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 № 3

по дисциплине «Разработка баз данных»

Студент группы *ИНБО-12-23. Албахтин И.В.* _____
(подпись)

Ассистент *Брайловский А.В.* _____
(подпись)

Москва 2025 г.

ПРАКТИЧЕСКАЯ РАБОТА №3. УСЛОВНАЯ ЛОГИКА, ПОДЗАПРОСЫ И ОБОБЩЕННЫЕ ТАБЛИЧНЫЕ ВЫРАЖЕНИЯ (CTE) В POSTGRES PRO

Цель:

Работа направлена на формирование глубокого понимания и практического применения инструментов для реализации сложной бизнес-логики непосредственно на уровне базы данных

Постановка задачи:

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.

Составить и выполнить три запроса, демонстрирующих разные типы подзапросов.

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).

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 (с коррелированным подзапросом) с использованием обобщенного табличного выражения (CTE).

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

ВЫПОЛНЕНИЕ ПРАКТИЧЕСКОЙ РАБОТЫ

Таблица 1. Таблица car (автомобиль)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 car_id	1	int4			[v]		
123 client_id	2	int4			[ ]		
A-Z brand	3	varchar(30)		default	[ ]		
A-Z model	4	varchar(30)		default	[ ]		
123 year	5	int4			[ ]		
A-Z license_plate	6	varchar(17)		default	[ ]		

Таблица 2. Таблица client (клиент)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 client_id	1	int4			[v]		
A-Z name	2	varchar(15)		default	[ ]		
A-Z surname	3	varchar(15)		default	[ ]		
A-Z phone	4	varchar(15)		default	[ ]		
A-Z email	5	varchar(254)		default	[ ]		

Таблица 3. Таблица diagnosis (диагностика)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 diagnosis_id	1	serial4			[v]	nextval('diagnosis_diagnosis_id_seq'::regclass)	
123 maintenance_id	2	int4			[ ]		
A-Z result	3	text		default	[ ]		

Таблица 4. Таблица invoice (счёт за работы)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 invoice_id	1	serial4			[v]	nextval('invoice_invoice_id_seq'::regclass)	
123 maintenance_id	2	int4			[ ]		
123 total_amount	3	numeric			[ ]		
A-Z payment_status	4	varchar(20)		default	[ ]		

Таблица 5. Таблица maintenance (обслуживание)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 maintenance_id	1	serial4			[v]	nextval('maintenance_maintenance_id_seq'::regclass)	
123 car_id	2	int4			[ ]		
123 worker_id	3	int4			[ ]		
123 part_id	4	int4			[ ]		
start_date	5	date			[ ]		
end_date	6	date			[ ]		
A-Z status	7	varchar(20)		default	[ ]	'planned'::character varying	

Таблица 6. Таблица maintenance_work (соединительная таблица между обслуживанием и типом работы)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	К
123 maintenance_work	1	serial4			[v]	nextval('maintenance_work_maintenance_work_seq'::regclass)	
123 maintenance_id	2	int4			[ ]		
123 work_type_id	3	int4			[ ]		

Таблица 7. Таблица part (запчасти)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
123 part_id	1	int4			[v]		
A-Z name	2	varchar(100)		default	[ ]		
123 price	3	numeric			[ ]		
123 supplier_id	4	int4			[ ]		

Таблица 8. Таблица part_warehouse (соединительная таблица между складом и запчастями)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	К
123 part_id	1	int4			[v]		
123 warehouse_id	2	int4			[ ]		
123 quantity	3	int4			[ ]		

Таблица 9. Таблица review (отзывы)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию
123 review_id	1	serial4			[v]	nextval('review_review_id_seq'::regclass)
123 client_id	2	int4			[ ]	
A-Z text	3	varchar(1000)		default	[ ]	
123 rating	4	int4			[ ]	
date	5	date			[ ]	

Таблица 10. Таблица supplier (поставщики)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию
123 supplier_id	1	int4			[v]	
A-Z name	2	varchar(300)		default	[ ]	
A-Z phone	3	varchar(15)		default	[ ]	

Таблица 11. Таблица warehouse (склад)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию
123 warehouse_id	1	int4			[v]	
A-Z address	2	varchar(100)		default	[ ]	

Таблица 12. Таблица warranty (гарантия)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	К
123 warranty_id	1	serial4			[v]	nextval('warranty_warranty_id_seq'::regclass)	
123 maintenance_id	2	int4			[ ]		
expiry_date	3	date			[ ]		

Таблица 13. Таблица work_type (тип работ)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умс
123 work_type_id	1	int4			[v]	
A-Z name	2	varchar(500)		default	[ ]	
A-Z description	3	text		default	[ ]	

Таблица 14. Таблица worker (сотрудник)

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию
123 worker_id	1	int4			[v]	
A-Z name	2	varchar(15)		default	[ ]	
A-Z position	3	varchar(20)		default	[ ]	
A-Z phone	4	varchar(15)		default	[ ]	

Задание 1. Использование CASE

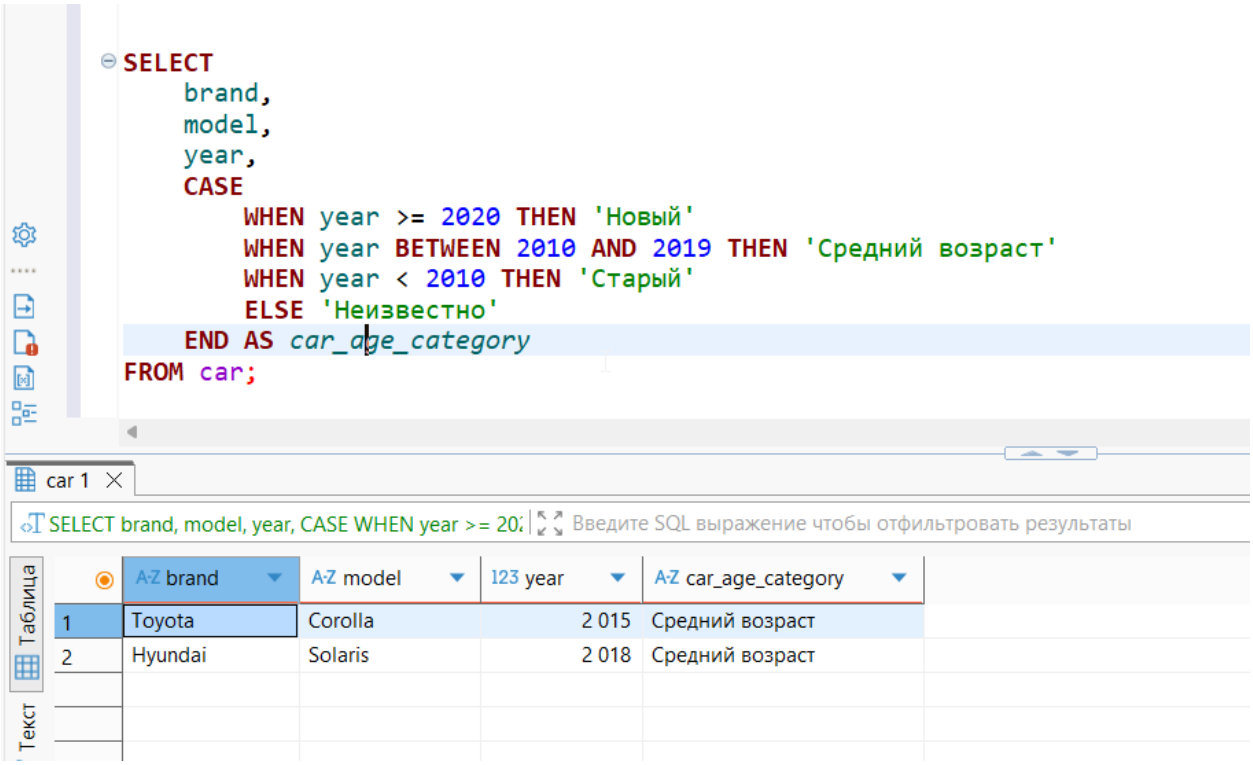

Рисунок 1 - Категоризируем автомобили по году выпуска

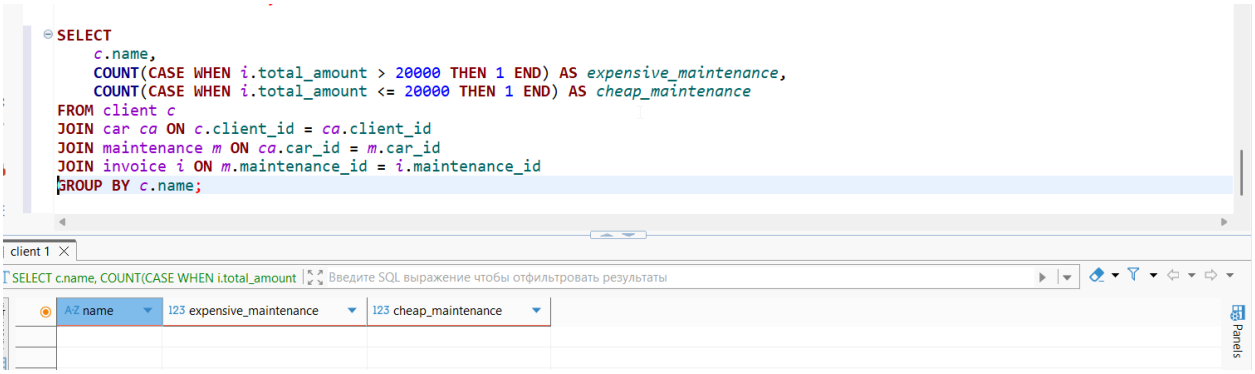

Рисунок 2 - Посчитаем, сколько ТО у каждого клиента было «дорогих» (сумма > 20 000) и «дешевых»

Задание 2. Подзапросы

```
SELECT
    i.invoice_id,
    i.total_amount,
    i.payment_status
FROM invoice i
WHERE i.total_amount > (SELECT AVG(total_amount) FROM invoice);
```

invoice 1

SELECT i.invoice_id, i.total_amount, i.payment_status | Введите SQL выражение чтобы отфильтровать результаты

	invoice_id	total_amount	payment_status
1	1	2 300	Оплачено

Рисунок 3 – Вычисляем среднюю сумму всех счетов

```
SELECT c.client_id, c.name
FROM client c
WHERE c.client_id IN (
    SELECT ca.client_id
    FROM car ca
    JOIN maintenance m ON ca.car_id = m.car_id
    JOIN invoice i ON m.maintenance_id = i.maintenance_id
    WHERE m.start_date >= (CURRENT_DATE - INTERVAL '1 year')
);
```

1

T c.client_id, c.name FROM client c WHERE c.cl | Введите SQL выражение чтобы отфильтровать результаты

client_id	name

Рисунок 4 – Обслуживания начатые в последний год

```sql
SELECT w.worker_id, w.name
FROM worker w
WHERE EXISTS (
    SELECT 1
    FROM maintenance m
    JOIN maintenance_work mw ON m.maintenance_id = mw.maintenance_id
    JOIN work_type wt ON mw.work_type_id = wt.work_type_id
    WHERE m.worker_id = w.worker_id      -- ← вот это делает подзапрос "коррелированным"
      AND LOWER(wt.name) LIKE '%замена%'
);

SELECT maintenance_id, worker_id
```

worker 1

SELECT w.worker_id, w.name FROM worker w WHERE | Введите SQL выражение чтобы отфильтровать результаты

	worker_id	name
1	1	Сергей

Рисунок 5 – Найти тип работы со словом «замена» по имени сотрудника через EXISTS

```sql
SELECT DISTINCT w.worker_id, w.name AS worker_name
FROM worker w
JOIN maintenance m ON w.worker_id = m.worker_id
JOIN maintenance_work mw ON m.maintenance_id = mw.maintenance_id
JOIN work_type wt ON mw.work_type_id = wt.work_type_id
WHERE LOWER(wt.name) LIKE '%замена%';
```

worker 1

SELECT DISTINCT w.worker_id, w.name AS worker_na | Введите SQL выражение чтобы отфильтровать результаты

	worker_id	worker_name
1	1	Сергей

Рисунок 6 - Найти тип работы со словом «замена» по имени сотрудника через JOIN

Задание 3. Использование обобщенных табличных выражений (CTE)

```sql
WITH replacement_workers AS (
    SELECT DISTINCT m.worker_id
    FROM maintenance m
    JOIN maintenance_work mw ON m.maintenance_id = mw.maintenance_id
    JOIN work_type wt ON mw.work_type_id = wt.work_type_id
    WHERE LOWER(wt.name) LIKE '%замена%'
)
SELECT w.worker_id, w.name
FROM worker w
JOIN replacement_workers rw ON w.worker_id = rw.worker_id;
```

orker 1

ITH replacement_workers AS (SELECT DISTINCT m | Введите SQL выражение чтобы отфильтровать результаты

	123 worker_id	A-Z name
1	1	Сергей

Рисунок 7 - Найти тип работы со словом «замена» по имени сотрудника через CTE

```sql
);

INSERT INTO categories (category_id, name, parent_id) VALUES
(1, 'Автомобили', NULL),
(2, 'Ремонт', 1),
(3, 'Запчасти', 1),
(4, 'Замена масла', 2),
(5, 'Тормозная система', 2),
(6, 'Фильтры', 3);

SELECT * FROM categories;
```

categories 1

SELECT * FROM categories | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст | Запись

	123 category_id	A-Z name	123 parent_id
1	1	Автомобили	[NULL]
2	2	Ремонт	1
3	3	Запчасти	1
4	4	Замена масла	2
5	5	Тормозная система	2
6	6	Фильтры	3

Значение

1

Рисунок 8 – Создание новой таблицы “Categories”

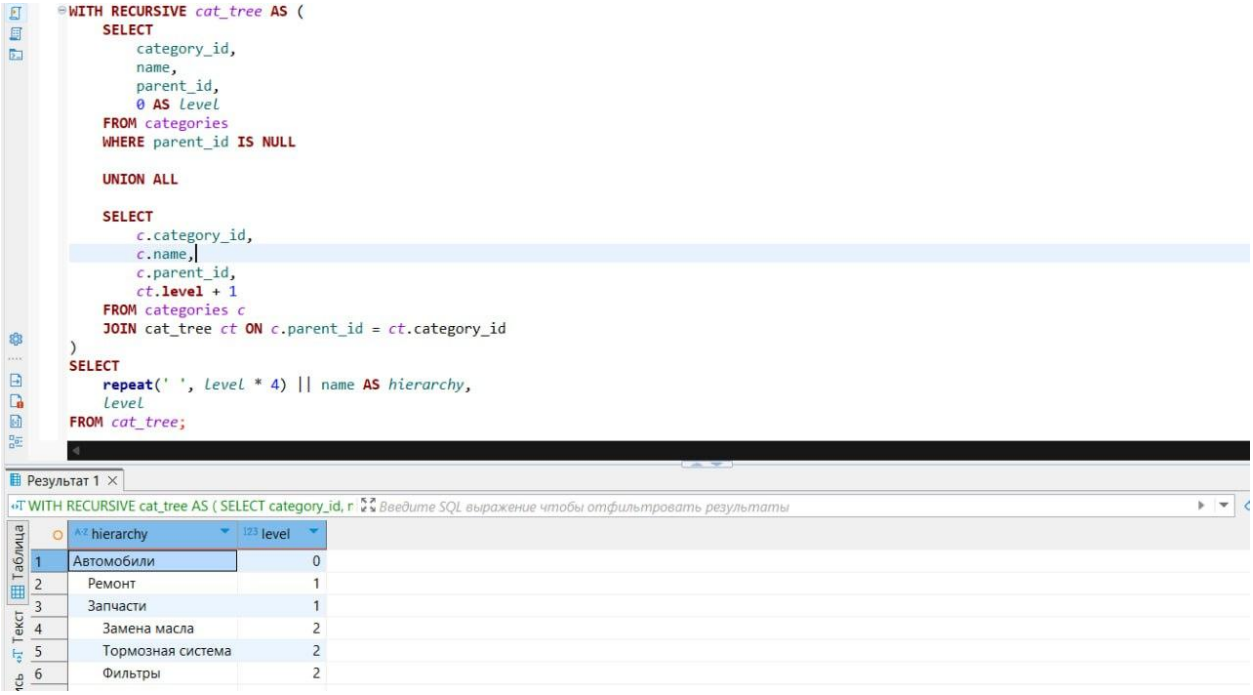

Рисунок 9 – Рекурсивное CTE

ВЫВОД

В ходе работы были изучены и реализованы основные приёмы работы с SQL-запросами: агрегатные функции, вложенные и коррелированные подзапросы, оператор EXISTS, а также рекурсивные запросы WITH RECURSIVE для построения иерархических структур. На практике удалось закрепить навыки соединения таблиц, фильтрации данных и анализа содержимого базы. Работа позволила понять, как формировать сложные выборки и использовать SQL для решения реальных задач обработки данных.