Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 4

«АНАЛИЗ ДАННЫХ. ПОСТРОЕНИЕ ИНФОЛОГИЧЕСКОЙ МОДЕЛИ ДАННЫХ БД»

по дисциплине «Проектирование и реализация баз данных»

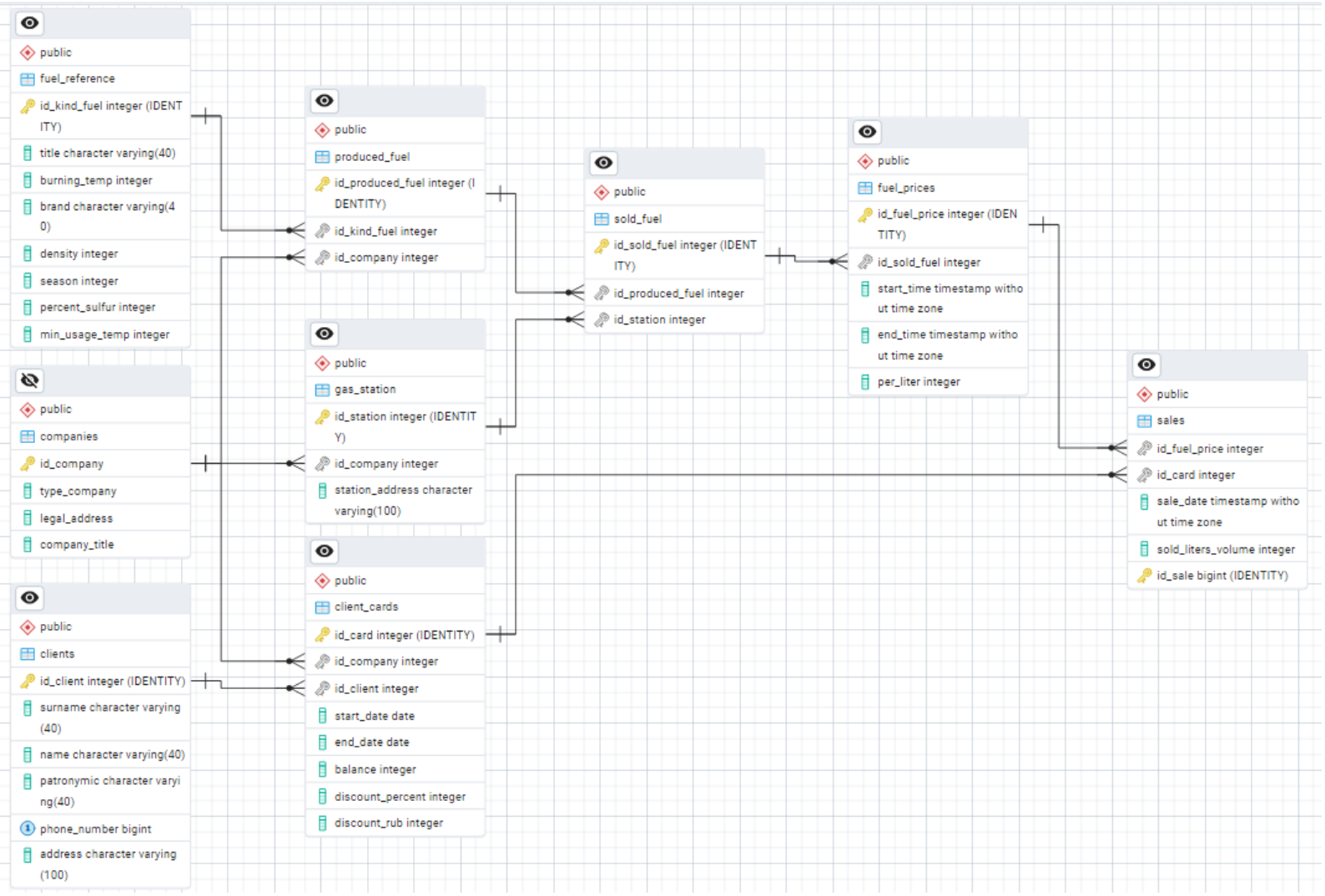
Обучающийся Проскуряков Роман Владимирович
Факультет прикладной информатики
Группа K3239
Направление подготовки 09.03.03 Прикладная информатика
Образовательная программа Мобильные и сетевые технологии 2023
Преподаватель Говорова Марина Михайловна

Санкт-Петербург
2024/2025

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 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.



Выполнение:

1. Запросы и представления на выборку данных к базе данных PostgreSQL

- Сколько раз заправлял автомобиль каждый из клиентов за заданный период.

```
SELECT clients.id_client, clients.surname, clients.name, clients.patronymic, count(id_sale)
FROM
clients
LEFT JOIN client_cards
ON clients.id_client = client_cards.id_client
LEFT JOIN sales
ON client_cards.id_card = sales.id_card
WHERE sale_date is NULL OR
    (sale_date >= '2023-01-01 00:00:00'
    AND sale_date <= '2024-01-10 00:00:00')
GROUP BY clients.id_client
```

	id_client [PK] integer	surname character varying (40)	name character varying (40)	patronymic character varying (40)	count bigint
1	10	Проскуряков	Роман	Владимирович	0
2	1	Иванов	Иван	Иванович	3
3	3	Сидоров	Сидор	Сидорович	5
4	2	Петров	Петр	Петрович	2

- Кто из клиентов не приобретал топливо в июле текущего года?

```
SELECT clients.id_client, clients.surname, clients.name, clients.patronymic FROM
clients
LEFT JOIN client_cards
ON clients.id_client = client_cards.id_client
LEFT JOIN sales
ON client_cards.id_card = sales.id_card
WHERE sale_date is NULL OR
    (EXTRACT(YEAR FROM sale_date) = EXTRACT(YEAR FROM CURRENT_DATE)
    AND EXTRACT(MONTH FROM sale_date) = 7)
GROUP BY clients.id_client
HAVING count(id_sale) = 0
```

	id_client [PK] integer	surname character varying (40)	name character varying (40)	patronymic character varying (40)
1	10	Проскуряков	Роман	Владимирович

- Найти клиента, купившего наибольший объем топлива по всей сети за истекший месяц.

```
WITH amount AS
(
    SELECT clients.id_client, clients.surname, clients.name, clients.patronymic,
sum(sold_liters_volume) AS S FROM
    clients
    LEFT JOIN client_cards
    ON clients.id_client = client_cards.id_client
    LEFT JOIN sales
    ON client_cards.id_card = sales.id_card
    WHERE sale_date >= date_trunc('month', CURRENT_DATE) - interval '1 month'
            AND sale_date < date_trunc('month', CURRENT_DATE)
    GROUP BY clients.id_client
)
```

```
SELECT amount.id_client, amount.surname, amount.name, amount.patronymic FROM
amount
WHERE amount.S in (SELECT max(amount.S) FROM amount)
```

	id_client [PK] integer	surname character varying (40)	name character varying (40)	patronymic character varying (40)
1	2	Петров	Петр	Петрович
2	3	Сидоров	Сидор	Сидорович

- Вывести данные клиента, купившего топлива на наибольшую сумму в заданный день.

```
WITH amount AS
(
    SELECT clients.id_client, clients.surname, clients.name, clients.patronymic,
    sum((sold_liters_volume * per_liter) * (100 - discount_percent) / 100 - discount_rub) AS
S FROM
    clients
    LEFT JOIN client_cards
    ON clients.id_client = client_cards.id_client
    LEFT JOIN sales
    ON client_cards.id_card = sales.id_card
    LEFT JOIN fuel_prices
    ON fuel_prices.id_fuel_price = sales.id_fuel_price
    WHERE sale_date::date = DATE('2025-07-04')
    GROUP BY clients.id_client
)

SELECT amount.id_client, amount.surname, amount.name, amount.patronymic FROM
amount
WHERE amount.S in (SELECT max(amount.S) FROM amount)
```

	id_client [PK] integer	surname character varying (40)	name character varying (40)	patronymic character varying (40)
1	1	Иванов	Иван	Иванович

- Какое топливо пользовалось наибольшим спросом в прошедшем году на АЗС конкретного поставщика?

```
WITH amount AS
(
    SELECT produced_fuel.id_produced_fuel, sum(sold_liters_volume) AS S FROM
    produced_fuel
    LEFT JOIN fuel_assortment
    ON produced_fuel.id_produced_fuel=fuel_assortment.id_produced_fuel
    LEFT JOIN filling_stations
    ON fuel_assortment.id_filling_station=filling_stations.id_filling_station
    LEFT JOIN fuel_prices
    ON fuel_assortment.id_fuel_offered=fuel_prices.id_fuel_offered
    LEFT JOIN sales
    ON fuel_prices.id_fuel_price=sales.id_fuel_price
    WHERE sale_date >= date_trunc('year', CURRENT_DATE) - interval '1 year'
            AND sale_date < date_trunc('year', CURRENT_DATE)
            AND filling_stations.id_company_owner=11
    GROUP BY produced_fuel.id_produced_fuel
)
```

```
SELECT amount.id_produced_fuel FROM
amount
WHERE amount.S in (SELECT max(amount.S) FROM amount)
```

	id_produced_fuel [PK] integer
1	13

- Сколько топлива каждого вида было продано за прошедший месяц по каждому поставщику на каждой заправке.

```
SELECT fuel_reference.id_kind_fuel, id_company_factory,
filling_stations.id_filling_station,
COALESCE(sum(sold_liters_volume), 0) AS S FROM
fuel_reference
LEFT JOIN
produced_fuel
ON fuel_reference.id_kind_fuel=produced_fuel.id_kind_fuel
LEFT JOIN fuel_assortment
ON produced_fuel.id_produced_fuel=fuel_assortment.id_produced_fuel
LEFT JOIN filling_stations
ON fuel_assortment.id_filling_station=filling_stations.id_filling_station
LEFT JOIN fuel_prices
ON fuel_assortment.id_fuel_offered=fuel_prices.id_fuel_offered
LEFT JOIN
    (SELECT * FROM sales
    WHERE sale_date >= date_trunc('month', CURRENT_DATE) - interval '1 month'
            AND sale_date < date_trunc('month', CURRENT_DATE)
    ) AS T
ON fuel_prices.id_fuel_price=T.id_fuel_price
GROUP BY fuel_reference.id_kind_fuel, filling_stations.id_filling_station,
id_company_factory
ORDER BY fuel_reference.id_kind_fuel
```

	id_kind_fuel integer	id_company_factory integer	id_filling_station integer	s bigint
1	16	12	9	0
2	16	12	10	0
3	16	12	11	0
4	17	9	7	0
5	18	13	8	25
6	18	12	9	0
7	18	12	10	0
8	19	13	8	20
9	19	13	12	25

- Какая из заправок продала топлива на наибольшую сумму по всем автозаправкам за последний год?

```
WITH amount AS
(
    SELECT filling_stations.id_filling_station,
    COALESCE(sum((sold_liters_volume * per_liter) * (100 - discount_percent) / 100 -
discount_rub), 0) AS S
```

```
    FROM
    filling_stations
    LEFT JOIN fuel_assortment
    ON fuel_assortment.id_filling_station=filling_stations.id_filling_station
    LEFT JOIN fuel_prices
    ON fuel_assortment.id_fuel_offered=fuel_prices.id_fuel_offered
    LEFT JOIN
            (SELECT * FROM sales
            WHERE EXTRACT(YEAR FROM sale_date) = EXTRACT(YEAR FROM
CURRENT_DATE)
            ) AS T
    ON fuel_prices.id_fuel_price=T.id_fuel_price
    LEFT JOIN client_cards
    ON T.id_card=client_cards.id_card
    GROUP BY filling_stations.id_filling_station
    ORDER BY filling_stations.id_filling_station
)

SELECT amount.id_filling_station FROM
amount
WHERE amount.S in (SELECT max(amount.S) FROM amount)
```

	id_filling_station [PK] integer
1	8

2. Представления

- Создать представление содержащее сведения обо всех АЗС и всех видах топлива, которые они продают

```
CREATE VIEW gas_station_fuel_summary AS
SELECT filling_stations.id_filling_station, id_kind_fuel FROM
filling_stations
LEFT JOIN
fuel_assortment
ON fuel_assortment.id_filling_station=filling_stations.id_filling_station
LEFT JOIN
produced_fuel
ON produced_fuel.id_produced_fuel=fuel_assortment.id_produced_fuel
ORDER BY filling_stations.id_filling_station, id_kind_fuel
```

	id_filling_station integer	id_kind_fuel integer
1	7	17
2	8	18
3	8	19
4	9	16
5	9	18
6	10	16
7	10	18
8	11	16
9	12	19
10	13	[null]

- Создать представление самая прибыльная АЗГС за истекший месяц для каждого производителя.

```
CREATE VIEW best_filling_stations_for_month AS
(
WITH amount AS
(
    SELECT filling_stations.id_company_owner, filling_stations.id_filling_station,
    COALESCE(sum((sold_liters_volume * per_liter) * (100 - discount_percent) / 100 -
discount_rub), 0) AS S
    FROM
    filling_stations
    LEFT JOIN fuel_assortment
    ON fuel_assortment.id_filling_station=filling_stations.id_filling_station
    LEFT JOIN fuel_prices
    ON fuel_assortment.id_fuel_offered=fuel_prices.id_fuel_offered
    LEFT JOIN
            (SELECT * FROM sales
            WHERE sale_date >= date_trunc('month', CURRENT_DATE) - interval '1 month'
                    AND sale_date < date_trunc('month', CURRENT_DATE)
            ) AS T
    ON fuel_prices.id_fuel_price=T.id_fuel_price
    LEFT JOIN client_cards
    ON T.id_card=client_           cards.id_card
    GROUP BY filling_stations.id_filling_station
```

```
    ORDER BY filling_stations.id_company_owner, filling_stations.id_filling_station
)

SELECT amount.id_company_owner, amount.id_filling_station, amount.S FROM
amount
JOIN
    (
    SELECT amount.id_company_owner, max(amount.S) AS M FROM
    amount
    GROUP BY amount.id_company_owner
    ) AS T
ON amount.id_company_owner=T.id_company_owner
    AND amount.S=T.M
)
```

	id_company_owner integer	id_filling_station integer	s bigint
1	8	7	0
2	8	13	0
3	10	8	21945
4	11	9	0
5	11	10	0
6	11	11	0

3. 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) **с использованием подзапросов**

- INSERT

Были записи о наличии в ассортименте без указанной цены. Исправим это.

```
SELECT id_fuel_offered FROM fuel_assortment
EXCEPT
SELECT id_fuel_offered FROM fuel_prices
```

	id_fuel_offered integer
1	15
2	14

```
WITH T AS
(
    SELECT id_fuel_offered FROM fuel_assortment
    EXCEPT
    SELECT id_fuel_offered FROM fuel_prices
)
INSERT INTO fuel_prices(id_fuel_offered, start_time, end_time, per_liter)
(
    SELECT id_fuel_offered, CURRENT_DATE as start_time, NULL as end_time, 9999999 as per_liter FROM T
)
```

```
INSERT 0 2

Запрос завершён успешно, время выполнения: 94 msec.
```

	id_fuel_offered integer

- UPDATE

Сделаем карты бессрочными, через которые, которые потратили у нас в сумме за предыдущий год больше, чем на 10000

	id_card [PK] integer	id_company integer	id_client integer	start_date date	end_date date	balance integer	discount_percent integer	discount_rub integer
1	1	8	1	2023-01-01	2024-01-01	1000	5	50
2	2	9	2	2023-02-01	2024-02-01	2000	10	100
3	3	10	3	2023-03-01	[null]	-5000	15	150
4	4	8	1	2023-01-01	2024-01-01	0	0	0
5	5	8	3	2023-01-01	2024-01-01	3000	0	0

```
UPDATE client_cards SET end_date=NULL WHERE id_card IN
(
    SELECT T2.id_card FROM
    (
            SELECT client_cards.id_card,
                    COALESCE(sum((sold_liters_volume * per_liter) * (100 - discount_percent) / 100 - discount_rub), 0) AS S
            FROM
            fuel_prices
```

```
            JOIN
                (SELECT * FROM sales
                WHERE sale_date >= date_trunc('year', CURRENT_DATE) - interval '1
year'
                    AND sale_date < date_trunc('year', CURRENT_DATE)
                ) AS T
            ON fuel_prices.id_fuel_price=T.id_fuel_price
            JOIN client_cards
            ON T.id_card=client_cards.id_card
            GROUP BY client_cards.id_card
            ORDER BY client_cards.id_card
    ) as T2 WHERE T2.S > 10000
)
```

Data Output | Сообщения | Notifications

```
UPDATE 3

Запрос завершён успешно, время выполнения: 104 msec.
```

	id_card [PK] integer	id_company integer	id_client integer	start_date date	end_date date	balance integer	discount_percent integer	discount_rub integer
1	2	9	2	2023-02-01	2024-02-01	2000	10	100
2	4	8	1	2023-01-01	2024-01-01	0	0	0
3	1	8	1	2023-01-01	[null]	1000	5	50
4	3	10	3	2023-03-01	[null]	-5000	15	150
5	5	8	3	2023-01-01	[null]	3000	0	0

- DELETE

Удалим из ассортимента все топлива, которые ни разу не были проданы (мы добавили их в шаге INSERT)

	id_fuel_offered [PK] integer	id_produced_fuel integer	id_filling_station integer
1	14	19	7
2	15	20	7

```
DELETE FROM fuel_assortment
WHERE NOT EXISTS
(
    SELECT 1 FROM
    fuel_prices
    LEFT JOIN sales
    ON sales.id_fuel_price=fuel_prices.id_fuel_price
    WHERE id_sale is NOT NULL
        AND fuel_prices.id_fuel_offered = fuel_assortment.id_fuel_offered
)
```

	id_fuel_offered [PK] integer	id_produced_fuel integer	id_filling_station integer

Также каскадно удалились и цены, т.к. я так настроил.

4. Простой и составной индексы для двух произвольных запросов и сравнение времени выполнения запросов без индексов и с индексами. Для получения плана запроса используется команда EXPLAIN

- Без индексов

```
EXPLAIN ANALYZE
SELECT * FROM clients
WHERE patronymic = 'Ryan-Jimenez'
AND address='354 Crystal Squares Mcdonaldport KY 66'
```

	QUERY PLAN text
1	Seq Scan on clients (cost=0.00..184.67 rows=1 width=70) (actual time=2.284..2.294 rows=1 loops=1)
2	Filter: (((patronymic)::text = 'Ryan-Jimenez'::text) AND ((address)::text = '354 Crystal Squares Mcdonaldport KY 66'::te...
3	Rows Removed by Filter: 6577
4	Planning Time: 0.179 ms
5	Execution Time: 2.317 ms

- С простым индексом

```
CREATE INDEX inx_simple ON clients(patronymic)
```

	QUERY PLAN text
1	Index Scan using inx_simple on clients (cost=0.28..8.30 rows=1 width=70) (actual time=0.063..0.064 rows=1 loops=...
2	Index Cond: ((patronymic)::text = 'Ryan-Jimenez'::text)
3	Filter: ((address)::text = '354 Crystal Squares Mcdonaldport KY 66'::text)
4	Planning Time: 2.087 ms
5	Execution Time: 0.085 ms

- С составным индексом

Сначала удалим прошлый идекс

```
DROP INDEX inx_simple
```

Теперь создадим новый

```
CREATE INDEX inx_hard ON clients(address, patronymic)
```

	QUERY PLAN text
1	Index Scan using inx_hard on clients (cost=0.28..8.30 rows=1 width=70) (actual time=0.107..0.109 rows=1 loops=1)
2	Index Cond: (((address)::text = '354 Crystal Squares Mcdonaldport KY 66'::text) AND ((patronymic)::text = 'Ryan-Jimenez'::te...
3	Planning Time: 0.185 ms
4	Execution Time: 0.133 ms

$$\frac{2.3}{0.1} = 23$$

С индексами с десятки раз быстрее!

Выводы

В ходе выполнения лабораторной работы были освоены ключевые навыки работы с PostgreSQL, такие как создание запросов, модификацию данных, использование подзапросов, а также оптимизацию запросов через индексы. Полученные знания и опыт будут полезны для дальнейшей работы с базами данных и разработки эффективных баз данных.