDB	пассажирский
9	YSSY	пассажирский
10	VOLG	пассажирский
11	UGAN	пассажирский
12	LEBL	пассажирский
13	UREG	пассажирский
14	WSSS	пассажирский

);

- Вывести список самолетов, “возраст” которых превышает средний “возраст” самолетов этого типа.

SELECT aircraft.aircraft_id From airport_scheme.aircraft
WHERE aircraft.production_year > (select AVG(aircraft.production_year) from airport_scheme.aircraft)

Query Query History Scratch Pad

```

1 SELECT aircraft.aircraft_id From airport_scheme.aircraft
2 WHERE aircraft.production_year > (select AVG(aircraft.production_year) from airport_scheme.aircraft)

```

Data Output Messages Notifications


Showing rows: 1 to 20 Page No: 1 of 1

	aircraft_id [PK] integer
1	2
2	3
3	8
4	11
5	12
6	14
7	16
8	19
9	20
10	22
11	27
12	28
13	31
14	33

Total rows: 20 Query complete 00:00:103 LF Ln 2, Col 101

- Определить тип самолетов, летающих во все аэропорты назначения.

```
WITH airport_counts AS (
  SELECT
    aircraft.type AS flight_type, COUNT(DISTINCT airport.code_isao) AS
airports_count
  FROM
    airport_scheme.aircraft
  JOIN
    airport_scheme.flight ON flight.aircraft_id = aircraft.aircraft_id
  JOIN
    airport_scheme.route ON route.route_id = flight.route_id
  JOIN
    airport_scheme.airport ON route.arrival_airport_id = airport.airport_id
  GROUP BY
    aircraft.type)
SELECT
  flight_type
FROM
  airport_counts
WHERE
  airports_count = (SELECT COUNT(DISTINCT code_isao) FROM
airport_scheme.airport)
```

flight_type
character varying (20) 

4.2 Представления

- Для пассажиров авиакомпании о рейсах в Москву на ближайшую неделю;
CREATE OR REPLACE VIEW view_moscow AS
SELECT
flight.flight_number,
flight.departure_time,
flight.arrival_time,

```

        flight.departure_date
FROM
    airport_scheme.flight
JOIN
    airport_scheme.route ON route.route_id = flight.flight_id
JOIN
    airport_scheme.airport ON airport.airport_id =
route.arrival_airport_id
JOIN
    airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id
WHERE
    airport.city = 'Москва' AND
    aircraft.type != 'грузовой' AND
    (flight.departure_date BETWEEN CURRENT_DATE AND
CURRENT_DATE + INTERVAL '7 days')

```

Query

Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

CREATE OR REPLACE VIEW wiew_moscow AS

SELECT

flight.flight_number,

flight.departure_time,

flight.arrival_time,

flight.departure_date

FROM

airport_scheme.flight

JOIN

airport_scheme.route ON route.route_id = flight.flight_id

JOIN

airport_scheme.airport ON airport.airport_id = route.arrival_airport_id

JOIN

airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id

WHERE

airport.city = 'Москва' AND

aircraft.type != 'грузовой' AND

(flight.departure_date BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '7 days')

Data Output

Messages

Notifications

CREATE VIEW

Query returned successfully in 81 msec.

Query Query History

1

2

3

4

5

6

7

8

9

10

11

12

13

14

CREATE OR REPLACE VIEW type_flight_month AS

SELECT

aircraft.type,

COUNT(*) as count_plane

FROM

airport_scheme.aircraft

JOIN

airport_scheme.flight ON flight.aircraft_id = aircraft.aircraft_id

WHERE

flight.departure_date BETWEEN CURRENT_DATE - INTERVAL '1 month' AND CURRENT_DATE

GROUP BY aircraft.type

Data Output Messages Notifications

CREATE VIEW

Query returned successfully in 95 msec.

Query Query History

1

SELECT * FROM type_flight_month

Data Output Messages Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	type character varying (20) 🔒	count_plane bigint 🔒
1	грузовой	3
2	пассажирский	28

4.3 Запросы на модификацию данных

- INSERT с подзапросом

Добавим билет для пассажира на рейс указанного тип самолета, который летит сегодня.

```
INSERT INTO airport_scheme.ticket (ticket_id, ticket_number, ticket_status,
purchase_date, seat_number, class_of_service, baggage_insurance, price,
flight_id, cashdesk_id, passport_data_id)
SELECT (SELECT COALESCE(MAX(ticket_id)) + 1 FROM
airport_scheme.ticket),
'TICK1',
'активен',
CURRENT_DATE,
(SELECT MIN(seat)
FROM generate_series(1, aircraft.seat_capacity) AS seat
WHERE seat NOT IN ( SELECT seat_number FROM airport_scheme.ticket
WHERE ticket.flight_id = flight.flight_id)
) AS seat_number,
'эконом',
TRUE,
15000,
flight.flight_id,
1,
passport_data.passport_data_id
FROM
airport_scheme.passport_data
JOIN
airport_scheme.flight ON TRUE
JOIN
airport_scheme.aircraft ON flight.aircraft_id = aircraft.aircraft_id
WHERE aircraft.type = 'пассажирский' AND flight.departure_date =
CURRENT_DATE
ORDER BY
flight.departure_date
LIMIT 1
ДО
```

Data Output

Messages

Notifications

Showing rows: 1 to 40

Page No: 1

of 1

	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer
30	30	T12374	использован	2025-04-15	38	бизнес	true	28000	30	30	30
31	31	T12375	активен	2025-04-16	39	эконом	false	15000	31	31	31
32	32	T12376	отменен	2025-04-14	40	первый	true	36000	32	32	32
33	33	T12377	активен	2025-04-17	41	эконом	true	15500	33	33	33
34	34	T12378	использован	2025-04-15	42	бизнес	false	27000	34	34	34
35	35	T12379	активен	2025-04-16	43	эконом	true	14500	35	35	35
36	36	T12380	отменен	2025-04-14	44	первый	false	33000	36	36	36
37	37	T12381	активен	2025-04-15	45	эконом	true	16000	37	37	37
38	38	T12382	использован	2025-04-13	46	бизнес	true	29000	38	38	38
39	39	T12383	активен	2025-04-16	47	эконом	false	15000	39	39	39
40	40	T12384	отменен	2025-04-14	48	первый	true	34000	40	40	40

ПОСЛЕ

Data Output Messages Notifications												
Showing rows: 1 to 41 Page No: 1 of 1												
	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer	
31	31	T12375	активен	2025-04-16	39	эконом	false	15000	31	31	31	31
32	32	T12376	отменен	2025-04-14	40	первый	true	36000	32	32	32	32
33	33	T12377	активен	2025-04-17	41	эконом	true	15500	33	33	33	33
34	34	T12378	использован	2025-04-15	42	бизнес	false	27000	34	34	34	34
35	35	T12379	активен	2025-04-16	43	эконом	true	14500	35	35	35	35
36	36	T12380	отменен	2025-04-14	44	первый	false	33000	36	36	36	36
37	37	T12381	активен	2025-04-15	45	эконом	true	16000	37	37	37	37
38	38	T12382	использован	2025-04-13	46	бизнес	true	29000	38	38	38	38
39	39	T12383	активен	2025-04-16	47	эконом	false	15000	39	39	39	39
40	40	T12384	отменен	2025-04-14	48	первый	true	34000	40	40	40	40
41	41	TICK1	активен	2025-05-21	1	эконом	true	15000	32	1	1	1

- UPDATE с подзапросом
Обновление цены билетов на рейсы следующего месяца.
UPDATE airport_scheme.ticket
SET price = price * 1.1
WHERE flight_id IN (SELECT flight_id FROM airport_scheme.flight
WHERE departure_date BETWEEN
CURRENT_DATE AND CURRENT_DATE + INTERVAL '1 month');
ДО

Data Output Messages Notifications												
Showing rows: 1 to 40 Page No: 1 of 1												
	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer	
1	1	T12345	активен	2025-04-15	12	эконом	true	15000	1	1	1	1
2	2	T12346	активен	2025-04-15	14	бизнес	false	30000	2	2	2	2
3	3	T12347	активен	2025-04-15	5	эконом	true	14000	3	3	3	3
4	4	T12348	использован	2025-04-14	8	первый	true	40000	4	4	4	4
5	5	T12349	активен	2025-04-16	3	эконом	false	12000	5	5	5	5
6	6	T12350	активен	2025-04-15	7	бизнес	true	25000	7	6	6	6
7	7	T12351	активен	2025-04-17	10	бизнес	false	25000	7	7	7	7
8	8	T12352	использован	2025-04-13	6	эконом	true	13000	8	8	8	8
9	9	T12353	активен	2025-04-15	15	первый	false	32000	9	9	9	9
10	10	T12354	отменен	2025-04-14	11	бизнес	true	27000	10	10	10	10
11	11	T12355	активен	2025-04-16	18	эконом	false	16000	11	11	11	11
12	12	T12356	активен	2025-04-17	20	первый	true	35000	12	12	12	12
13	13	T12357	использован	2025-04-14	21	эконом	false	14000	13	13	13	13
14	14	T12358	активен	2025-04-15	22	бизнес	true	29000	14	14	14	14
15	15	T12359	отменен	2025-04-16	23	эконом	true	14500	15	15	15	15
16	16	T12360	активен	2025-04-17	24	первый	false	34000	16	16	16	16
17	17	T12361	активен	2025-04-15	25	эконом	true	15500	17	17	17	17
18	18	T12362	активен	2025-04-13	26	бизнес	false	25000	7	18	18	18
19	19	T12363	активен	2025-04-16	27	эконом	true	16000	19	19	19	19
20	20	T12364	отменен	2025-04-14	28	первый	false	33000	20	20	20	20
21	21	T12365	активен	2025-04-17	29	эконом	true	15000	21	21	21	21
22	22	T12366	использован	2025-04-15	30	бизнес	true	28000	22	22	22	22
23	23	T12367	активен	2025-04-17	31	эконом	false	14000	23	23	23	23
24	24	T12368	отменен	2025-04-14	32	первый	true	34000	24	24	24	24
25	25	T12369	активен	2025-04-15	33	эконом	true	15000	25	25	25	25
26	26	T12370	активен	2025-04-13	34	бизнес	false	25000	7	26	26	26
27	27	T12371	активен	2025-04-16	35	эконом	true	16000	27	27	27	27
28	28	T12372	отменен	2025-04-14	36	первый	false	35000	28	28	28	28
29	29	T12373	активен	2025-04-17	37	эконом	true	15000	29	29	29	29
30	30	T12374	использован	2025-04-15	38	бизнес	true	28000	30	30	30	30
31	31	T12375	активен	2025-04-16	39	эконом	false	15000	31	31	31	31
32	32	T12376	отменен	2025-04-14	40	первый	true	36000	32	32	32	32
33	33	T12377	активен	2025-04-17	41	эконом	true	15500	33	33	33	33

ПОСЛЕ

	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer	
1	1	T12345	активен	2025-04-15	12	эконом	true	15000	1	1	1	1
2	2	T12346	активен	2025-04-15	14	бизнес	false	30000	2	2	2	2
3	3	T12347	активен	2025-04-15	5	эконом	true	14000	3	3	3	3
4	4	T12348	использован	2025-04-14	8	первый	true	40000	4	4	4	4
5	5	T12349	активен	2025-04-16	3	эконом	false	12000	5	5	5	5
6	6	T12350	активен	2025-04-15	7	бизнес	true	25000	7	6	6	6
7	7	T12351	активен	2025-04-17	10	бизнес	false	25000	7	7	7	7
8	8	T12352	использован	2025-04-13	6	эконом	true	13000	8	8	8	8
9	9	T12353	активен	2025-04-15	15	первый	false	32000	9	9	9	9
10	10	T12354	отменен	2025-04-14	11	бизнес	true	27000	10	10	10	10
11	11	T12355	активен	2025-04-16	18	эконом	false	16000	11	11	11	11
12	12	T12356	активен	2025-04-17	20	первый	true	35000	12	12	12	12
13	13	T12357	использован	2025-04-14	21	эконом	false	14000	13	13	13	13
14	14	T12358	активен	2025-04-15	22	бизнес	true	29000	14	14	14	14
15	15	T12359	отменен	2025-04-16	23	эконом	true	14500	15	15	15	15
16	16	T12360	активен	2025-04-17	24	первый	false	34000	16	16	16	16
17	17	T12361	активен	2025-04-15	25	эконом	true	15500	17	17	17	17
18	18	T12362	активен	2025-04-13	26	бизнес	false	25000	7	18	18	18
19	19	T12363	активен	2025-04-16	27	эконом	true	16000	19	19	19	19
20	20	T12364	отменен	2025-04-14	28	первый	false	33000	20	20	20	20
21	21	T12365	активен	2025-04-17	29	эконом	true	15000	21	21	21	21
22	22	T12366	использован	2025-04-15	30	бизнес	true	28000	22	22	22	22
23	23	T12367	активен	2025-04-17	31	эконом	false	14000	23	23	23	23
24	24	T12368	отменен	2025-04-14	32	первый	true	34000	24	24	24	24
25	25	T12369	активен	2025-04-15	33	эконом	true	15000	25	25	25	25
26	26	T12370	активен	2025-04-13	34	бизнес	false	25000	7	26	26	26
27	27	T12371	активен	2025-04-16	35	эконом	true	16000	27	27	27	27
28	28	T12372	отменен	2025-04-14	36	первый	false	35000	28	28	28	28
29	29	T12373	активен	2025-04-17	37	эконом	true	15000	29	29	29	29
30	30	T12374	использован	2025-04-15	38	бизнес	true	28000	30	30	30	30
31	31	T12375	активен	2025-04-16	39	эконом	false	15000	31	31	31	31
32	32	T12376	отменен	2025-04-14	40	первый	true	39600	32	32	32	32
33	33	T12377	активен	2025-04-17	41	эконом	true	17050	33	33	33	33

- DELETE с подзапросом
удаление билетов на рейсы с датой вылета раньше текущей даты
DELETE FROM airport_scheme.ticket
WHERE flight_id IN (
 SELECT flight_id
 FROM airport_scheme.flight
 WHERE departure_date < CURRENT_DATE
);
ДО

	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer	
1	1	T12345	активен	2025-04-15	12	эконом	true	15000	1	1	1	1
2	2	T12346	активен	2025-04-15	14	бизнес	false	30000	2	2	2	2
3	3	T12347	активен	2025-04-15	5	эконом	true	14000	3	3	3	3
4	4	T12348	использован	2025-04-14	8	первый	true	40000	4	4	4	4
5	5	T12349	активен	2025-04-16	3	эконом	false	12000	5	5	5	5
6	6	T12350	активен	2025-04-15	7	бизнес	true	25000	7	6	6	6
7	7	T12351	активен	2025-04-17	10	бизнес	false	25000	7	7	7	7
8	8	T12352	использован	2025-04-13	6	эконом	true	13000	8	8	8	8
9	9	T12353	активен	2025-04-15	15	первый	false	32000	9	9	9	9
10	10	T12354	отменен	2025-04-14	11	бизнес	true	27000	10	10	10	10
11	11	T12355	активен	2025-04-16	18	эконом	false	16000	11	11	11	11
12	12	T12356	активен	2025-04-17	20	первый	true	35000	12	12	12	12
13	13	T12357	использован	2025-04-14	21	эконом	false	14000	13	13	13	13
14	14	T12358	активен	2025-04-15	22	бизнес	true	29000	14	14	14	14
15	15	T12359	отменен	2025-04-16	23	эконом	true	14500	15	15	15	15
16	16	T12360	активен	2025-04-17	24	первый	false	34000	16	16	16	16
17	17	T12361	активен	2025-04-15	25	эконом	true	15500	17	17	17	17
18	18	T12362	активен	2025-04-13	26	бизнес	false	25000	7	18	18	18
19	19	T12363	активен	2025-04-16	27	эконом	true	16000	19	19	19	19
20	20	T12364	отменен	2025-04-14	28	первый	false	33000	20	20	20	20
21	21	T12365	активен	2025-04-17	29	эконом	true	15000	21	21	21	21
22	22	T12366	использован	2025-04-15	30	бизнес	true	28000	22	22	22	22
23	23	T12367	активен	2025-04-17	31	эконом	false	14000	23	23	23	23
24	24	T12368	отменен	2025-04-14	32	первый	true	34000	24	24	24	24
25	25	T12369	активен	2025-04-15	33	эконом	true	15000	25	25	25	25
26	26	T12370	активен	2025-04-13	34	бизнес	false	25000	7	26	26	26
27	27	T12371	активен	2025-04-16	35	эконом	true	16000	27	27	27	27
28	28	T12372	отменен	2025-04-14	36	первый	false	35000	28	28	28	28
29	29	T12373	активен	2025-04-17	37	эконом	true	15000	29	29	29	29
30	30	T12374	использован	2025-04-15	38	бизнес	true	28000	30	30	30	30
31	31	T12375	активен	2025-04-16	39	эконом	false	15000	31	31	31	31
32	32	T12376	отменен	2025-04-14	40	первый	true	39600	32	32	32	32
33	33	T12377	активен	2025-04-17	41	эконом	true	17050	33	33	33	33

Total rows: 41 Query complete 00:00:00.151

LF Ln 1, Col 1

	ticket_id [PK] integer	ticket_number character varying (15)	ticket_status character varying (15)	purchase_date date	seat_number integer	class_of_service character varying (20)	baggage_insurance boolean	price integer	flight_id integer	cashdesk_id integer	passport_data_id integer
1	32	T12376	отменен	2025-04-14	40	первый	true	39600	32	32	32
2	33	T12377	активен	2025-04-17	41	эконом	true	17050	33	33	33
3	34	T12378	использован	2025-04-15	42	бизнес	false	29700	34	34	34
4	35	T12379	активен	2025-04-16	43	эконом	true	15950	35	35	35
5	36	T12380	отменен	2025-04-14	44	первый	false	36300	36	36	36
6	37	T12381	активен	2025-04-15	45	эконом	true	17600	37	37	37
7	38	T12382	использован	2025-04-13	46	бизнес	true	31900	38	38	38
8	39	T12383	активен	2025-04-16	47	эконом	false	16500	39	39	39
9	40	T12384	отменен	2025-04-14	48	первый	true	37400	40	40	40
10	41	TICK1	активен	2025-05-21	1	эконом	true	16500	32	1	1

- Простой индекс
ДО создания индекса

Query

Query History

1

EXPLAIN ANALYZE

2


SELECT * FROM airport_scheme.passport_data WHERE passport_data_id = 140

Data Output


Messages

Notifications


≡+





▼




▼










SQL

QUERY PLAN

text



1

Seq Scan on passport_data (cost=0.00..4.50 rows=1 width=22) (actual time=0.215..0.237 rows=1 loops=...

2

Filter: (passport_data_id = 140)

3

Rows Removed by Filter: 199

4

Planning Time: 3.524 ms

5

Execution Time: 1.855 ms

После создания индекса

Query
Query History

1
CREATE INDEX idx_passport_data_id ON airport_scheme.passport_data(passport_data_id)

Data Output
Messages
Notifications

CREATE INDEX

Query returned successfully in 222 msec.

Query
Query History

1
2
EXPLAIN ANALYZE
SELECT * FROM airport_scheme.passport_data WHERE passport_data_id = 140

Data Output
Messages
Notifications

+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	QUERY PLAN	
	text	🔒
1	Index Scan using idx_passport_data_id on passport_data (cost=0.14..8.16 rows=1 width=22) (actual time=0.154..0.157 rows=1 loops=...	
2	Index Cond: (passport_data_id = 140)	
3	Planning Time: 0.227 ms	
4	Execution Time: 0.217 ms	

Query Query History

1 DROP INDEX airport_scheme.idx_passport_data_id

Data Output Messages Notifications

DROP INDEX

- Составной индекс
До создания индекса

Query Query History

```
1 1 EXPLAIN ANALYZE
2 SELECT * FROM airport_scheme.passport_data
3 WHERE (passenger_id BETWEEN 41 AND 140) AND series = 'AB'
4
5
```

Data Output Messages Notifications

SQL

	QUERY PLAN
	text
1	Seq Scan on passport_data (cost=10000000000.00..100000000005.50 rows=80 width=22) (actual time=0.136..0.189 rows=100 loops=...
2	Filter: ((passenger_id >= 41) AND (passenger_id <= 140) AND ((series)::text = 'AB'::text))
3	Rows Removed by Filter: 100
4	Planning Time: 3.511 ms
5	Execution Time: 1.464 ms

После создания индекса

Query Query History

1 CREATE INDEX idx_passport ON airport_scheme.passport_data(passenger_id, series)

Data Output Messages Notifications

CREATE INDEX

Query Query History

1 EXPLAIN ANALYZE
2 SELECT * FROM airport_scheme.passport_data
3 WHERE (passenger_id BETWEEN 41 AND 140) AND series = 'AB'
4
5

Data Output Messages Notifications



QUERY PLAN		text	
1	Bitmap Heap Scan on passport_data	(cost=5.41..8.81 rows=80 width=22) (actual time=0.069..0.102 rows=100 loops...	
2	Recheck Cond: ((passenger_id >= 41) AND (passenger_id <= 140) AND ((series)::text = 'AB'::text))		
3	Heap Blocks: exact=1		
4	-> Bitmap Index Scan on idx_passport	(cost=0.00..5.39 rows=80 width=0) (actual time=0.043..0.044 rows=100 loop...	
5	Index Cond: ((passenger_id >= 41) AND (passenger_id <= 140) AND ((series)::text = 'AB'::text))		
6	Planning Time: 0.200 ms		
7	Execution Time: 0.171 ms		

Query		Query History
1	DROP INDEX airport_scheme.idx_passport	
Data Output		Messages Notifications
DROP INDEX		

Выводы

В данной лабораторной работе были получены основные навыки создания представлений и запросов на выборку данных к базе данных PostgreSQL, а также навыки по работе с индексами.

