



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА □ Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИИТ)

Кафедра цифровой трансформации (ЦТ)

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ

по дисциплине «Разработка баз данных»

Практическое занятие №3

Студенты группы

ИКБО-65-23 Олефиров.Г.Г.

(подпись)

Ассистент

Морозов Д.В.

(подпись)

Отчет представлен

«___»_____ 2025 г.

Москва 2025 г.

ПРАКТИЧЕСКАЯ РАБОТА №3.

Условная логика, подзапросы и обобщенные табличные выражения (CTE) в POSTGRES PRO

Постановка задачи: основываясь на индивидуальной схеме данных, составьте необходимые запросы:

Задание 1: использование оператора CASE 1. Составить запрос, использующий поисковое выражение CASE для категоризации данных по какому-либо числовому признаку из вашей БД (например, цена, количество, возраст). Запрос должен содержать не менее трех условий WHEN и ветку ELSE. 2. Составить запрос, в котором оператор CASE используется внутри агрегатной функции (например, SUM или COUNT) для выполнения условной агрегации.

Задание 2: использование подзапросов (часть 1) 1. Скалярный подзапрос: найти все записи в таблице, у которых значение в некотором числовом столбце превышает среднее (или максимальное/минимальное) значение по этому столбцу. 2. Многострочный подзапрос с IN: вывести информацию из одной таблицы на основе идентификаторов, полученных из связанной таблицы по определенному критерию (в данном случае, обязательно по дате).

Постановка задачи: основываясь на индивидуальной схеме данных, составьте необходимые запросы:

Задание 2: использование подзапросов (часть 2) 3. Коррелированный подзапрос с EXISTS: найти все записи из родительской таблицы, для которых существует хотя бы одна связанная запись в дочерней таблице, удовлетворяющая текстовому условию. 4. Альтернативное решение с JOIN: решите задачу из пункта выше (2.3, «Коррелированный подзапрос с EXISTS»), но на этот раз с использованием оператора соединения JOIN.

Задание 3: использование обобщенных табличных выражений (CTE). 1. Стандартное CTE: переписать запрос из Задания 2.3 («Коррелированный подзапрос с EXISTS») с использованием обобщенного табличного

выражения (CTE). 2. Рекурсивное СТЕ: используя имеющуюся в вашей схеме данных таблицу с иерархической структурой, написать рекурсивный запрос с помощью WITH RECURSIVE для вывода всей иерархии с указанием уровня вложенности.

1. Оператор CASE

The screenshot shows the DBeaver 25.0.5 interface. In the top navigation bar, the tabs are: Базы данных, Прекры, База данных, Справка. Below the navigation bar, there's a tree view of database objects under the schema 'olefirov_gg'. A script editor window contains the following SQL query:

```
SELECT
    title,
    Price,
    CASE
        WHEN Price >= 1000 THEN 'Дорогой продукт'
        WHEN Price BETWEEN 200 AND 999 THEN 'Средняя цена'
        WHEN Price > 0 AND Price < 200 THEN 'Дешевый продукт'
        ELSE 'Нет категории'
    END AS price_category
FROM Bakery_product;
```

The results of the query are displayed in a table viewer titled 'bakery_product 1'. The table has columns: Таблица, title, price, price_category. The data is as follows:

Таблица	title	price	price_category
1	Бородинский хлеб	85.5	Дешевый продукт
2	Батон нарезной	65	Дешевый продукт
3	Торт Наполеон	1 200	Дорогой продукт
4	Эклеры	45	Дешевый продукт
5	Пирог с яблоками	350	Средняя цена
6	Печенье овсяное	25	Дешевый продукт
7	Круассан	75	Дешевый продукт
8	Чизкейк	280	Средняя цена

Рисунок 1 - Категоризация товаров по цене

1.2. Агрегатная функция с CASE

The screenshot shows the DBeaver interface with a multi-step query being run against a database named 'olefirov_gg'. The query counts products based on price ranges across different bakeries.

```

SELECT
    b.Address AS bakery_address,
    COUNT(CASE WHEN bp.Price < 200 THEN 1 END) AS cheap_products,
    COUNT(CASE WHEN bp.Price BETWEEN 200 AND 999 THEN 1 END) AS medium_products,
    COUNT(CASE WHEN bp.Price >= 1000 THEN 1 END) AS expensive_products
FROM Bakery b
JOIN Bakery_product bp ON b.ID_Bakery = bp.ID_Bakery
GROUP BY b.Address;

```

The results are displayed in a table titled 'bakery_1' with columns: 'bakery_address', 'cheap_products', 'medium_products', and 'expensive_products'. The data is as follows:

bakery_address	cheap_products	medium_products	expensive_products
ул. Ленина, д. 25	2	0	0
пр. Победы, д. 10	1	0	1
ул. Мира, д. 5	1	1	0
ул. Садовая, д. 15	1	1	0

Рисунок 2 - подсчет дешевых/дорогих товаров в каждой пекарне

2. Подзапросы

The screenshot shows the DBeaver interface with a query that uses a scalar subquery to filter products based on their price relative to the average price of all products.

```

SELECT Title, Price
FROM Bakery_product
WHERE Price > (SELECT AVG(Price) FROM Bakery_product);

```

The results are displayed in a table titled 'bakery_product_1' with columns: 'title' and 'price'. The data is as follows:

title	price
Торт Наполеон	1 200
Пирог с яблоками	350
Чизкейк	280

Рисунок 3 - Скалярный подзапрос: товары дороже средней цены

2.1. Многострочный подзапрос

```

SELECT ID_Order, Order_amount, Order_time, Order_status
FROM "Order"
WHERE ID_Client IN (
    SELECT ID_Client
    FROM "Order"
    WHERE EXTRACT(YEAR FROM CURRENT_DATE) = 2025
);

```

	id_order	order_amount	order_time	order_status
1	1	1 500	10:30:00	Доставлен
2	2	850	11:45:00	В обработке
3	3	1 200	12:15:00	Готов к выдаче
4	4	650	13:20:00	Доставляется
5	5	1 800	14:30:00	Принят

Рисунок 4 - Заказы, сделанные в 2025 году

2.2. Коррелированный подзапрос

```

SELECT c.Surname, c.Name, c.Phone_number
FROM Client c
WHERE EXISTS (
    SELECT 1
    FROM Review r
    WHERE r.ID_Client = c.ID_Client
    AND r.Evaluation >= 4
);

```

	surname	name	phone_number
1	Иванов	Иван	+7 (916) 123-45-67
2	Петрова	Мария	+7 (917) 234-56-78
3	Сидоров	Алексей	+7 (918) 345-67-89
4	Смирнов	Дмитрий	+7 (920) 567-89-01

Рисунок 5 - Клиенты, оставившие отзывы с оценкой ≥ 4

DBeaver 25.0.5 - <dbstud> Script-5

```

SELECT DISTINCT c.Surname, c.Name, c.Phone_number
FROM Client c
JOIN Review r ON c.ID_Client = r.ID_Client
WHERE r.evaluation >= 4;
    
```

client 1

	Фамилия	Имя	Номер телефона
1	Иванов	Иван	+7 (916) 123-45-67
2	Петрова	Мария	+7 (917) 234-56-78
3	Смирнов	Дмитрий	+7 (920) 567-89-01
4	Сидоров	Алексей	+7 (918) 345-67-89

Рисунок 6 – Альтернатива через JOIN:

3. Обобщённые табличные выражения (CTE)

DBeaver 25.0.5 - <dbstud> Script-5

```

WITH good_reviews AS (
  SELECT DISTINCT ID_Client
  FROM Review
  WHERE Evaluation >= 4
)
SELECT c.Surname, c.Name, c.Phone_number
FROM Client c
JOIN good_reviews gr ON c.ID_Client = gr.ID_Client;
    
```

client 1

	Фамилия	Имя	Номер телефона
1	Иванов	Иван	+7 (916) 123-45-67
2	Петрова	Мария	+7 (917) 234-56-78
3	Сидоров	Алексей	+7 (918) 345-67-89
4	Смирнов	Дмитрий	+7 (920) 567-89-01

Рисунок 7 - Переписываем задание 2.3 через WITH

The screenshot shows the DBeaver interface with a SQL script editor and a results grid.

SQL Script Content:

```
WITH RECURSIVE staff AS (
    -- корень: сотрудники без начальника
    SELECT
        ID_Employee,
        Surname,
        Name,
        Manager_ID,
        0 AS level
    FROM Employee
    WHERE Manager_ID IS NULL
    UNION all
    -- рекурсивная часть: подчинённые
    SELECT
        e.ID_Employee,
        e.Surname,
        e.Name,
        e.Manager_ID,
        s.level + 1
    FROM Employee e
    JOIN staff s ON e.Manager_ID = s.ID_Employee
)
SELECT repeat(' ', Level * 4) || Surname || ' ' || Name AS hierarchy, Level
FROM staff
ORDER BY level, hierarchy;
```

Results Grid:

Таблица	hierarchy	level
1	Дмитриева Анна	0
2	Васильев Андрей	1
3	Козлова Ирина	1
4	Николаева Ольга	1
5	Федоров Сергей	1

Рисунок 8 - Построение дерева иерархии сотрудников