**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: Хисаметдинова Д.Н.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

**Оглавление**

[Цель работы 3](#_Toc148889485)

[Практическое задание 3](#_Toc148889487)

[Вариант 17. БД «Телефонный провайдер» 3](#_Toc148889495)

[Выполнение ЛР 4](#_Toc148889496)

[Вывод по работе 19](#_Toc148889498)35

# 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

# Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).

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

# Изучить графическое представление запросов и просмотреть историю запросов.

1. 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.

# Вариант 17. БД «Телефонный провайдер»

**Описание предметной области:** Информационная система служит для хранения информации об абонентах телефонной компании и для учета оплаты всех видов услуг абонентами.

Каждый абонент подключен к определенному тарифу. Тариф определяет базовое количество минут, ГБт, смс. Кроме того, он может подключить дополнительные услуги за отдельную плату. Необходимо знать текущий баланс клиента. У клиента могут быть подключены сторонние ресурсы, требующие оплаты, не зависящие от текущего тарифа.

Клиент может менять тариф.

В системе должны храниться сведения о продолжительности разговоров каждого абонента, о стоимости внутренних и междугородных переговоров, о задолженности абонента.

БД должна содержать следующий минимальный набор сведений: ФИО абонента. Номер телефона. Адрес абонента. Город. Зона (город, республика, СНГ, дальнее зарубежье). Страна. Стоимость тарифа. Сроки действия тарифа. Продолжительность разговора в минутах. Дата звонка. Время звонка. Код зоны. Цена минуты. Сумма оплаты. Дата оплаты. Статус оплаты. Дата фактической оплаты.

**Задание 2.** Создать запросы:

* Вывести суммарное время переговоров каждого абонента за заданный период.
* Найти среднюю продолжительность разговора абонента АТС.
* Вывести количество междугородных переговоров каждого абонента.
* Вывести список абонентов, не внѐсших оплату за прошедший месяц.
* Сколько звонков было сделано в каждый из следующих городов: в Москву, Лондон, Париж.
* Вывести список абонентов, звонивших только в ночное время.
* Вывести список абонентов, время разговоров которых превышает среднее для этой же зоны.

**Задание 3.** Создать представление:

* Содержащее сведения обо всех абонентах и их переговорах за прошедший месяц.
* Найти самую популярную зону звонков за истекший год..

**Задание 4.** Создать хранимые процедуры:

* вывести список всех звонков заданного абонента.
* вывести задолженность по оплате для заданного абонента.
* рассчитать общую стоимость звонков по каждой зоне за истекшую неделю.

**Задание 5.** Создать необходимые триггеры.

# Выполнение практического задания

# Запросы:

# Вывести суммарное время переговоров каждого абонента за заданный период.

1. SELECT
2. client\_id,
3. SUM(duration\_minutes) AS total\_duration\_minutes
4. FROM
5. (
6. SELECT
7. cl.client\_id,
8. EXTRACT(EPOCH FROM (dc.call\_end\_time - dc.call\_start\_time)) / 60 AS duration\_minutes
9. FROM
10. client cl
11. JOIN contract ct ON cl.client\_id = ct.client\_id
12. JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id
13. JOIN domestic\_call dc ON pt.phone\_on\_tariff\_number = dc.phone\_number
14. WHERE
15. dc.call\_start\_time >= '2000-01-01' AND
16. dc.call\_end\_time <= '2022-01-31'
17. UNION ALL
18. SELECT
19. cl.client\_id,
20. EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60 AS duration\_minutes
21. FROM
22. client cl
23. JOIN contract ct ON cl.client\_id = ct.client\_id
24. JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id
25. JOIN international\_call ic ON pt.phone\_on\_tariff\_number = ic.phone\_number
26. WHERE
27. ic.international\_call\_start\_time >= '2000-01-01' AND
28. ic.international\_call\_end\_time <= '2022-01-31'
29. ) combined\_calls
30. GROUP BY
31. client\_id;

# 

# 

# 2) Найти среднюю продолжительность разговора абонента АТС.

SELECT

  client\_id,

  AVG(duration\_minutes) AS average\_call\_duration,

  COUNT(\*) AS total\_calls,

  SUM(duration\_minutes) AS total\_duration\_minutes

FROM

  (

    SELECT

      cl.client\_id,

      EXTRACT(EPOCH FROM (dc.call\_end\_time - dc.call\_start\_time)) / 60 AS duration\_minutes

    FROM

      client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN domestic\_call dc ON pt.phone\_on\_tariff\_number = dc.phone\_number

    WHERE

      dc.call\_start\_time >= '2000-01-01' AND

      dc.call\_end\_time <= '2022-01-31' AND

      cl.client\_id = 87

    UNION ALL

    SELECT

      cl.client\_id,

      EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60 AS duration\_minutes

    FROM

      client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN international\_call ic ON pt.phone\_on\_tariff\_number = ic.phone\_number

    WHERE

      ic.international\_call\_start\_time >= '2000-01-01' AND

      ic.international\_call\_end\_time <= '2022-01-31' AND

      cl.client\_id = 87

  ) combined\_calls

GROUP BY

  client\_id;

# 

# 

# 3) Вывести количество международных переговоров каждого абонента.

SELECT client.client\_id, COUNT(\*) AS total\_international\_calls

FROM client

JOIN contract ON client.client\_id = contract.client\_id

JOIN phone\_on\_tariff ON contract.contract\_id = phone\_on\_tariff.contract\_id

JOIN international\_call ON phone\_on\_tariff.phone\_on\_tariff\_number = international\_call.phone\_number

GROUP BY client.client\_id;

# 

# 4) Вывести список абонентов, не внѐсших оплату за прошедший месяц.

# (в генераторе данных оплат в прошлом месяце вообще не было, поэтому добавила)

INSERT INTO "payment" ("payment\_id", "payment\_status", "phone\_number", "payment\_date", "payment\_amount") VALUES (101, True, '81839725897', '2023-10-13', '117');

INSERT INTO "payment" ("payment\_id", "payment\_status", "phone\_number", "payment\_date", "payment\_amount") VALUES (102, True, '85126663834', '2023-10-21', '535');

INSERT INTO "payment" ("payment\_id", "payment\_status", "phone\_number", "payment\_date", "payment\_amount") VALUES (103, True, '88686637097', '2023-10-18', '134');

SELECT client.client\_id

FROM client

WHERE NOT EXISTS (

  SELECT 1

  FROM contract

  JOIN phone\_on\_tariff ON contract.contract\_id = phone\_on\_tariff.contract\_id

  JOIN payment ON phone\_on\_tariff.phone\_on\_tariff\_number = payment.phone\_number

  WHERE client.client\_id = contract.client\_id

    AND payment.payment\_date >= date\_trunc('month', current\_date - interval '1 month')

    AND payment.payment\_date < date\_trunc('month', current\_date)

)

GROUP BY client.client\_id;

# 

# 5) Сколько звонков было сделано в каждый из следующих городов: в Москву, Лондон, Париж.

INSERT INTO "call\_zone" ("call\_zone\_id", "country", "cost\_per\_minute\_call\_zone", "region", "city") VALUES (16, 'Russia', '1', 'Central Russia', 'Moscow');

INSERT INTO "call\_zone" ("call\_zone\_id", "country", "cost\_per\_minute\_call\_zone", "region", "city") VALUES (17, 'United Kingdom', '100', 'England', 'London');

INSERT INTO "call\_zone" ("call\_zone\_id", "country", "cost\_per\_minute\_call\_zone", "region", "city") VALUES (18, 'France', '80', 'Paris region', 'Paris');

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (31, '84925489447', '2001-05-03 16:51:26', '2001-05-03 17:40:57', '84034176596', 16);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (32, '87972625997', '2016-09-23 12:16:19', '2016-09-23 13:01:25', '89991677667', 16);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (33, '84371531905', '2000-09-04 00:13:32', '2000-09-04 00:16:09', '86754876851', 18);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (34, '89622760197', '2017-08-09 05:11:22', '2017-08-09 05:13:39', '88803311209', 17);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (35, '89742497510', '2010-06-22 01:23:40', '2010-06-22 04:38:35', '82307566768', 16);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (36, '81539108668', '2004-10-03 02:29:31', '2004-10-03 10:04:15', '86300043270', 17);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (37, '88686637097', '2021-03-27 07:16:38', '2021-03-27 16:31:09', '81517335555', 16);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (38, '83164665019', '2003-01-05 22:58:43', '2003-01-05 23:21:03', '88963847481', 18);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (39, '84157873461', '2022-09-25 12:29:38', '2022-09-25 16:32:10', '80680854021', 16);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (40, '88686637097', '2023-01-11 20:18:06', '2023-01-11 22:09:58', '88427512477', 18);

-- Для Москвы

SELECT COUNT(\*) AS total\_calls\_to\_moscow

FROM international\_call

WHERE call\_zone\_id = (SELECT call\_zone\_id FROM call\_zone WHERE city = 'Moscow');

-- Для Лондона

SELECT COUNT(\*) AS total\_calls\_to\_london

FROM international\_call

WHERE call\_zone\_id = (SELECT call\_zone\_id FROM call\_zone WHERE city = 'London');

-- Для Парижа

SELECT COUNT(\*) AS total\_calls\_to\_paris

FROM international\_call

WHERE call\_zone\_id = (SELECT call\_zone\_id FROM call\_zone WHERE city = 'Paris');

# 

# 

# 

# 6) Вывести список абонентов, звонивших только в ночное время.

SELECT

  client\_id,

  SUM(duration\_minutes) AS total\_night\_duration\_minutes

FROM

  (

    SELECT

      cl.client\_id,

      EXTRACT(EPOCH FROM (dc.call\_end\_time - dc.call\_start\_time)) / 60 AS duration\_minutes,

      dc.call\_start\_time AT TIME ZONE 'UTC' as start\_time\_utc,

      dc.call\_end\_time AT TIME ZONE 'UTC' as end\_time\_utc

    FROM

      client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN domestic\_call dc ON pt.phone\_on\_tariff\_number = dc.phone\_number

    WHERE

      (dc.call\_start\_time::time >= '22:00' OR dc.call\_start\_time::time < '06:00') AND

      (dc.call\_end\_time::time >= '22:00' OR dc.call\_end\_time::time < '06:00')

    UNION ALL

    SELECT

      cl.client\_id,

      EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60 AS duration\_minutes,

      ic.international\_call\_start\_time AT TIME ZONE 'UTC' as start\_time\_utc,

      ic.international\_call\_end\_time AT TIME ZONE 'UTC' as end\_time\_utc

    FROM

      client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN international\_call ic ON pt.phone\_on\_tariff\_number = ic.phone\_number

    WHERE

      (ic.international\_call\_start\_time::time >= '22:00' OR ic.international\_call\_start\_time::time < '06:00') AND

      (ic.international\_call\_end\_time::time >= '22:00' OR ic.international\_call\_end\_time::time < '06:00')

  ) combined\_calls

GROUP BY

  client\_id;

# 

# 

# 7) Вывести список абонентов, время разговоров которых превышает среднее для этой же зоны.

# (Запрос, выводящий и client\_id, и среднюю продожительность в его call\_zone, и продолжительность конкретного звонка)

WITH CallDurations AS (

    SELECT

        cl.client\_id,

        ic.call\_zone\_id,

        EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60 AS call\_duration\_minutes

    FROM

        client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN international\_call ic ON pt.phone\_on\_tariff\_number = ic.phone\_number

),

AverageDurations AS (

    SELECT

        call\_zone\_id,

        AVG(call\_duration\_minutes) AS avg\_call\_duration\_minutes

    FROM

        CallDurations

    GROUP BY

        call\_zone\_id

)

SELECT

    cd.client\_id,

    cd.call\_duration\_minutes,

    ad.avg\_call\_duration\_minutes

FROM

    CallDurations cd

JOIN

    AverageDurations ad ON cd.call\_zone\_id = ad.call\_zone\_id

WHERE

    cd.call\_duration\_minutes > ad.avg\_call\_duration\_minutes;

# 

# (Запрос, непосредственно выводящий только client\_id)

WITH CallDurations AS (

    SELECT

        cl.client\_id,

        ic.call\_zone\_id,

        (EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60) AS call\_duration\_minutes

    FROM

        client cl

    JOIN contract ct ON cl.client\_id = ct.client\_id

    JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

    JOIN international\_call ic ON pt.phone\_on\_tariff\_number = ic.phone\_number

),

AverageDurations AS (

    SELECT

        call\_zone\_id,

        AVG(call\_duration\_minutes) AS avg\_call\_duration\_minutes

    FROM

        CallDurations

    GROUP BY

        call\_zone\_id

)

SELECT DISTINCT

    cd.client\_id

FROM

    CallDurations cd

JOIN

    AverageDurations ad ON cd.call\_zone\_id = ad.call\_zone\_id

WHERE

    cd.call\_duration\_minutes > ad.avg\_call\_duration\_minutes;

# 

# Задание 3. Создать представление:

# Содержащее сведения обо всех абонентах и их переговорах за прошедший месяц.

# (так как в прошлый месяц не было звонков, добавила INSERTами)

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (41, '88686637097', '2023-10-01 20:18:06', '2023-10-01 20:48:06', '88427505477', 2);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (42, '83939283170', '2023-10-23 16:19:05', '2023-10-23 16:39:05', '88007805173', 4);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (43, '84548540986', '2023-10-26 12:17:24', '2023-10-26 13:15:24', '83950008013', 5);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (44, '89175874533', '2023-10-27 18:25:57', '2023-10-27 19:25:57', '86924801576', 6);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (45, '87395751914', '2023-10-03 11:42:40', '2023-10-03 18:42:40', '81028402443', 14);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (46, '84608960717', '2023-10-02 12:38:06', '2023-10-02 14:58:06', '86600198499', 15);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (47, '83495694331', '2023-10-09 16:04:02', '2023-10-09 16:34:02', '85525676401', 29);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (48, '87816542499', '2023-10-08 12:42:37', '2023-10-08 16:42:37', '85666856990', 5);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (49, '81341681439', '2023-10-06 04:05:38', '2023-10-06 05:05:30', '84662915076', 5);

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (201, '87127921928', '2023-10-03 21:52:31', '2023-10-03 23:52:31', '88125094548');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (202, '81290751976', '2023-10-01 03:51:21', '2023-10-01 04:51:21', '89415803323');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (203, '86948251411', '2023-10-04 11:14:04', '2023-10-04 12:14:04', '89246801095');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (204, '82884816243', '2023-10-08 10:16:29', '2023-10-08 12:15:29', '84557604220');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (205, '86948251411', '2023-10-02 10:21:40', '2023-10-02 12:25:45', '83833805987');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (206, '83253580355', '2023-10-09 05:58:30', '2023-10-09 05:59:30', '88939347162');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (207, '85411706214', '2023-10-06 02:15:38', '2023-10-06 03:10:38', '81453154633');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (208, '82651331557', '2023-10-12 17:14:15', '2023-10-12 19:16:15', '85768773631');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (209, '81351445183', '2023-10-05 04:01:07', '2023-10-05 09:01:07', '82303663629');

INSERT INTO "domestic\_call" ("domestic\_call\_id", "phone\_number", "call\_start\_time", "call\_end\_time", "domestic\_callee\_number") VALUES (210, '84445401791', '2023-10-07 00:08:53', '2023-10-07 01:08:53', '81655367593');

# (Текст запроса на создание VIEW)

CREATE OR REPLACE VIEW client\_calls\_summary AS

SELECT

  cl.client\_id,

  COALESCE(SUM(dc.duration\_minutes), 0) AS total\_domestic\_duration\_minutes,

  COALESCE(SUM(ic.duration\_minutes), 0) AS total\_international\_duration\_minutes,

  COALESCE(COUNT(dc.domestic\_call\_id), 0) AS number\_of\_domestic\_calls,

  COALESCE(COUNT(ic.international\_call\_id), 0) AS number\_of\_international\_calls,

  STRING\_AGG(DISTINCT cz.country, ', ') AS countries\_called -- список стран международных звонков

FROM

  client cl

JOIN contract ct ON cl.client\_id = ct.client\_id

JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id

LEFT JOIN (

  SELECT domestic\_call\_id, phone\_number, EXTRACT(EPOCH FROM (call\_end\_time - call\_start\_time)) / 60 AS duration\_minutes

  FROM domestic\_call

  WHERE call\_start\_time >= date\_trunc('month', CURRENT\_DATE - INTERVAL '1 month') AND

        call\_start\_time < date\_trunc('month', CURRENT\_DATE)

) dc ON pt.phone\_on\_tariff\_number = dc.phone\_number

LEFT JOIN (

  SELECT international\_call\_id, phone\_number, call\_zone\_id, EXTRACT(EPOCH FROM (international\_call\_end\_time - international\_call\_start\_time)) / 60 AS duration\_minutes

  FROM international\_call

  WHERE international\_call\_start\_time >= date\_trunc('month', CURRENT\_DATE - INTERVAL '1 month') AND

        international\_call\_start\_time < date\_trunc('month', CURRENT\_DATE)

) ic ON pt.phone\_on\_tariff\_number = ic.phone\_number

LEFT JOIN call\_zone cz ON ic.call\_zone\_id = cz.call\_zone\_id

GROUP BY cl.client\_id;

# 

# 

# Найти самую популярную зону звонков за истекший год.

# (Вставляю новые даты за прошлый год)

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (50, '88686637097', '2022-10-01 20:18:06', '2022-10-01 20:48:06', '88427505478', 10);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (51, '83939283170', '2022-01-23 16:19:05', '2022-01-23 16:39:05', '88007805178', 4);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (52, '84548540986', '2022-12-26 12:17:24', '2022-12-26 13:15:24', '83950008018', 5);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (53, '89175874533', '2022-08-27 18:25:57', '2022-08-27 19:25:57', '86924801578', 2);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (54, '87395751914', '2022-05-03 11:42:40', '2022-05-03 18:42:40', '81028402448', 10);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (55, '84608960717', '2022-03-02 12:38:06', '2022-03-02 14:58:06', '86600198498', 10);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (56, '83495694331', '2022-07-09 16:04:02', '2022-07-09 16:34:02', '85525676408', 10);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (57, '87816542499', '2022-04-08 12:42:37', '2022-04-08 16:42:37', '85666856998', 5);

INSERT INTO "international\_call" ("international\_call\_id", "phone\_number", "international\_call\_start\_time", "international\_call\_end\_time", "international\_callee\_number", "call\_zone\_id") VALUES (58, '81341681439', '2022-06-06 04:05:38', '2022-06-06 05:05:30', '84662915078', 5);

CREATE OR REPLACE VIEW most\_popular\_call\_zone\_yearly AS

SELECT

  cz.call\_zone\_id,

  cz.city,

  cz.country,

  COUNT(\*) AS call\_count

FROM

  international\_call ic

JOIN call\_zone cz ON ic.call\_zone\_id = cz.call\_zone\_id

WHERE

  ic.international\_call\_start\_time >= date\_trunc('year', CURRENT\_DATE - INTERVAL '1 year') AND

  ic.international\_call\_start\_time < date\_trunc('year', CURRENT\_DATE)

GROUP BY

  cz.call\_zone\_id, cz.city, cz.country

ORDER BY

  call\_count DESC

LIMIT 1;

# 

# 

# Создание INSERT, UPDATE и DELETE запросов

# UPDATE запрос

UPDATE phone\_on\_tariff

SET current\_balance = current\_balance + 500

WHERE tariff\_id IN (

    SELECT tariff\_id FROM basic\_tariff WHERE gb\_amount < 100

)

AND phone\_on\_tariff\_number IN (

    SELECT pot.phone\_on\_tariff\_number

    FROM phone\_on\_tariff pot

    JOIN internal\_service\_inclusion isi ON pot.phone\_on\_tariff\_number = isi.phone\_number

    JOIN internal\_service isv ON isi.internal\_service\_id = isv.internal\_service\_id

    WHERE isv.internal\_service\_periodicity = 'yearly'

)

AND phone\_on\_tariff\_number NOT IN (

    -- Исключаю номера с количеством звонков выше среднего

    SELECT dc.phone\_number

    FROM domestic\_call dc

    GROUP BY dc.phone\_number

    HAVING AVG(EXTRACT(EPOCH FROM (dc.call\_end\_time - dc.call\_start\_time))) > (

        SELECT AVG(EXTRACT(EPOCH FROM (d.call\_end\_time - d.call\_start\_time)))

        FROM domestic\_call d

    )

);

# 

# INSERT

INSERT INTO payment (phone\_number, payment\_date, payment\_amount, payment\_status)

SELECT

    pot.phone\_on\_tariff\_number,

    CURRENT\_DATE,

    100.00,

    true

FROM

    phone\_on\_tariff pot

JOIN contract c ON pot.contract\_id = c.contract\_id

JOIN client cl ON c.client\_id = cl.client\_id

WHERE

    pot.tariff\_id IN (SELECT tariff\_id FROM basic\_tariff WHERE gb\_amount > 10)

AND NOT EXISTS (

    SELECT 1 FROM payment p

    WHERE p.phone\_number = pot.phone\_on\_tariff\_number

      AND p.payment\_date = CURRENT\_DATE

);

# 

# DELETE

DELETE FROM contract

WHERE client\_id IN (

    SELECT cl.client\_id

    FROM client cl

    JOIN phone\_on\_tariff pot ON cl.client\_id = (

        SELECT ct.client\_id

        FROM contract ct

        WHERE ct.client\_id = cl.client\_id

        LIMIT 1

    )

    LEFT JOIN external\_service\_inclusion esi ON pot.phone\_on\_tariff\_number = esi.phone\_number

    WHERE esi.external\_service\_id IS NULL -- Нет активных услуг

      AND pot.current\_balance < (

          SELECT AVG(p.current\_balance)

          FROM phone\_on\_tariff p

          WHERE p.tariff\_id = pot.tariff\_id

      )

      AND pot.deletion\_date IS NULL -- Активные тарифы

)

AND NOT EXISTS (

    -- Убедимся, что у клиента нет активных звонков

    SELECT 1

    FROM phone\_on\_tariff pt

    JOIN domestic\_call dc ON pt.phone\_on\_tariff\_number = dc.phone\_number

    WHERE pt.contract\_id = contract.contract\_id

      AND dc.call\_start\_time > CURRENT\_DATE - INTERVAL '1 year'

);

# Создание индексов

# Оператор EXPLAIN ANALYZE в запросе без индексов:

* EXPLAIN ANALYZE WITH CallDurations AS (
* SELECT
* cl.client\_id,
* ic.call\_zone\_id,
* EXTRACT(EPOCH FROM (ic.international\_call\_end\_time - ic.international\_call\_start\_time)) / 60 AS call\_duration\_minutes
* FROM
* client cl
* JOIN contract ct ON cl.client\_id = ct.client\_id
* JOIN phone\_on\_tariff pt ON ct.contract\_id = pt.contract\_id
* JOIN international\_call ic ON pt.phone\_number = ic.phone\_number
* ),
* AverageDurations AS (
* SELECT
* call\_zone\_id,
* AVG(call\_duration\_minutes) AS avg\_call\_duration\_minutes
* FROM
* CallDurations
* GROUP BY
* call\_zone\_id
* )
* SELECT DISTINCT
* cd.client\_id
* FROM
* CallDurations cd
* JOIN
* AverageDurations ad ON cd.call\_zone\_id = ad.call\_zone\_id
* WHERE
* cd.call\_duration\_minutes > ad.avg\_call\_duration\_minutes;

# 

# 

# 

# (Добавила простые индексы)

# 

# 

# 

# (Добавила составные индексы)

-- Составной индекс на contract\_id и client\_id в таблице contract

CREATE INDEX idx\_contract\_client ON contract(client\_id, contract\_id);

-- Составной индекс на phone\_number и contract\_id в таблице phone\_on\_tariff

CREATE INDEX idx\_phone\_on\_tariff\_contract ON phone\_on\_tariff(contract\_id, phone\_on\_tariff\_number);

-- Составной индекс на phone\_number и call\_zone\_id в таблице international\_call для ускорения JOIN

CREATE INDEX idx\_international\_call\_phone\_zone ON international\_call(phone\_number, call\_zone\_id);

-- Составной индекс для таблицы international\_call, который может помочь в фильтрации по датам и агрегации

CREATE INDEX idx\_international\_call\_date\_zone ON international\_call(call\_zone\_id, international\_call\_start\_time, international\_call\_end\_time);

# 

# 

# 

# Вывод: в данном SELECT-запросе использование простых индексов замедляет выполнение, так как он включает в себя много комплексных операций с данными, и запрос не оптимизируется, а только замедляется, тратя время на чтение этих индексов, а в случае со сложными индексами, написанными специально под этот запрос, всё наоборот – операция ускоряется, JOINы оптимизируются.

# 

# Создам простые индексы для другого запроса

# Снова EXPLAIN ANALYZE до индексов:

# 

-- Индекс на client\_id в таблице contract для улучшения связывания

CREATE INDEX idx\_contract\_client\_id ON contract(client\_id);

-- Индекс на contract\_id в таблице phone\_on\_tariff для улучшения связывания

CREATE INDEX idx\_phone\_on\_tariff\_contract\_id ON phone\_on\_tariff(contract\_id);

-- Индекс на phone\_number в таблице payment для улучшения связывания

CREATE INDEX idx\_payment\_phone\_number ON payment(phone\_number);

-- Индекс на payment\_date в таблице payment для улучшения фильтрации по дате

CREATE INDEX idx\_payment\_payment\_date ON payment(payment\_date);

# 

EXPLAIN ANALYZE

SELECT client.client\_id

FROM client

WHERE NOT EXISTS (

  SELECT 1

  FROM contract

  JOIN phone\_on\_tariff ON contract.contract\_id = phone\_on\_tariff.contract\_id

  JOIN payment ON phone\_on\_tariff.phone\_on\_tariff\_number = payment.phone\_number

  WHERE client.client\_id = contract.client\_id

    AND payment.payment\_date >= date\_trunc('month', current\_date - interval '1' month)

    AND payment.payment\_date < date\_trunc('month', current\_date)

)

GROUP BY client.client\_id;

# 

# (Создала составной индекс для данного запроса)

-- Составной индекс для фильтрации платежей во временном промежутке

CREATE INDEX idx\_payment\_date\_phone\_number ON payment(payment\_date, phone\_number);

# 

# 

# С помощью DROP INDEX удаляла индексы

# Вывод: этот запрос слишком прост для индексов, поэтому скорость: без индексов > сложные индексы > простые индексы

# Вывод по работе: были изучены практические навыки создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

SELECT

  cz.city,

  COUNT(ic.\*) AS total\_calls

FROM

  international\_call ic

JOIN

  call\_zone cz ON ic.call\_zone\_id = cz.call\_zone\_id

WHERE

  cz.city IN ('Moscow', 'London', 'Paris')

GROUP BY

  cz.city;

# 

CREATE OR REPLACE VIEW most\_popular\_call\_zone\_yearly AS

SELECT

  cz.call\_zone\_id,

  cz.city,

  cz.country,

  COUNT(\*) AS call\_count

FROM

  international\_call ic

JOIN call\_zone cz ON ic.call\_zone\_id = cz.call\_zone\_id

WHERE

  ic.international\_call\_start\_time BETWEEN date\_trunc('year', CURRENT\_DATE - INTERVAL '1 year') AND (date\_trunc('year', CURRENT\_DATE) - INTERVAL '1 second')

GROUP BY

  cz.call\_zone\_id, cz.city, cz.country

HAVING

  COUNT(\*) = (

    SELECT MAX(call\_count) FROM (

      SELECT

        COUNT(\*) AS call\_count

      FROM

        international\_call ic2

      JOIN call\_zone cz2 ON ic2.call\_zone\_id = cz2.call\_zone\_id

      WHERE

        ic2.international\_call\_start\_time >= date\_trunc('year', CURRENT\_DATE - INTERVAL '1 year') AND

        ic2.international\_call\_start\_time < date\_trunc('year', CURRENT\_DATE)

      GROUP BY

        cz2.call\_zone\_id

    ) AS max\_subquery

);

# 

# 

# Интересный факт😊

# 