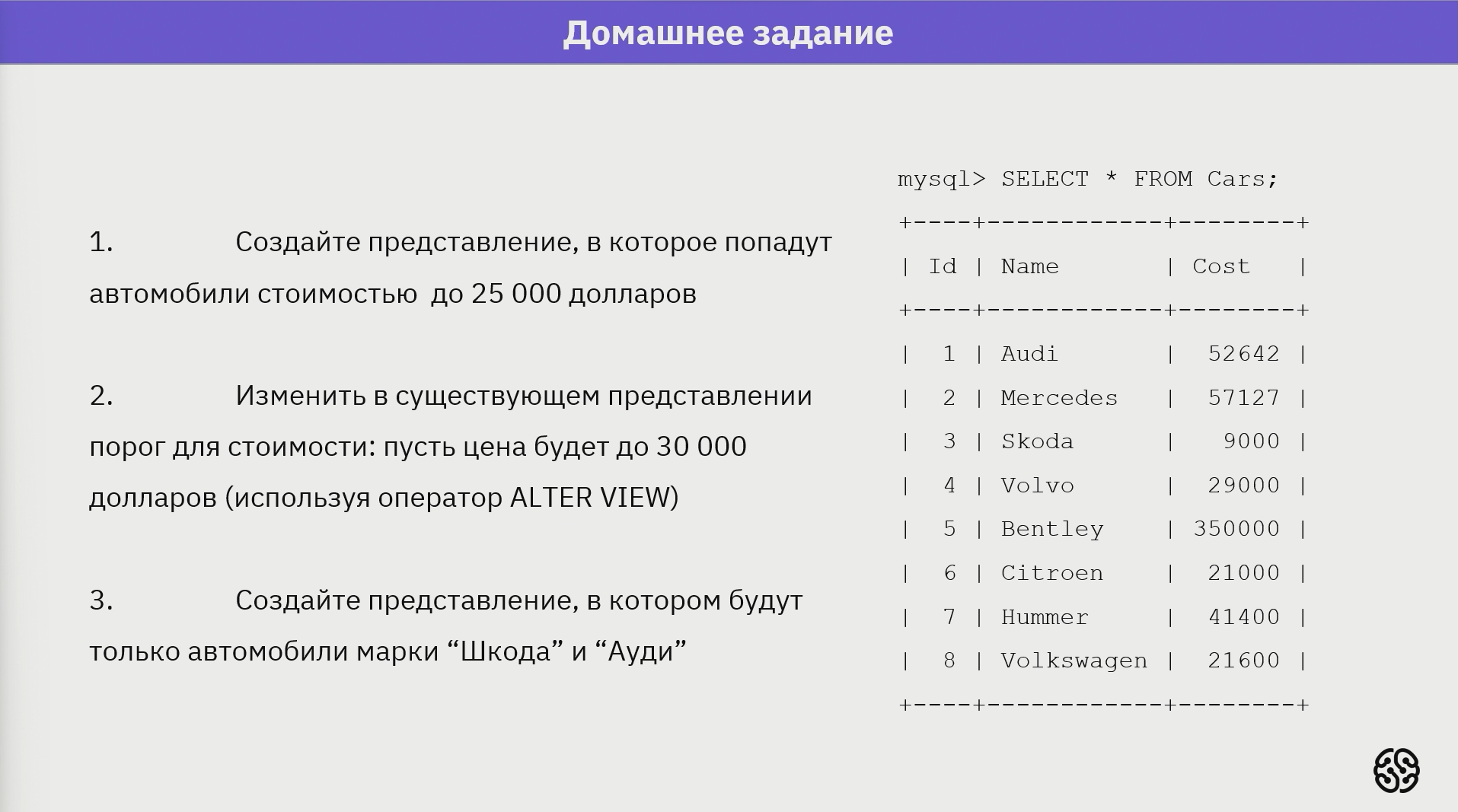
**Базы данных и SQL. Обучение в записи**

**Урок 10. Семинар: SQL – оконные функции**

*Ссылка на репозиторий, с выполненным Д/З:*

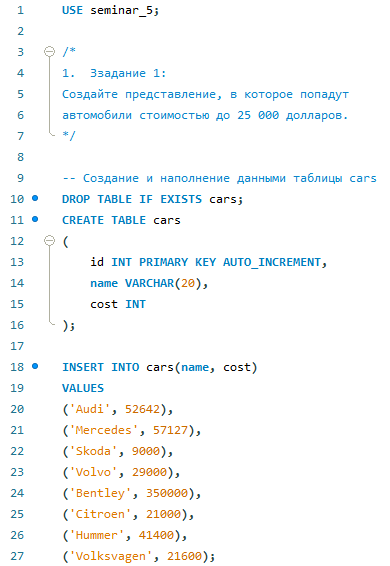
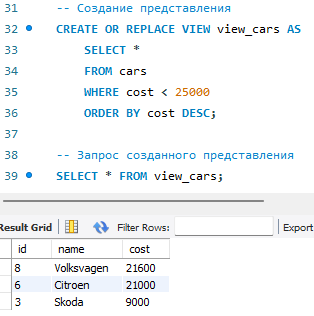
[*https://github.com/olgashenkel/Databases-and-SQL*](https://github.com/olgashenkel/Databases-and-SQL)

**ЧАСТЬ I.**

****

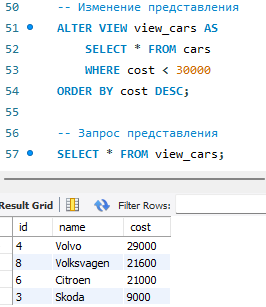
1. ***Ход выполнения задания 1.1:***

*Создайте представление, в которое попадут автомобили стоимостью до 25 000 долларов.*

***1)*** ***2) ***

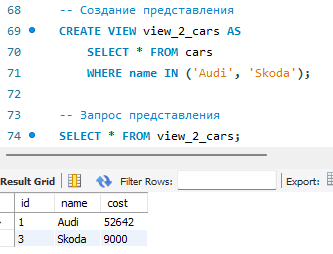
1. ***Ход выполнения задания 1.2:***

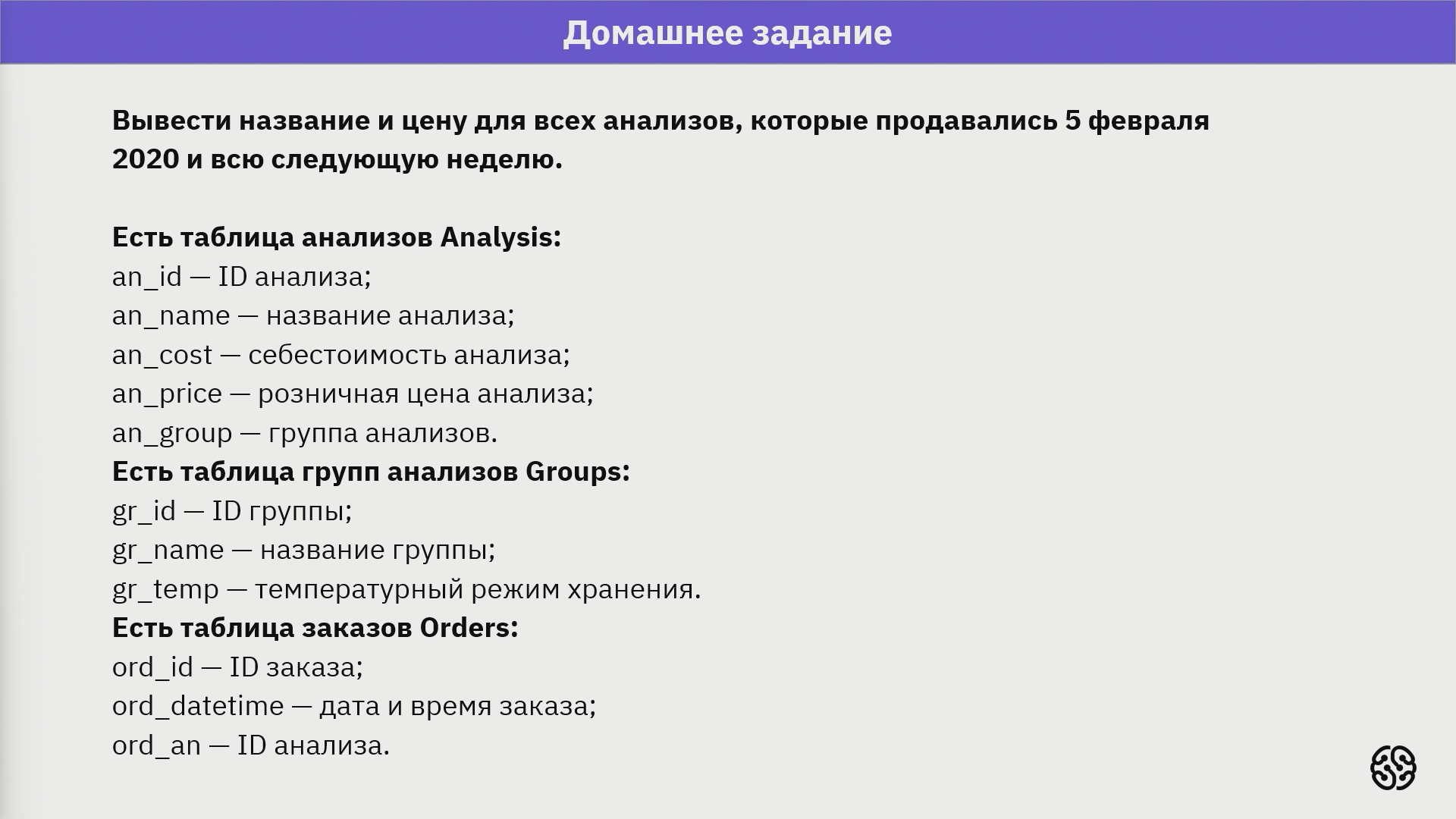
*Изменить в существующем представлении порог стоимости: пусть цена будет до 30000 долларов (используя оператор ALTER VIEW).*

**

1. ***Ход выполнения задания 1.3:***

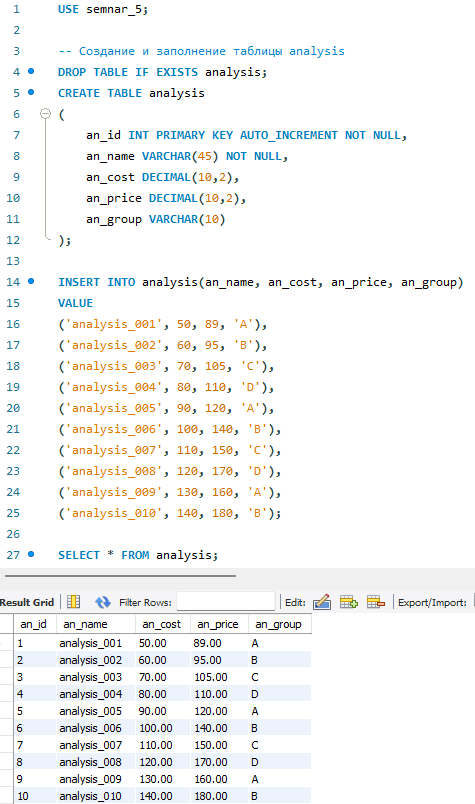
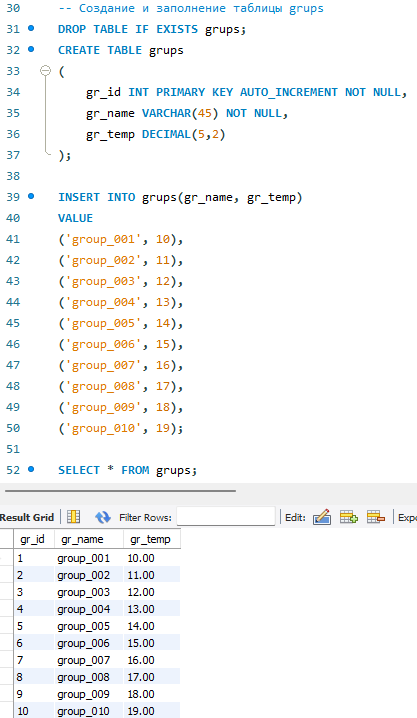
*Создайте представление, в котором будут только автомобили марки Skoda и Audi.*

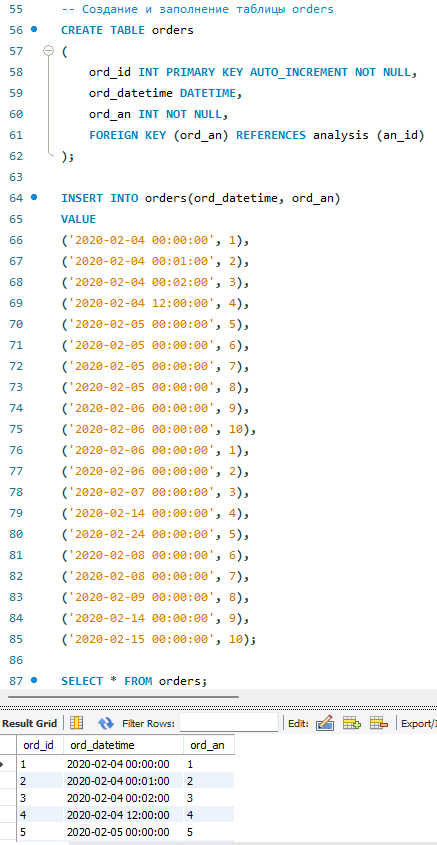
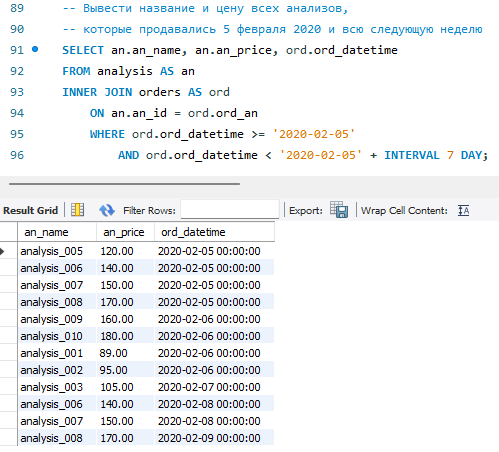
****

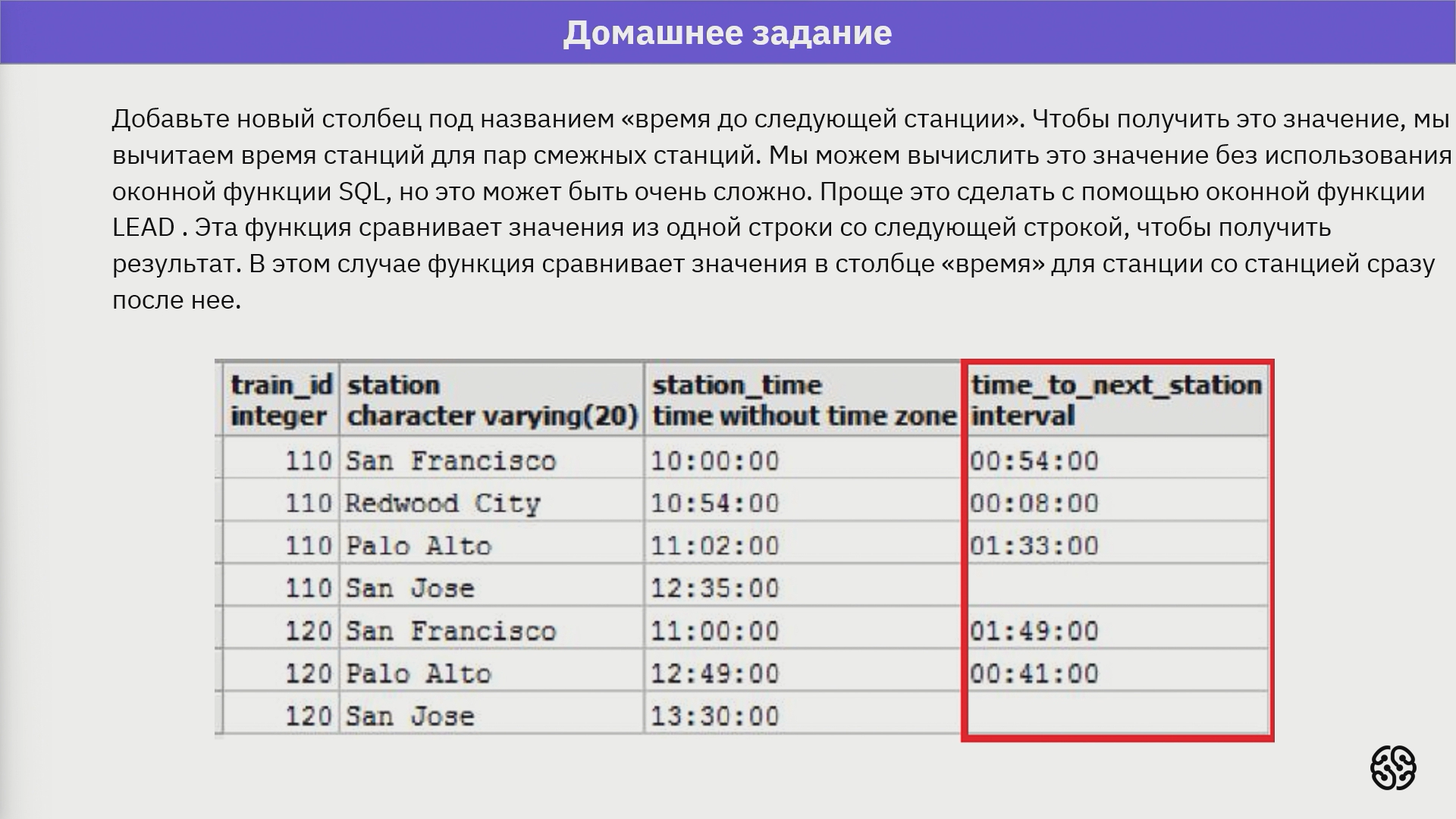
****

1. ***Ход выполнения задания 2:***

*Вывести название и цену всех анализов, которые продавались 5 февраля 2020 и всю следующую неделю.*

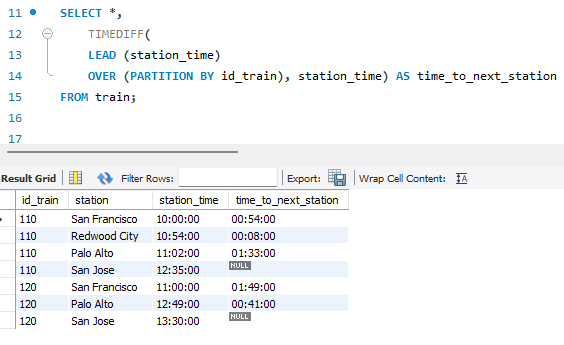
***1)  2) ***

***3)  4) ***

****

1. ***Ход выполнения задания 3:***

*Добавьте новый столбец под названием «время до следующей станции».*

******

**ЧАСТЬ II.**

1. ***Ход выполнения задания 0: Запрос на создание начальной таблицы, выполнять задания на основе этих данных***

*Вы можете воспользоваться заготовкой fiddle* [*https://dbfiddle.uk/oQi939Ap*](https://dbfiddle.uk/oQi939Ap)

*Или использовать код ниже для создания таблиц*

*USE seminar\_5;*

*-- Запрос на создание начальной таблицы, выполнять задания на основе этих данных*

*-- Создание таблицы Employees*

*DROP TABLE IF EXISTS Employees;*

*CREATE TABLE Employees (*

*employee\_id INT PRIMARY KEY,*

*employee\_name VARCHAR(255)*

*);*

*-- Создание таблицы Orders*

*DROP TABLE IF EXISTS Orders;*

*CREATE TABLE Orders (*

*order\_id INT PRIMARY KEY,*

*employee\_id INT,*

*customer\_id INT,*

*order\_date DATE,*

*total\_amount DECIMAL(10, 2),*

*FOREIGN KEY (employee\_id) REFERENCES Employees(employee\_id)*

*);*

*-- Создание таблицы Customers*

*DROP TABLE IF EXISTS Customers;*

*CREATE TABLE Customers (*

*customer\_id INT PRIMARY KEY,*

*customer\_name VARCHAR(255)*

*);*

*-- Наполнение таблиц данными*

*INSERT INTO Employees (employee\_id, employee\_name)*

*VALUES*

*(1, 'Alice Johnson'), (2, 'Bob Brown'), (3, 'Charlie Davis'),*

*(4, 'David Wilson'), (5, 'Emily Clark');*

*INSERT INTO Customers (customer\_id, customer\_name)*

*VALUES*

*(1, 'John Doe'), (2, 'Jane Smith'), (3, 'Alice Johnson'),*

*(4, 'Bob Brown'), (5, 'Charlie Davis'), (6, 'Emily Clark'),*

*(7, 'David Wilson'), (8, 'Laura Adams'), (9, 'Michael Thompson'),*

*(10, 'Sarah Parker');*

*INSERT INTO Orders (order\_id, employee\_id, customer\_id, order\_date, total\_amount)*

*VALUES*

*(1, 1, 1, '2023-01-15', 550.00),*

*(2, 2, 2, '2023-02-20', 600.00),*

*(3, 3, 3, '2023-03-10', 300.00),*

*(4, 4, 4, '2023-04-25', 750.00),*

*(5, 5, 5, '2023-05-18', 450.00),*

*(6, 1, 6, '2023-06-12', 500.00),*

*(7, 2, 7, '2023-07-21', 700.00),*

*(8, 3, 8, '2023-08-30', 620.00),*

*(9, 4, 9, '2023-09-14', 480.00),*

*(10, 5, 10, '2023-10-03', 510.00);*

1. ***Ход выполнения задания 1: Получение заказов по сотруднику***

*Создайте хранимую процедуру GetEmployeeOrders, которая принимает идентификатор сотрудника в качестве параметра и возвращает все заказы, обработанные этим сотрудником.*

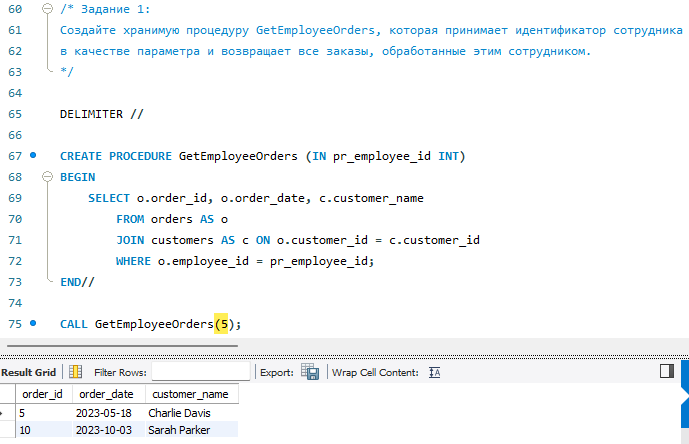
*В результате запроса будут столбцы:*

* *order\_id: идентификатор заказа*
* *order\_date: дата заказа*
* *customer\_name: имя клиента*

*Подсказка: Используйте оператор CREATE PROCEDURE для создания хранимой*

*процедуры. Для объединения таблиц используйте оператор JOIN. Параметр*

*передается в процедуру через IN.*

**

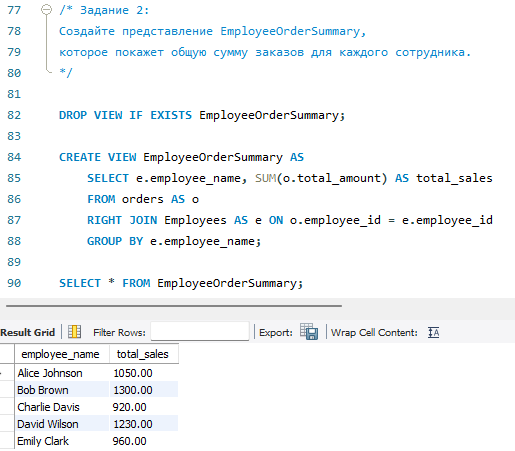
1. ***Ход выполнения задания 2: Создание представления для суммы заказов по сотрудникам***

*Создайте представление EmployeeOrderSummary, которое покажет общую сумму заказов для каждого сотрудника.*

*В результате запроса будут столбцы:*

* *employee\_name: имя сотрудника*
* *total\_sales: общая сумма заказов*

*Подсказка: Используйте оператор CREATE VIEW для создания представления. Для агрегирования данных используйте функцию SUM и группировку GROUP BY.*

**

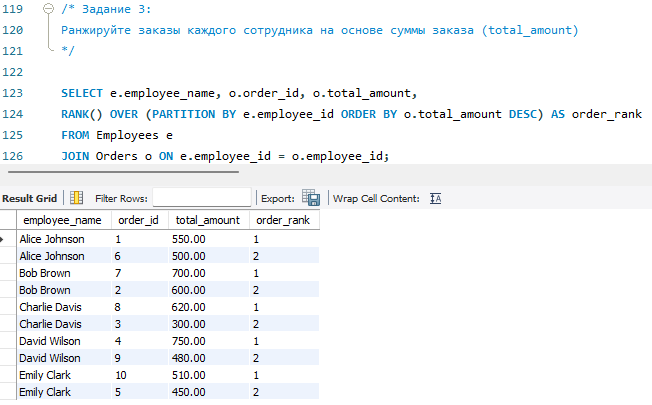
1. ***Ход выполнения задания 3: Использование оконных функций для ранжирования заказов***

*Ранжируйте заказы каждого сотрудника на основе суммы заказа (total\_amount).*

*В результате запроса будут столбцы:*

* *employee\_name: имя сотрудника*
* *order\_id: идентификатор заказа*
* *total\_amount: сумма заказа*
* *order\_rank: ранг заказа по сумме для каждого сотрудника*

*Подсказка: Используйте функцию RANK() с PARTITION BY для разделения данных по сотрудникам и ORDER BY для сортировки по сумме заказа. Объедините таблицы с помощью JOIN.*

**

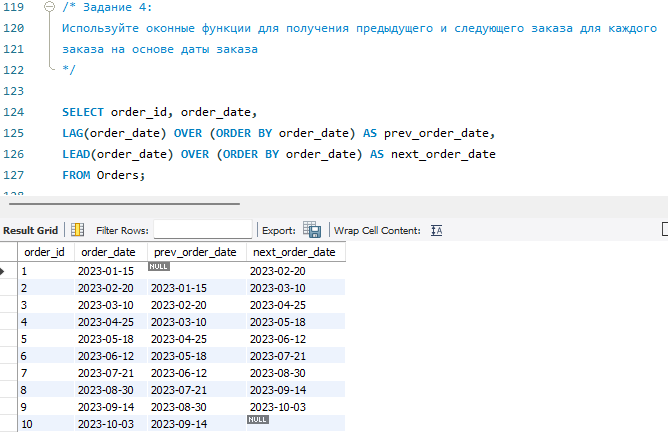
1. ***Ход выполнения задания 4: Получение предыдущего и следующего заказа для каждого заказа***

*Используйте оконные функции для получения предыдущего и следующего заказа для каждого заказа на основе даты заказа.*

*В результате запроса будут столбцы:*

* *order\_id: идентификатор заказа*
* *order\_date: дата заказа*
* *prev\_order\_date: дата предыдущего заказа*
* *next\_order\_date: дата следующего заказа*

*Подсказка: Используйте оконные функции LAG и LEAD для получения предыдущего и следующего значений. Примените ORDER BY для упорядочивания заказов по дате.*

**

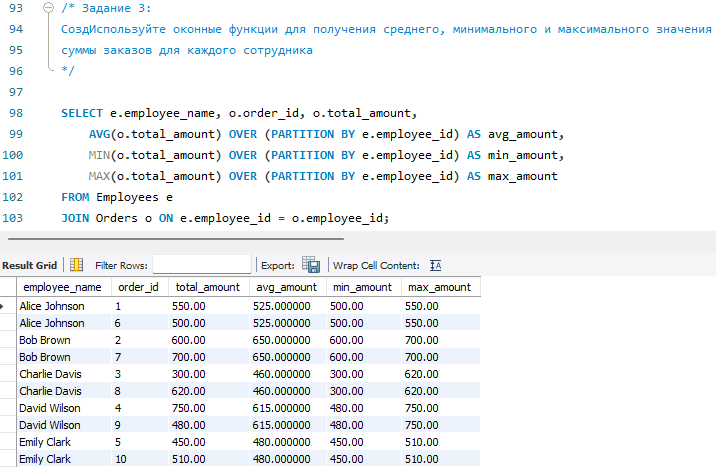
1. ***Ход выполнения задания 5: Аналитические функции для среднего, минимального и максимального значения заказов***

*Используйте оконные функции для получения среднего, минимального и максимального значения суммы заказов для каждого сотрудника.*

*В результате запроса будут столбцы:*

* *employee\_name: имя сотрудника*
* *order\_id: идентификатор заказа*
* *total\_amount: сумма заказа*
* *avg\_amount: средняя сумма заказа для каждого сотрудника*
* *min\_amount: минимальная сумма заказа для каждого сотрудника*
* *max\_amount: максимальная сумма заказа для каждого сотрудника*

*Подсказка: Используйте оконные функции AVG, MIN и MAX с PARTITION BY для разделения данных по сотрудникам. Объедините таблицы с помощью JOIN.*

**

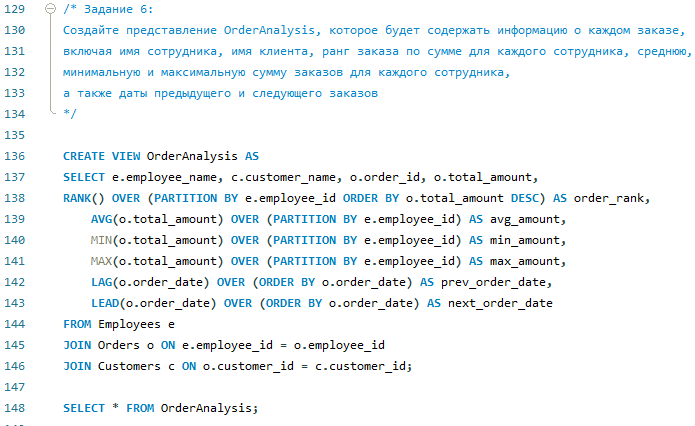
1. ***Ход выполнения задания 6: Комплексный анализ заказов***

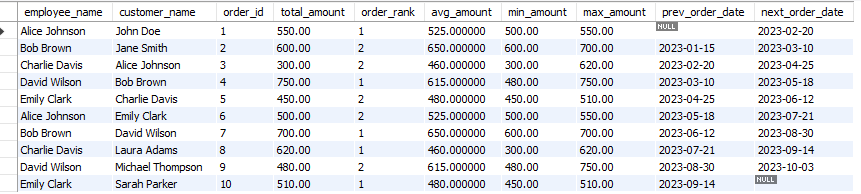
*Создайте представление OrderAnalysis, которое будет содержать информацию о каждом заказе, включая имя сотрудника, имя клиента, ранг заказа по сумме для каждого сотрудника, среднюю, минимальную и максимальную сумму заказов для каждого сотрудника, а также даты предыдущего и следующего заказов.*

*В результате запроса будут столбцы:*

* *employee\_name: имя сотрудника*
* *customer\_name: имя клиента*
* *order\_id: идентификатор заказа*
* *total\_amount: сумма заказа*
* *order\_rank: ранг заказа по сумме для каждого сотрудника*
* *avg\_amount: средняя сумма заказа для каждого сотрудника*
* *min\_amount: минимальная сумма заказа для каждого сотрудника*
* *max\_amount: максимальная сумма заказа для каждого сотрудника*
* *prev\_order\_date: дата предыдущего заказа*
* *next\_order\_date: дата следующего заказа*

*Подсказка: Создайте представление с помощью CREATE VIEW. Используйте функции RANK, AVG, MIN, MAX, LAG и LEAD для анализа данных. Объедините данные из таблиц Employees, Orders и Customers с помощью JOIN.*

**

**