

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«Магнитогорский государственный технический университет  
им. Г. И. Носова»**  
(ФГБОУ ВО «МГТУ им. Г.И. Носова»)

Кафедра бизнес-информатики и информационных технологий

**Отчет**  
по учебной – технологической (проектно-технологической) практике

Исполнитель: Елшанский В.Д. студент 2 курса, группы АПИб-22-2

Руководитель практики: \_\_\_\_\_ Шаранова Регина Ришатовна, ассистент кафедры БИиИТ

Руководитель практики  
от Профильной организации: \_\_\_\_\_ Ошурков Вячеслав Александрович, руководитель направления  
консалтинга службы бизнес-анализа и консалтинга ЗАО «КонсОМ СКС»

Отчет защищен «16» июля 2024 г. с оценкой \_\_\_\_\_  
(оценка) (подпись)

Магнитогорск, 2024

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Магнитогорский государственный технический университет  
им. Г. И. Носова  
(ФГБОУ ВО «МГТУ им. Г.И. Носова»)

Кафедра бизнес-информатики и информационных технологий

**РАБОЧИЙ ПЛАН-ГРАФИК**

по учебной – технологической (проектно-технологической) практике

в период с 03.07.2024 по 16.07.2024

Обучающемуся Елшанскому Владимиру Дмитриевичу группы АПИ6-22-2

№	Этапы практики по выполнению программы практики и индивидуального задания	Срок исполнения
1	Установочное собрание по организации практики	03.07.2024
2	Выполнение индивидуального задания	04.07.2024 – 09.07.2024
3	Оформление отчёта по учебно-эксплуатационной практике	10.07.2024 – 14.07.2024
4	Защита отчёта по учебно-эксплуатационной практике	16.07.2024

Руководитель практики от МГТУ им. Г.И. Носова

Ассистент кафедры БИиИТ

\_\_\_\_\_  
(подпись)

/ Шаранова Р.Р./

Руководитель практики от Профильной организации

Руководитель направления  
консалтинга службы бизнес-  
анализа и консалтинга ЗАО  
«КонсОМ СКС»

\_\_\_\_\_  
(подпись)

/ Ошурков В.А./



## Дневник прохождения практики

Студента Елшанский В.Д.

Группы АПИб-22-2

курса 2

Направления 09.03.03 Прикладная информатика (разработка компьютерных игр и AR/VR-приложений (виртуальной/дополненной реальности))

Сроки практики: с 03.07.2024 по 16.07.2024 г.

Дата	Краткое содержание выполненной работы	Отметка о выполнении
1. Организационно-подготовительный этап		
03.07.2024	1.1 Участие в установочном собрании по организации практики. Получение индивидуального задания.	
10.07.2024	1.2 Вводный инструктаж представителя закрытого акционерного общества «КонсОМ СКС» обучающимся по правилам ТБ, производственной и противопожарной безопасности	
2. Основной этап		
04.07.2024	Установка PostgreSQL сервера и программного продукта DBeaver	
05.07.2024 – 09.07.2024	Описание SQL-запросов	
3. Отчетный этап		
11.07.2024 – 16.07.2024	Подготовка и сдача отчета по практике	

Руководитель практики  
от МГТУ им. Г.И. Носова

\_\_\_\_\_

(подпись)

Шаранова Р.Р.

## СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	6
1 ОРГАНИЗАЦИОННО-ПОДГОТОВИТЕЛЬНЫЙ ЭТАП .....	7
1.1 УЧАСТИЕ В УСТАНОВОЧНОМ СОБРАНИИ ПО ОРГАНИЗАЦИИ ПРАКТИКИ. ПОЛУЧЕНИЕ ИНДИВИДУАЛЬНОГО ЗАДАНИЕ .....	7
1.2 ВВОДНЫЙ ИНСТРУКТАЖ ПРЕДСТАВИТЕЛЯ ЗАКРЫТОГО АКЦИОНЕРНОГО ОБЩЕСТВА «КОНСОМ СКС» ОБУЧАЮЩИМСЯ.....	7
2 ОСНОВНОЙ ЭТАП .....	8
2.1 УСТАНОВКА PostgreSQL.....	8
2.2 УСТАНОВКА И ПОДГОТОВКА ПРОГРАММНОГО ПРОДУКТА DBEAVER.....	9
3 ОПИСАНИЕ SQL-ЗАПРОСОВ .....	14
3.1 ВЫПОЛНЕНИЕ «ЗАДАНИЯ 1» .....	14
3.2 ВЫПОЛНЕНИЕ «ЗАДАНИЯ 2» .....	14
3.3 ВЫПОЛНЕНИЕ «ЗАДАНИЯ 3» .....	15
3.4 ВЫПОЛНЕНИЕ «ЗАДАНИЯ 4» .....	16
4 ОТЧЕТНЫЙ ЭТАП.....	17
ЗАКЛЮЧЕНИЕ.....	18
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	19
ПРИЛОЖЕНИЯ .....	20
ПРИЛОЖЕНИЕ А РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ «ЗАДАНИЯ 1» .....	20
ПРИЛОЖЕНИЕ Б РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ «ЗАДАНИЯ 2» .....	24
ПРИЛОЖЕНИЕ В РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ «ЗАДАНИЯ 3» .....	29
ПРИЛОЖЕНИЕ Г РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ «ЗАДАНИЯ 4» .....	33
ПРИЛОЖЕНИЕ Д СПРАВКА О ПРОВЕРКЕ В СИСТЕМЕ «АНТИПЛАГИАТ.РУ».....	37

## **ВВЕДЕНИЕ**

В соответствии с учебным планом учебная - технологическая (проектно-технологическая) практика была пройдена в ЗАО «КонсОМ СКС» с 3 июля 2024 г. по 16 июля 2024 г. (14 дней). Учебная - технологическая (проектно-технологическая) практика проводится с целью закрепления полученных студентом теоретических знаний и приобретения практических навыков, необходимых для самостоятельной работы в IT-компании. Актуальность прохождения практики обуславливается тем, что написание дипломной работы основывается на материалах прохождения учебной практики, в связи с этим прохождение практики облегчает написание выпускной квалификационной работы.

Целями учебной - технологическая (проектно-технологическая) практики являются:

1. закрепление и углубление теоретических знаний, полученных при изучении дисциплин учебного плана;
2. приобретение и развитие необходимых практических умений и навыков в соответствии с требованиями к уровню подготовки выпускника;
3. приобретение опыта самостоятельной профессиональной деятельности.

Задачами учебной - технологическая(проектно-технологическая) практики являются:

1. изучение нотации SQL;
2. разработка SQL-запросов для выполнения индивидуального задания.

## **1 ОРГАНИЗАЦИОННО-ПОДГОТОВИТЕЛЬНЫЙ ЭТАП**

### **1.1 Участие в установочном собрании по организации практики. Получение индивидуального задания**

В первый день практики состоялось установочное собрание, на котором были обсуждены ключевые аспекты организации и проведения практики. На собрании присутствовали куратор практики от предприятия и студенты.

Основные темы, освещённые на собрании:

1. обзор структуры компании и её основных подразделений;
2. цели и задачи технологической практики;
3. правила внутреннего распорядка и требования к безопасности;
4. основные этапы практики и ожидаемые результаты.

После установочного собрания каждому студенту было выдано индивидуальное задание, включающее установку PostgreSQL сервера и программного продукта DBeaver, описание шагов и практических заданий.

### **1.2 Вводный инструктаж представителя закрытого акционерного общества «КонсОМ СКС» обучающимся**

В первый день практики состоялся вводный инструктаж, проведённый представителем «КонсОМ СКС» для обучающихся. На инструктаже были рассмотрены следующие ключевые аспекты:

1. Общие сведения о компании. Представитель компании дал обзор деятельности «КонсОМ СКС», её основных направлений работы и достижений.
2. Организационная структура. были представлены основные подразделения компании и их функции, а также краткое описание ролей и обязанностей сотрудников.
3. Корпоративная культура. Обсуждены ценности компании, стандарты поведения, ожидания от сотрудников и принципы взаимодействия внутри коллектива.
4. Правила внутреннего распорядка. Описаны основные правила и нормы поведения, график работы, требования к посещаемости и оформлению документов.
5. Меры безопасности: Ознакомление с основными требованиями по охране труда и технике безопасности, а также противопожарной безопасности, актуальными для работы в компании.

## 2 ОСНОВНОЙ ЭТАП

### 2.1 Установка PostgreSQL

После получения индивидуального задания, необходимо было установить на свой персональный компьютер PostgreSQL сервер, представляющее из себя систему управления базами данных. С помощью PostgreSQL можно создавать, хранить базы данных и работать с данными с помощью запросов на языке SQL.

Для установки PostgreSQL необходимо заранее скачать с сайта <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads> подходящую для персонального компьютера версию. После завершения скачивания, запускаем и проходим установку программного продукта, выбираем путь, где будет установлено ПО. Также для дальнейшей работы необходимо придумать пароль и порт, который установлен по умолчанию как 5432. Также необходимо выбрать необходимые для работы версию и драйвера. Эти данные представлены на рисунке 1 и 2 соответственно.

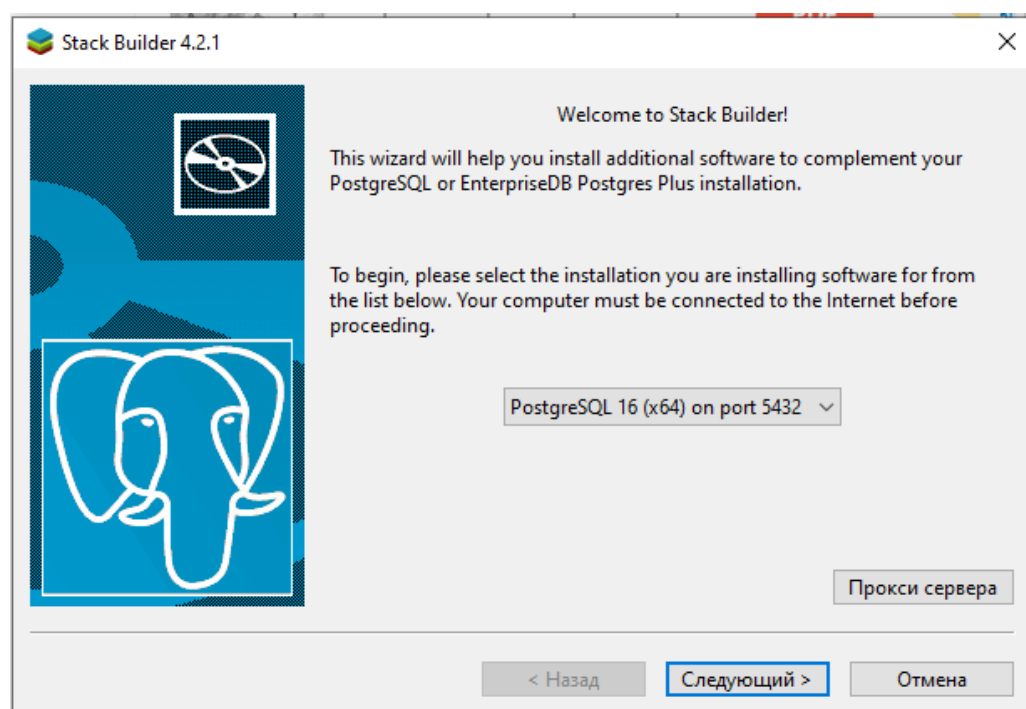


Рисунок 1 – Выбор версии для установки PostgreSQL



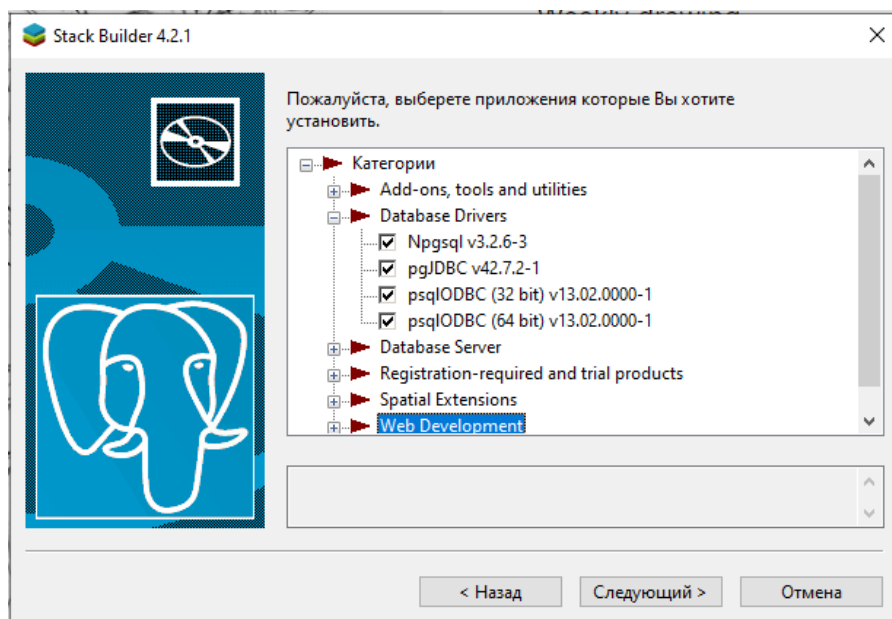


Рисунок 2 – Выбор драйверов для PostgreSQL

После необходимо выбрать директорию для загрузки выбранных пакетов и после их установки перезагрузить персональный компьютер. Проверить работу сервера можно в диспетчере задач, как представлено на рисунке 3.

> pg_ctl - starts/stops/restarts the...	0%	0,9 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	2,1 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	1,8 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	1,7 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	2,8 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	2,0 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	1,8 МБ	0 МБ/с	0 Мбит/с
PostgreSQL Server	0%	2,9 МБ	0 МБ/с	0 Мбит/с

Рисунок 3 – Проверка работы PostgreSQL

После установки и проверки работы сервера можно начинать установку программного продукта DBeaver.

## 2.2 Установка и подготовка программного продукта DBeaver

Для предоставления результатов исследования нужно установить нужное программное обеспечение, в нашем случае это DBeaver.

DBeaver – это универсальный и многофункциональный клиентом для работы с базами данных, который поддерживает все популярные реляционные и нереляционные базы данных, такие

как MySQL, PostgreSQL, SQL Server, DB2 и многие другие. Он предоставляет удобный графический интерфейс, мощные возможности для написания SQL-запросов, инструменты для визуализации данных и доступен как в платной версии (DBeaver PRO), так и в бесплатной версии (DBeaver Community). Это отличный выбор для специалистов, работающих с базами данных, из-за широкого функционала, удобства использования и доступности.

Для начала заходим на сайт [dbeaver.io](https://dbeaver.io) и нажимаем кнопку «Download» как выделенно на рисунке 4.

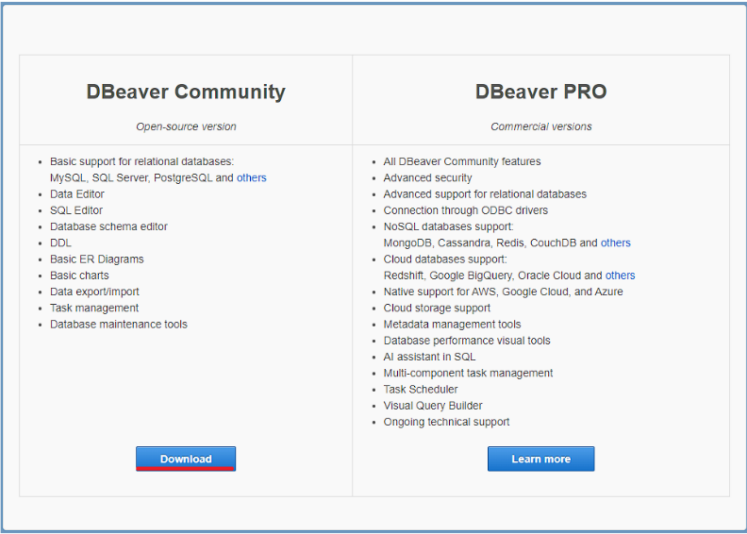


Рисунок 4 – Скачивание ПО (шаг 1)

Мы попадаем на сайт следующую вкладку, где нам надо нажать «Windows Installer» и скачать пакет ПО, как выделенно на рисунке 5.

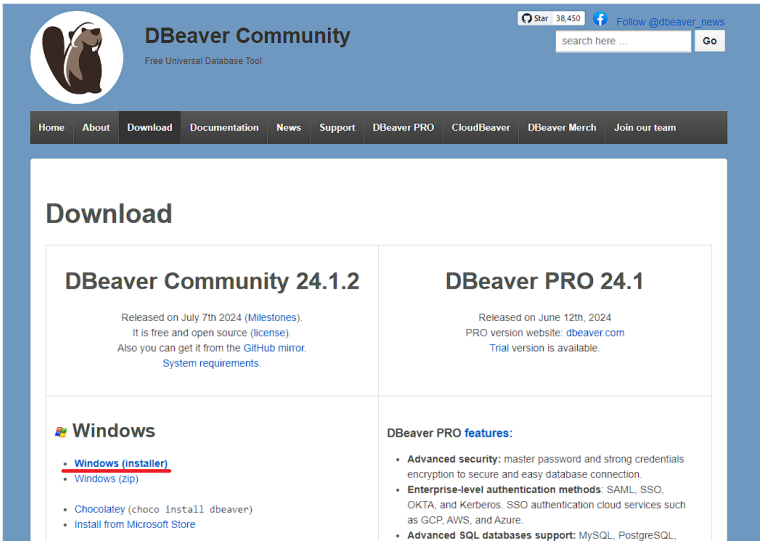


Рисунок 5- Скачивание ПО (шаг 2)

После завершения скачивания, запускаем и проходим установку программного продукта, выбираем путь, где будет установлено ПО, как показано на рисунке 6.

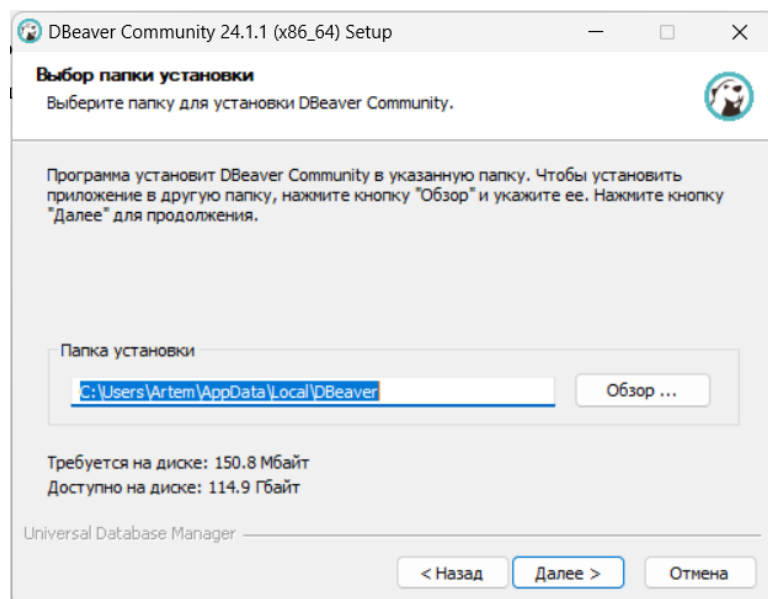


Рисунок 6 – Выбор репозитория ПО

После установки программного продукта необходимо его подключить к серверу SQL. Для этого необходимо в меню «База данных» выбрать «Новое соединение». После нажатия открывается меню, содержащее типы соединения для базы данных. После выполнения ручного поиска для добавления PostgreSQL, как показано на рисунке 7, отображенного цифрой «1». Когда тип соединения выбран, нажимается кнопка «Далее». Эти данные обозначены цифрой «2» на рисунке 7.

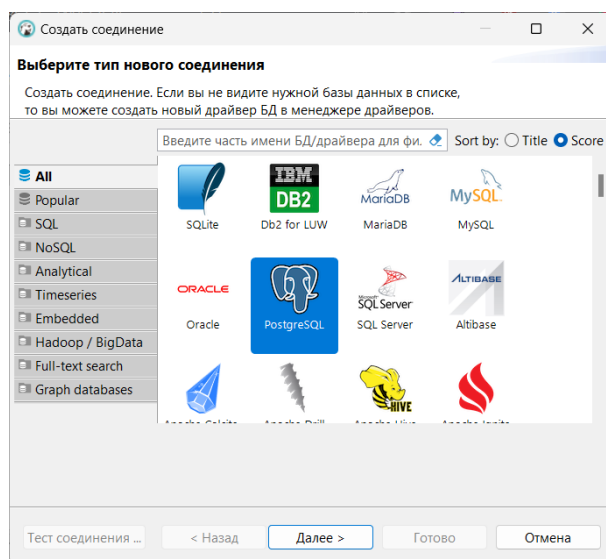


Рисунок 7 – Добавление PostgreSQL

После необходимо ввести хост, порт, логин и пароль пользователя, которые были введены при установке PostgreSQL. Данные с обозначены цифрами «1», «2» и «3» обведены на рисунке 8. А также при в разделе «Свойства драйвера» при необходимости скачать предлагаемые драйвера.

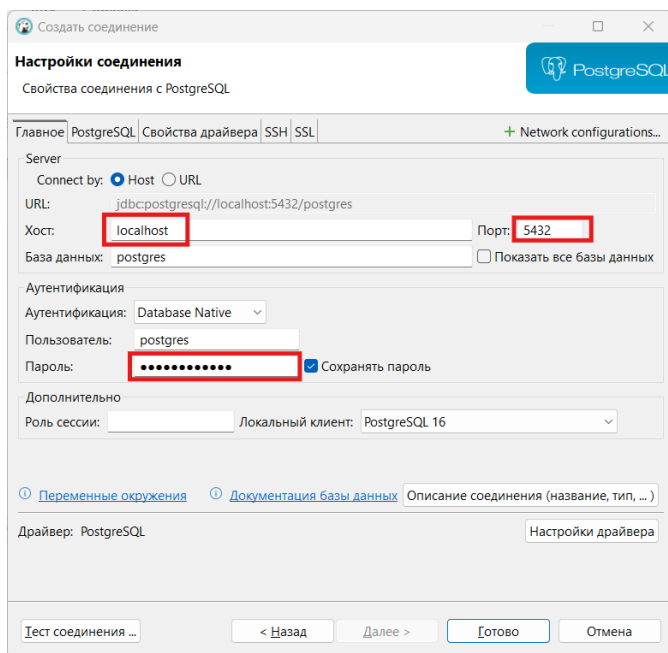


Рисунок 8 – настройка PostgreSQL

После подключения сервера необходимо восстановить базу данных из файла «dvd-rental.backup». Чтобы выбрать необходимую базу данных, нужно раскрыть сервер PostgreSQL в области «Проекты», затем раскрыть папку «General» и выбрать нужную базу данных «postgres» правой кнопкой мыши. В контекстном меню выбираем необходимое действие «Восстановить» для подключения к базе данных, как показано на рисунке 9.

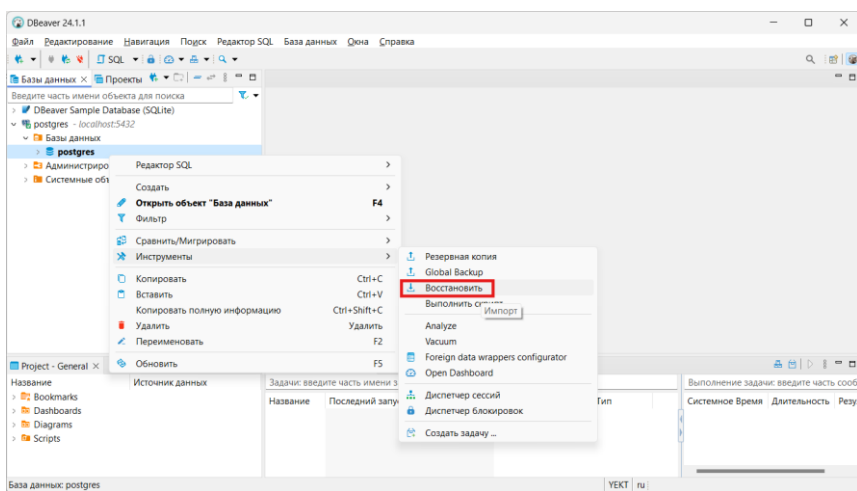


Рисунок 9 – Восстановление базы данных

После необходимо ввести выбрать нужный файл и нажать кнопку «Старт», как показано на рисунке 10. В итоге после восстановления базы данных, она открывается в виде диаграммы, где видны таблицы и их связи, как представлено на рисунке 11.

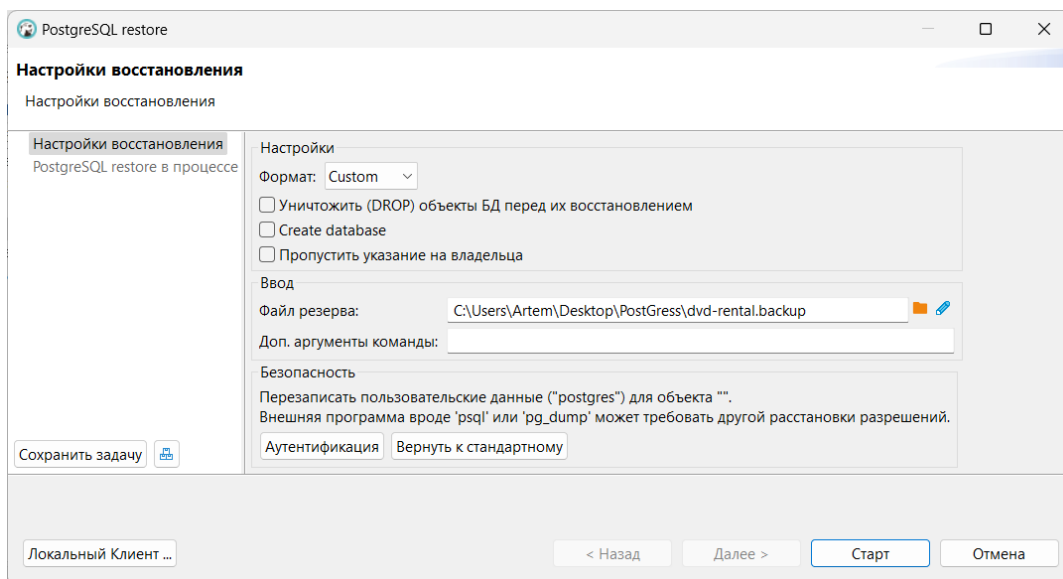


Рисунок 10 – Выбор файла «dvd-rental.backup»

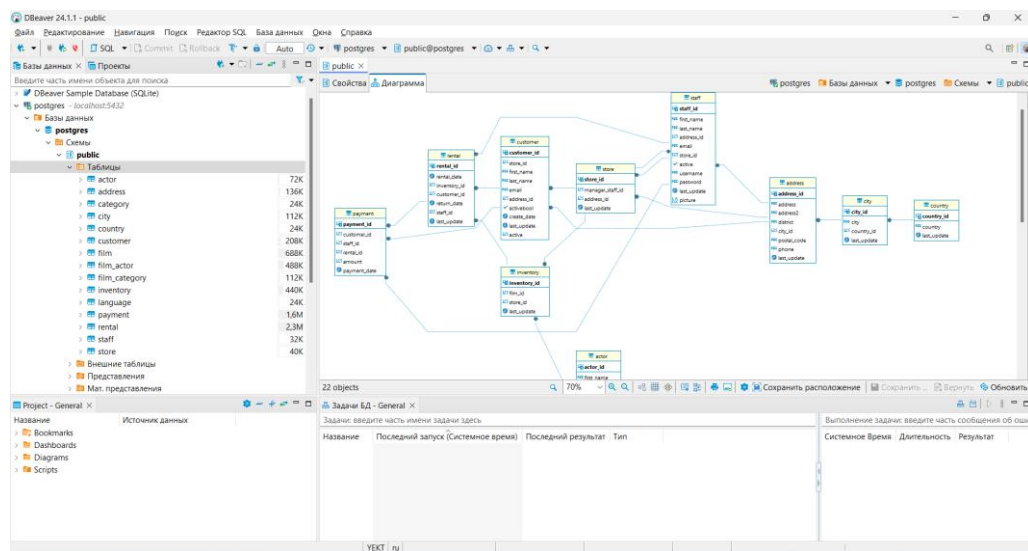


Рисунок 11 – Восстановленная диаграмма

После установки программного обеспечения можно приступить к выполнению.

## **3 ОПИСАНИЕ SQL-ЗАПРОСОВ**

### **3.1 Выполнение «Задания 1»**

В рамках этого задания требовалось продемонстрировать способность составлять базовые SQL-запросы, присваивать имена столбцам, использовать возможности фильтрации и сортировки записей в таблицах с применением ключевых операторов SQL. Также необходимо было продемонстрировать умение выполнять преобразование текстовых, числовых и временных значений с помощью соответствующих функций SQL для работы со строками, датами и числами.

Для выполнения заданий в этом блоке необходимо было использовать следующие команды и операторы:

- select – позволяет взять данные из таблицы;
- distinct – указывает на то, что должны быть выведены только уникальные значения;
- as – используется для переименования столбца или таблицы с псевдонимами и создания вычисляемых колонок;
- from – указывает на то из какой таблицы берутся данные;
- order by – сортирует числа по возрастанию и строки по алфавиту;
- where – позволяет задать условия для отбора данных;
- like – позволяет задать условие для нахождения «похожих» данных;
- not – позволяет инвертировать условие;
- between – используется для выбора данных в заданном диапазоне;
- and и or – логический оператор, позволяющий совмещать условия;
- order by desc – используется для сортировки по убыванию;
- limit – задает максимальное количество строк, которые нужно вывести;
- concat\_ws – соединят поля столбцов с разделителем;
- char\_length – позволяет подсчитать количества символов в строке;
- cast – используется для приведения типов данных;
- lower – приводит символы в строке к нижнему регистру;
- split\_part – позволяет разделить строку на части с помощью разделителя и вернуть нужную часть.

Все SQL-запросы, а также результаты из выполнения представлены в приложении А.

### **3.2 Выполнение «Задания 2»**

В процессе выполнения этих заданий мы стремились продемонстрировать нашу компетенцию в области функций агрегации и группировки строк, а также умения фильтровать

данные после группировки. Ключевой момент заключался в том, чтобы продемонстрировать навыки работы с методами соединения таблиц через различные типы операций JOIN.

Для выполнения заданий в этом блоке необходимо было использовать следующие команды и операторы:

- select, distinct, as, from, order by, order by desc, limit, where – изученные в предыдущем задании;

- left join – предназначен для соединения таблиц и вывода результирующей таблицы, в которой данные полностью пересекаются по условию и дополняются записями из левой таблицы, даже если они не соответствуют условию;

- right join – предназначен для соединения таблиц и вывода результирующей таблицы, в которой данные полностью пересекаются по условию и дополняются записями из правой таблицы, даже если они не соответствуют условию;

- inner join – предназначен для соединения таблиц и вывода результирующей таблицы, в которой данные полностью пересекаются по условию;

- cross join – реализует операцию декартова произведения в реляционной алгебре;

- round – округляет значения до определенного знака;

- sum() – возвращает общую сумму числового столбца;

- min() – возвращает минимальное значение в столбце;

- max() – возвращает максимальное значение в столбце;

- group by – группирует строки с одинаковым значением в сводные строки;

- having – используется для фильтрации результата с оператором group by;

- count() – возвращает количество строк, соответствующее заданному критерию;

- case when условие then *результат1* else *результат* end – возвращает условие, которое выполняется.

Все SQL-запросы, а также результаты из выполнения представлены в приложении Б.

### 3.3 Выполнение «Задания 3»

В рамках выполнения данной группы заданий была поставлена цель продемонстрировать знания в работе с оконными функциями.

Оконные функции – это особый тип функций, которые вычисляют значение для каждой строки в результирующем наборе, основанном на группе связанных строк.

Для выполнения заданий в этом блоке необходимо было использовать следующие функции:

- select, distinct, as, from, order by, order by desc, limit, where, inner join и др. – изученные в предыдущем задании;
- row\_number() – присваивает уникальный номер каждой строке в определенном наборе;
- dense\_rank() – функция возвращает ранг каждой строки. Но в отличие от функции rank(), она для одинаковых значений возвращает ранг, не пропуская следующий;
- over – определяет окно или определяемый пользователем набор строк внутри результирующего набора запроса;
- partition by – разделяет результирующий набор запроса на секции;
- unbounded preceding – указывает, что окно начинается с первой строки группы;
- lag() – обращается к данным из предыдущей строки окна;
- last\_value() – выводит последнее значение в определенном наборе;
- unbounded following – указать, что окно заканчивается на последней строке группы.
- date() – возвращает только дату без времени
- with – временный результирующий набор данных, к которому можно обращаться в последующих запросах.

Все SQL-запросы, а также результаты из выполнения представлены в приложении В.

### **3.4 Выполнение «Задания 4»**

Целью данного задания было показать умение использовать различные инструменты SQL для эффективного анализа данных. Задание охватывало работу с массивами, составлением подзапросов, использованием табличных выражений, созданием материализованных представлений и оценкой производительности запросов.

Для выполнения заданий в этом блоке необходимо было использовать следующие команды и операторы:

- select, distinct, as, from, order by, order by desc, limit, where, inner join и др. – изученные в предыдущем задании;
- any – возвращает значение true, если любое из значений подчиненного запроса удовлетворяет условию;
- all – возвращает значение true, если все значения подчиненного запроса удовлетворяют условию.

Все SQL-запросы, а также результаты из выполнения представлены в приложении Г.



#### **4 ОТЧЕТНЫЙ ЭТАП**

На заключительном этапе нашей практики мы провели важный момент – презентацию и защиту нашего отчета о проделанной работе.

Первым шагом на этом пути стала подготовка структурированного отчета. Это было не просто сборкой информации, но и тщательным анализом полученных результатов. Мы уделяли особое внимание форматированию и логике представления данных, чтобы сделать отчет понятным и доступным как для нас, так и для наших слушателей.

После того как отчет был готов, следующим шагом стало его представление перед научным руководителем. Эта стадия была особенно важной, поскольку она давала нам возможность получить предварительные замечания и предложения по улучшению нашего проекта до окончательной защиты. Мы тщательно подготавливались к этому мероприятию, репетировали презентацию, а также готовились ответить на возможные вопросы, которые могут возникнуть во время обсуждения.

Защита отчета проходила в формате диалога между нами и нашим научным руководителем. Это было отличной возможностью услышать мнение эксперта о нашей работе, узнать его взгляды на полученные результаты и получить конкретные рекомендации по их улучшению.

В результате обсуждения мы получили ценную обратную связь и рекомендации, которые помогут нам улучшить нашу работу.

## **ЗАКЛЮЧЕНИЕ**

В ходе практики были изучены нотации SQL и разработаны SQL-запросы для выполнения индивидуального задания. Одним из основных инструментов, используемых в рамках практики, стал PostgreSQL - мощная открытая СУБД, обеспечивающая высокую производительность и надежность. Для удобства работы с этой платформой мы также использовали программное обеспечение DBeaver, которое предлагает широкий спектр функций для управления базами данных, включая графическое создание схем, редактирование запросов и мониторинг производительности. Было настроено подключение к серверу SQL и восстановлена база данных из файла «dvd-rental.backup». Прохождение практики дало нам возможность значительно расширить набор практических навыков, связанных с SQL, PostgreSQL и DBeaver.

Несмотря на то, что в начале практики мы столкнулись с трудностями, связанными с отсутствием непосредственного опыта работы с базами данных, мы смогли преодолеть эти препятствия.

Также стоит отметить, что мы улучшили наши навыки сбора информации и коммуникации, впервые начали проходить практику в предприятии. Трудности проявились в начале, после получения задания, так как до практики с базами данных мы ещё не работали.

Представленные запросы были успешно защищены на итоговой презентации. Проведение публичной защиты наших выводов дало нам возможность убедиться в глубоком понимании изученного материала и уверенно выступить перед аудиторией.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. SQL JOIN: типы и примеры // TProger. [Электронный ресурс]. Режим доступа: <https://tproger.ru/articles/sql-join> (Дата обращения: 11.07.2024).
2. SQL JOIN // SchoolsW3. [Электронный ресурс]. Режим доступа: [https://www.schoolsw3.com/sql/sql\\_join.php](https://www.schoolsw3.com/sql/sql_join.php) (Дата обращения: 11.07.2024).
3. Учимся применять оконные функции // This is data. [Электронный ресурс]. Режим доступа: <https://thisisdata.ru/blog/uchimsya-primenyat-okonnyye-funktsii/> (Дата обращения: 12.07.2024).
4. Справочник SQL // Code.mu. [Электронный ресурс]. Режим доступа: <https://code.mu/ru/sql/manual/> (Дата обращения: 12.07.2024).
5. Подробная Шпаргалка SQL на 2023 год // Uproger.com [Электронный ресурс]. Режим доступа: <https://uproger.com/shpargalki-sql-2023/> (Дата обращения: 11.07.2024).
6. Оконные функции SQL простым языком с примерами // Хабр. [Электронный ресурс]. Режим доступа: <https://habr.com/ru/articles/664000/> (Дата обращения: 11.07.2024).
7. Оконные функции SQL // SQL Academy. [Электронный ресурс]. Режим доступа: <https://sql-academy.org/ru/guide/windows-functions> (Дата обращения: 11.07.2024).
8. Обобщённое табличное выражение, оператор WITH // SQL Academy. [Электронный ресурс]. Режим доступа: <https://sql-academy.org/ru/guide/operator-with> (Дата обращения: 10.07.2024).
9. Массивы и Списки в SQL Server // Interface. [Электронный ресурс]. Режим доступа: <https://www.interface.ru/home.asp?artId=22765> (Дата обращения: 11.07.2024).
10. Используем все возможности индексов в PostgreSQL // Хабр. [Электронный ресурс]. Режим доступа: <https://habr.com/ru/companies/vk/articles/453046/> (Дата обращения: 11.07.2024).

## ПРИЛОЖЕНИЯ

### Приложение А Результат выполнения «Задания 1»

--1. Выведите уникальные названия городов из таблицы городов

```
select distinct city from city c order by city;
```

city 1 X

select distinct city from city c order by city Введите SQL выражение чтобы отфильтровать результаты

Таблица	city
1	A Corua (La Corua)
2	Abha
3	Abu Dhabi
4	Acua
5	Adana
6	Addis Abeba
7	Aden
8	Adoni

Рисунок А. 1 – Результат выполнения задания 1

/\*Задание 2. Доработайте запрос из предыдущего задания,  
\* чтобы запрос выводил только те города, названия которых начинаются на "L" и заканчиваются на "a",  
\* и названия не содержат пробелов.\*/

```
select distinct city from city c where city like 'L%a' and city not like '% %' order by city;
```

city 1 X

select distinct city from city c where city like 'L%a' and city not like '% %' Введите SQL выражение чтобы отфильтровать результаты

Таблица	city
1	Лиераја
2	Lima
3	Loja
4	Luzinia

Рисунок А. 2 – Результат выполнения задания 2

/\*Задание 3. Получите из таблицы платежей за прокат фильмов информацию по платежам,  
\* которые выполнялись в промежуток с 17 июня 2005 года по 19 июня 2005 года  
\* включительно и стоимость которых превышает 1.00. Платежи нужно отсортировать по дате платежа\*/

```
select * from payment p where payment_date between '2005-06-17' and '2005-06-20' and amount > 1.00  
order by payment_date ;  
select * from payment p;
```

payment 1 X

select \* from payment p Введите SQL выражение чтобы отфильтровать результаты

Таблица	payment_id	customer_id	staff_id	rental_id	amount	payment_date
1	1	1	1	76	2,99	2005-05-25 11:30:37.000
2	2	2	1	573	0,99	2005-05-28 10:35:23.000
3	3	3	1	1 185	5,99	2005-06-15 00:54:12.000
4	4	4	2	1 422	0,99	2005-06-15 18:02:53.000
5	5	5	2	1 476	9,99	2005-06-15 21:08:46.000
6	6	6	1	1 725	4,99	2005-06-16 15:18:57.000
7	7	7	1	2 308	4,99	2005-06-18 08:41:48.000
8	8	8	2	2 363	0,99	2005-06-18 13:33:59.000
9	9	9	1	3 284	3,99	2005-06-21 06:24:45.000
10	10	10	2	4 526	5,99	2005-07-08 03:17:05.000

Рисунок А. 3 – Результат выполнения задания 3

--Задание 4. Выведите информацию о 10-ти последних платежах за прокат фильмов

```
select * from payment p order by payment_date desc limit 10;
```

payment 1 X

select \* from payment p order by payment\_date desc limit 10; Введите SQL выражение чтобы отфильтровать результаты

	123 payment_id	123 customer_id	123 staff_id	123 rental_id	123 amount	payment_date
1	630	23	2	15 532	2,99	2006-02-14 15:16:03.000
2	302	11	1	11 646	0,99	2006-02-14 15:16:03.000
3	600	22	1	12 222	4,99	2006-02-14 15:16:03.000
4	417	15	2	13 968	0	2006-02-14 15:16:03.000
5	253	9	1	15 813	4,99	2006-02-14 15:16:03.000
6	145	5	2	13 209	0,99	2006-02-14 15:16:03.000
7	416	15	1	13 798	3,98	2006-02-14 15:16:03.000
8	578	21	1	14 933	2,99	2006-02-14 15:16:03.000
9	385	14	1	13 780	4,99	2006-02-14 15:16:03.000
10	781	28	2	12 938	2,99	2006-02-14 15:16:03.000

Рисунок А. 4 – Результат выполнения задания 4

/\* Задание 5. Выведите следующую информацию по покупателям:

- Фамилия и имя (в одной колонке через пробел)
- Электронная почта
- Длину значения поля email
- Дату последнего обновления записи о покупателе (без времени)

Каждой колонке задайте наименование на русском языке.

```
*/
select
concat_ws(' ',first_name,last_name) as Имя_покупателя,
email as Электронная_почта,
char_length(email) as Длина_почты,
cast(last_update as date) as Дата_последнего_обновления_записи
from customer c ;
```

customer 1 X

select concat\_ws(' ',first\_name,last\_name) as Имя\_покупателя; Введите SQL выражение чтобы отфильтровать результаты

	ABC Имя_покупателя	ABC Электронная_почта	123 Длина_почты	Дата_последнего_обновления_записи
1	MARY SMITH	MARY.SMITH@sakilacustomer.org	29	2006-02-15
2	PATRICIA JOHNSON	PATRICIA.JOHNSON@sakilacustomer.org	35	2006-02-15
3	LINDA WILLIAMS	LINDA.WILLIAMS@sakilacustomer.org	33	2006-02-15
4	BARBARA JONES	BARBARA.JONES@sakilacustomer.org	32	2006-02-15
5	ELIZABETH BROWN	ELIZABETH.BROWN@sakilacustomer.org	34	2006-02-15
6	JENNIFER DAVIS	JENNIFER.DAVIS@sakilacustomer.org	33	2006-02-15
7	MARIA MILLER	MARIA.MILLER@sakilacustomer.org	31	2006-02-15
8	SUSAN WILSON	SUSAN.WILSON@sakilacustomer.org	31	2006-02-15
9	MARGARET MOORE	MARGARET.MOORE@sakilacustomer.org	33	2006-02-15
10	DOROTHY TAYLOR	DOROTHY.TAYLOR@sakilacustomer.org	33	2006-02-15
11	LISA ANDERSON	LISA.ANDERSON@sakilacustomer.org	32	2006-02-15
12	NANCY THOMAS	NANCY.THOMAS@sakilacustomer.org	31	2006-02-15

Рисунок А. 5 – Результат выполнения задания 5

/\*Задание 6. Выведите одним запросом только активных покупателей, имена которых KELLY или WILLIE.  
\* Все буквы в фамилии и имени из верхнего регистра должны быть переведены в нижний регистр\*/

```
select
lower(concat_ws(' ',first_name,last_name))
from customer c
where (first_name = 'KELLY' or first_name = 'WILLIE') and activebool = true;
```

Результат 1

select lower(concat\_ws(' ',first\_name,last\_name)) from ci

Таблица	1	2	3	4
1	kelly torres			
2	willie howell			
3	willie markham			
4	kelly knott			

Рисунок А. 6 – Результат выполнения задания 6

/\*Задание 7. Выведите одним запросом информацию о фильмах, у которых рейтинг “R” и стоимость аренды указана от 0.00 до 3.00 включительно, а также фильмы с рейтингом “PG-13” и стоимостью аренды больше или равной 4.00\*/

```
select * from film f where (rating = 'R' and rental_rate between 0.00 and 3.00)
or (rating = 'PG-13' and rental_rate >= 4.00);
```

film 1

select \* from film f where (rating = 'R' and rental\_rate be

Таблица	123 film_id	ABC title	ABC description	123 release_year	123 language_id	123 original_language_id	123 rent
1	7	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who must Discover a Butler i	2 006	1	[NULL]	
2	17	ALONE TRIP	A Fast-Paced Character Study of a Composer And a Dog who must Our	2 006	1	[NULL]	
3	23	ANACONDA CONFESSIONS	A Lacklustre Display of a Dentist And a Dentist who must Fight a Girl i	2 006	1	[NULL]	
4	24	ANALYZE HOOSIERS	A Thoughtful Display of a Explorer And a Pastry Chef who must Overcc	2 006	1	[NULL]	
5	28	ANTHEM LUKE	A Touching Panorama of a Waitress And a Woman who must Outrace	2 006	1	[NULL]	
6	30	ANYTHING SAVANNAH	A Epic Story of a Pastry Chef And a Woman who must Chase a Feminis	2 006	1	[NULL]	
7	40	ARMY FLINTSTONES	A Boring Saga of a Database Administrator And a Womanizer who mus	2 006	1	[NULL]	
8	44	ATTACKS HATE	A Fast-Paced Panorama of a Technical Writer And a Mad Scientist who	2 006	1	[NULL]	
9	45	ATTRACTION NEWTON	A Astounding Panorama of a Composer And a Frisbee who must Reacl	2 006	1	[NULL]	
10	48	BACKLASH UNDEFEATED	A Stunning Character Study of a Mad Scientist And a Mad Cow who m	2 006	1	[NULL]	
11	49	BADMAN DAWNI	A Emotional Panorama of a Pioneer And a Commercials who must Ensu	2 006	1	[NULL]	

Рисунок А. 7 – Результат выполнения задания 7

or (rating = 'PG-13' and rental\_rate >= 4.00);

--Задание 8. Получите информацию о трёх фильмах с самым длинным описанием фильма

```
select * from film f order by char_length(description) desc limit 3 ;
```

film 1

select \* from film f order by char\_length(description) des

Таблица	123 film_id	ABC title	ABC description	123 release_year	123 language_id	123 original_language_id	123 rent
1	217	DAZED PUNK	A Action-Packed Story of a Pioneer And a Technical Writer who must D	2 006	1	[NULL]	
2	116	CANDIDATE PERDITION	A Brilliant Epistle of a Composer And a Database Administrator who m	2 006	1	[NULL]	
3	274	EGG IGBY	A Beautiful Documentary of a Boat And a Sumo Wrestler who must Suc	2 006	1	[NULL]	

Рисунок А. 8 – Результат выполнения задания 8

/\*Задание 9. Выведите Email каждого покупателя, разделив значение Email на 2 отдельных колонки:  
 • в первой колонке должно быть значение, указанное до @,  
 • во второй колонке должно быть значение, указанное после @  
 \*/

```

select
split_part(email, '@', 1) as Пользователь,
split_part(email, '@', 2) as Домен
from customer c ;

```

Результат 1

	Пользователь	Домен
1	MARY.SMITH	sakilacustomer.org
2	PATRICIA.JOHNSON	sakilacustomer.org
3	LINDA.WILLIAMS	sakilacustomer.org
4	BARBARA.JONES	sakilacustomer.org
5	ELIZABETH.BROWN	sakilacustomer.org
6	JENNIFER.DAVIS	sakilacustomer.org
7	MARIA.MILLER	sakilacustomer.org
8	SUSAN.WILSON	sakilacustomer.org
9	MARGARET.MOORE	sakilacustomer.org
10	DOROTHY.TAYLOR	sakilacustomer.org
11	LISA.ANDERSON	sakilacustomer.org

Рисунок А. 9 – Результат выполнения задания 9

/\*Задание 10. Доработайте запрос из предыдущего задания, скорректируйте значения в новых колонках:  
 \* первая буква должна быть заглавной, остальные строчными.\*  
 \*/

```

select initcap(split_part(email, '@', 1)) as Пользователь,
initcap(split_part(email, '@', 2)) as Домен
from customer c ; --Тут слова после точек тоже с большой пишутся
select
upper(substring(split_part(email, '@', 1) from 1 for 1)) ||
lower(substring(split_part(email, '@', 1) from 2)) as Пользователь,
upper(substring(split_part(email, '@', 2) from 1 for 1)) ||
lower(substring(split_part(email, '@', 2) from 2)) as Домен
from customer c ; --Тут правильно но громоздко

```

Результат 1

	Пользователь	Домен
1	MARY SMITH	sakilacustomer.org
2	PATRICIA JOHNSON	sakilacustomer.org
3	LINDA WILLIAMS	sakilacustomer.org
4	BARBARA JONES	sakilacustomer.org
5	ELIZABETH BROWN	sakilacustomer.org
6	JENNIFER DAVIS	sakilacustomer.org
7	MARIA MILLER	sakilacustomer.org
8	SUSAN WILSON	sakilacustomer.org
9	MARGARET MOORE	sakilacustomer.org
10	DOROTHY TAYLOR	sakilacustomer.org
11	LISA ANDERSON	sakilacustomer.org
12	NANCY THOMAS	sakilacustomer.org
13	KAREN JACKSON	sakilacustomer.org
14	BETTY WHITE	sakilacustomer.org
15	HELEN HARRIS	sakilacustomer.org

Рисунок А. 10 – Результат выполнения задания 10

## Приложение Б

### Результат выполнения «Задания 2»

--Задание 1. Выведите для каждого покупателя его адрес, город и страну проживания.

```

select c3.country as Страна, c2.city as Город, a.address as Адрес, c.first_name || ' ' || c.last_name as Имя_покупателя from customer c
inner join address a on a.address_id = c.address_id
inner join city c2 on c2.city_id = a.city_id
inner join country c3 on c3.country_id = c2.country_id ;

```

country(+)	Страна	Город	Адрес	Имя_покупателя
1	Japan	Sasebo	1913 Hanoi Way	MARY SMITH
2	United States	San Bernardino	1121 Loja Avenue	PATRICIA JOHNSON
3	Greece	Athenai	692 Joliet Street	LINDA WILLIAMS
4	Myanmar	Myingyan	1566 Inegl Manor	BARBARA JONES
5	Taiwan	Nantou	53 Idfu Parkway	ELIZABETH BROWN
6	United States	Laredo	1795 Santiago de Compostela Way	JENNIFER DAVIS
7	Yugoslavia	Kragujevac	900 Santiago de Compostela Parkway	MARIA MILLER
8	New Zealand	Hamilton	478 Joliet Way	SUSAN WILSON
9	Oman	Masqat	613 Korolev Drive	MARGARET MOORE
10	Iran	Esfahan	1531 Sal Drive	DOROTHY TAYLOR
11	Japan	Sagamihara	1542 Tarlac Parkway	LISA ANDERSON
12	India	Yamuna Naqar	808 Bhopal Manor	NANCY THOMAS

Рисунок Б. 11 – Результат выполнения задания 1

/\*Задание 2. С помощью SQL-запроса посчитайте для каждого магазина количество его покупателей.

- Доработайте запрос и выведите только те магазины, у которых количество покупателей больше 300. Для решения используйте фильтрацию по сгруппированным строкам с функцией агрегации.
- Доработайте запрос, добавив в него информацию о городе магазина, фамилии и имени продавца, который работает в нём.

```

select store_id as Магазин, count(*) as Количество_покупателей from customer c
group by c.store_id;

select store_id as Магазин, count(*) as Количество_покупателей from customer c
group by c.store_id
having count(*) > 300;

select c2.city as Город_магазина, s2.first_name as Имя_продавца, s2.last_name
as Фамилия_продавца, s.store_id as Магазин, count(*) as Количество_покупателей from customer c
inner join store s on s.store_id = c.store_id
inner join address a on a.address_id = s.address_id
inner join city c2 on c2.city_id = a.city_id
inner join staff s2 on s2.store_id = s.store_id
group by c2.city, s2.first_name, s2.last_name, s.store_id
having count(*) > 300;

select c2.city as Город_магазина, concat_ws(' ', s2.first_name, s2.last_name)
as Имя_Фамилия_продавца, s.store_id as Магазин, count(*) as Количество_покупателей from customer c
inner join store s on s.store_id = c.store_id
inner join address a on a.address_id = s.address_id
inner join city c2 on c2.city_id = a.city_id
inner join staff s2 on s2.store_id = s.store_id
group by c2.city, s2.first_name, s2.last_name, s.store_id
having count(*) > 300;

```

city(+)	Город_магазина	Имя_Фамилия_продавца	Магазин	Количество_покупателей
1	Lethbridge	Mike Hillier	1	326

Рисунок Б. 12 – Результат выполнения задания 2



-- Задание 3. Выведите топ-5 покупателей, которые взяли в аренду за всё время  
-- наибольшее количество фильмов

```
select customer_id as Покупатель, count(payment_id) as Количество_аренд from payment p
group by customer_id
order by count(payment_id) desc limit 5;
```

payment 1 X

select customer\_id as Покупатель, count(payment\_id) as Ki | Введите SQL выражение чтобы отфильтровать результаты

Таблица	123 Покупатель	123 Количество_аренд
1	148	46
2	526	45
3	236	42
4	144	42
5	75	41

Рисунок Б. 13 – Результат выполнения задания 3

/\*Задание 4. Посчитайте для каждого покупателя 4 аналитических показателя:

- количество взятых в аренду фильмов;
- общую стоимость платежей за аренду всех фильмов (значение округлите до целого числа);
- минимальное значение платежа за аренду фильма;
- максимальное значение платежа за аренду фильма.

\*/

```
select
customer_id as Покупатель,
count(rental_id) as Количество_взятых_фильмов,
round(sum(amount)) as Общая_стоимость_платежей,
min(amount) as Минимальная_стоимость_платежа,
max(amount) as Максимальная_стоимость_платежа
from payment p
group by customer_id
order by customer_id;
```

payment 1 X

select customer\_id as Покупатель, count(rental\_id) as Ki | Введите SQL выражение чтобы отфильтровать результаты

Таблица	123 Покупатель	123 Количество_взятых_фильмов	123 Общая_стоимость_платежей	123 Минимальная_стоимость_платежа	123 Максимальная_стоимость_платежа
1	1	32	119	0,99	9,99
2	2	27	129	0,99	10,99
3	3	26	136	0,99	10,99
4	4	22	82	0,99	8,99
5	5	38	145	0,99	9,99
6	6	28	94	0,99	7,99
7	7	33	152	0,99	8,99
8	8	24	93	0,99	9,99
9	9	23	90	0,99	7,99
10	10	25	100	0,99	8,99
11	11	24	107	0,99	9,99
12	12	28	104	0,99	10,99
13	13	27	132	0,99	11,99

Рисунок Б. 4 – Результат выполнения задания 4

/\*Задание 5. Используя данные из таблицы городов, составьте одним запросом всевозможные пары городов так, чтобы в результате не было пар с одинаковыми названиями городов. Для решения необходимо использовать декартово произведение\*/

```
select c.city as Первый_город, c2.city as Второй_город from city c
cross join city c2
where c.city != c2.city;

select c.city, c2.city from city c, city c2 where c.city < c2.city;
```

city 1 X

select c.city, c2.city from city c, city c2 where c.city < c2.city

Таблица	city	city
1	A Corua (La Corua)	Abha
2	A Corua (La Corua)	Abu Dhabi
3	A Corua (La Corua)	Acua
4	A Corua (La Corua)	Adana
5	A Corua (La Corua)	Addis Abeba
6	A Corua (La Corua)	Aden
7	A Corua (La Corua)	Adoni
8	A Corua (La Corua)	Ahmadnagar
9	A Corua (La Corua)	Akishima
10	A Corua (La Corua)	Akron
11	A Corua (La Corua)	al-Ayn

Рисунок Б. 5 – Результат выполнения задания 5

/\*Задание 6. Используя данные из таблицы rental о дате выдачи фильма в аренду (поле rental\_date) и дате возврата (поле return\_date), вычислите для каждого покупателя среднее количество дней, за которые он возвращает фильмы\*/

```
select customer_id as Покупатель, round(avg(date(return_date)-date(rental_date)),2)
as Средняя_разница_в_днях from rental r
group by customer_id
order by customer_id;
```

rental 1 X

select customer\_id as Покупатель, round(avg(date(return\_date)-date(rental\_date)),2) as Средняя\_разница\_в\_днях from rental r group by customer\_id order by customer\_id;

Таблица	Покупатель	Средняя_разница_в_днях
1	1	4,47
2	2	5,52
3	3	5,88
4	4	3,86
5	5	5,03
6	6	5,43
7	7	5,61
8	8	4,54
9	9	4,55
10	10	6
11	11	5,7
12	12	5,29
13	13	5,04
14	14	5,11

Рисунок Б. 6 – Результат выполнения задания 6

/\*Задание 7. Посчитайте для каждого фильма, сколько раз его брали в аренду, а также общую стоимость аренды фильма за всё время\*/

```

select f.title as Название_фильма, count(r.rental_id)
as Сколько_раз_брали_в_аренду, sum(coalesce(p.amount, 0))
as Общая_стоимость_аренды from rental r
left join inventory i on i.inventory_id = r.inventory_id
right join film f on f.film_id = i.film_id
left join payment p on p.rental_id = r.rental_id
group by f.film_id
order by f.film_id;

```

film 1

select f.title as Название\_фильма, count(r.rental\_id) | Введите SQL выражение чтобы отфильтровать результаты

	ABC Название_фильма	123 Сколько_раз_брали_в_аренду	123 Общая_стоимость_аренды
1	ACADEMY DINOSAUR	23	36,77
2	ACE GOLDFINGER	7	52,93
3	ADAPTATION HOLES	12	37,88
4	AFFAIR PREJUDICE	23	91,77
5	AFRICAN EGG	12	51,88
6	AGENT TRUMAN	21	126,79
7	AIRPLANE SIERRA	15	82,85
8	AIRPORT POLLOCK	18	102,82
9	ALABAMA DEVIL	12	71,88
10	ALADDIN CALENDAR	23	131,77
11	ALAMO VIDEOTAPE	24	35,76
12	ALASKA PHANTOM	26	44,74
13	ALI FOREVER	9	54,91
14	ALICE FANTASIA	0	0
15	ALIEN CENTER	22	90,78
16	ALLEY EVOLUTION	14	52,86

Рисунок Б. 7 – Результат выполнения задания 7

--Задание 8. Доработайте запрос из предыдущего задания и выведите с помощью него фильмы, которые ни разу не брали в аренду.

```

select f.title as Название_фильма, count(r.rental_id) as Сколько_раз_брали_в_аренду
from rental r
left join inventory i on i.inventory_id = r.inventory_id
right join film f on f.film_id = i.film_id
group by f.film_id
having count(r.rental_id) = 0
order by f.film_id;

```

film 1

select f.title as Название\_фильма, count(r.rental\_id) | Введите SQL выражение чтобы отфильтровать результаты

	ABC Название_фильма	123 Сколько_раз_брали_в_аренду
1	ALICE FANTASIA	0
2	APOLLO TEEN	0
3	ARGONAUTS TOWN	0
4	ARK RIDGEMONT	0
5	ARSENIC INDEPENDENCE	0
6	BOONDOCK BALLROOM	0
7	BUTCH PANTHER	0
8	CATCH AMISTAD	0
9	CHINATOWN GLADIATOR	0
10	CHOCOLATE DUCK	0
11	COMMANDMENTS EXPRESS	0
12	CROSSING DIVORCE	0
13	CROWDS TELEMAR	0

Рисунок Б. 8 – Результат выполнения задания 8

```

/*Задание 9. Посчитайте количество продаж, выполненных каждым продавцом.
 * Добавьте вычисляемую колонку «Премия». Если количество продаж превышает 7 300,
 * то значение в колонке будет «Да»,
 * иначе должно быть значение «Нет».*
select
s.staff_id as Продавец,
count(p.payment_id) as Количество_продаж,
(case when count(p.payment_id)>7300 then 'Да' else 'Нет' end) as Премия
from payment p
inner join staff s on s.staff_id = p.staff_id
group by s.staff_id
order by s.staff_id;

```

staff 1	×	select s.staff_id as Продавец, count(p.payment_id) as			Введите SQL выражение чтобы отфильтровать результаты
123	Продавец	123	Количество_продаж	As	Премия
1	1	8 057	Да		
2	2	7 992	Да		

Рисунок Б. 9 – Результат выполнения задания 9

## Приложение В

### Результат выполнения «Задания 3»

/\*Задание 1. Сделайте запрос к таблице payment и с помощью оконных функций добавьте вычисляемые колонки \* согласно условиям:

- Пронумеруйте все платежи от 1 до N по дате
- Пронумеруйте платежи для каждого покупателя, сортировка платежей должна быть по дате
- Посчитайте нарастающим итогом сумму всех платежей для каждого покупателя, сортировка должна быть по дате платежа, а затем по сумме платежа от наименьшей к большей
- Пронумеруйте платежи для каждого покупателя по стоимости платежа от наибольших к меньшим так, чтобы платежи с одинаковым значением имели одинаковое значение номера.

Можно составить на каждый пункт отдельный SQL-запрос, а можно объединить все колонки в одном запросе.

```
*/
select
payment_date as Дата_платежа,
row_number () over (order by payment_date) as Номер_платежа,
customer_id as Айди_покупателя,
row_number() over(partition by customer_id) as Номер_платежа_покупателя,
amount as Сумма_платежа,
dense_rank() over (partition by customer_id order by amount desc) as Номер_платежа_по_стоимости,
sum(amount) over(partition by customer_id order by payment_date , amount rows unbounded preceding) as Нарастающий
from payment p
order by payment_date, amount;
```

payment 1 X

select payment\_date as Дата\_платежа, row\_number

	Дата_платежа	123 Номер_платежа	123 Айди_покупателя	123 Номер_платежа_покупателя	123 Сумма_платежа	123 Номер_платежа_по_стоимости	123 Нарас
1	2005-05-24 22:53:30.000	1	130	1	2,99	5	
2	2005-05-24 22:54:33.000	2	459	1	2,99	8	
3	2005-05-24 23:03:39.000	3	408	1	3,99	5	
4	2005-05-24 23:04:41.000	4	333	1	4,99	3	
5	2005-05-24 23:05:21.000	5	222	1	6,99	2	
6	2005-05-24 23:08:07.000	6	549	1	0,99	6	
7	2005-05-24 23:11:53.000	7	269	1	1,99	8	
8	2005-05-24 23:31:46.000	8	239	1	4,99	5	

Рисунок В. 1 – Результат выполнения задания 1

--Задание 2. С помощью оконной функции выведите для каждого покупателя стоимость платежа и --стоимость платежа из предыдущей строки со значением по умолчанию 0.0 с сортировкой по дате.

```
select
customer_id as Айди_покупателя,
amount as Платеж,
lag(amount,1,0.0) over (partition by customer_id order by payment_date) as Предыдущий_платеж
from payment p
order by customer_id, payment_date;
```

payment 1 X

select customer\_id as Айди\_покупателя, amount as

	123 Айди_покупателя	123 Платеж	123 Предыдущий_платеж
1	1	2,99	0
2	1	0,99	2,99
3	1	5,99	0,99
4	1	0,99	5,99
5	1	9,99	0,99
6	1	4,99	9,99
7	1	4,99	4,99
8	1	0,99	4,99
9	1	3,99	0,99
10	1	5,99	3,99
11	1	5,99	5,99
12	1	4,99	5,99
13	1	4,99	4,99

Рисунок В. 2 – Результат выполнения задания 2

-- Задание 3. С помощью оконной функции определите, на сколько каждый следующий платеж покупателя больше или меньше текущего.

```

select
customer_id as Айди_покупателя,
amount as Размер_платежа_покупателя,
amount - lead (amount,1,0.0) over (partition by customer_id) as Разница_разм_след_и_текущ_платежа
from payment p;

```

payment 1 X

select customer\_id as Айди\_покупателя, amount as Введите SQL выражение чтобы отфильтровать результаты

	123 Айди_покупателя	123 Размер_платежа_покупателя	123 Разница_разм_след_и_текущ_платежа
1	1	2,99	2
2	1	0,99	-5
3	1	5,99	5
4	1	0,99	-9
5	1	9,99	5
6	1	4,99	0
7	1	4,99	4
8	1	0,99	-3
9	1	3,99	-2
10	1	5,99	0
11	1	5,99	1
12	1	4,99	0
13	1	4,99	-3

Рисунок В. 3 – Результат выполнения задания 3

--Задание 4. С помощью оконной функции для каждого покупателя выведите данные о его последней оплате аренды.

```

select
*
from payment p
where payment_date in ((select last_value(payment_date) over (partition by customer_id) from payment p2));
with RankedPayments as (
select
*,
row_number () over (partition by customer_id order by payment_date desc) as rn
from payment p)
select
*
from RankedPayments
where rn = 1;

```

payment 1 X

with RankedPayments as (select \*, row\_number () over Введите SQL выражение чтобы отфильтровать результаты

	123 payment_id	123 customer_id	123 staff_id	123 rental_id	123 amount	payment_date	123 rn
1	32	1	1	15 315	5,99	2005-08-22 20:03:46.000	1
2	59	2	2	15 907	4,99	2005-08-23 17:39:35.000	1
3	85	3	1	15 619	2,99	2005-08-23 07:10:14.000	1
4	107	4	2	15 635	1,99	2005-08-23 07:43:00.000	1
5	145	5	2	13 209	0,99	2006-02-14 15:16:03.000	1
6	173	6	2	15 603	0,99	2005-08-23 06:41:32.000	1
7	206	7	1	14 222	5,99	2005-08-21 04:49:48.000	1
8	230	8	1	15 805	4,99	2005-08-23 14:31:19.000	1
9	253	9	1	15 813	4,99	2006-02-14 15:16:03.000	1

Рисунок В. 4 – Результат выполнения задания 4

\*/Задание 5. С помощью оконной функции выведите для каждого сотрудника сумму продаж за август 2005 года с нарастающим итогом по каждому сотруднику и по каждой дате продажи (без учёта времени) с сортировкой по дате\*/

```

select
staff_id as Айди_работника,
date(payment_date) as Дата,
sum (amount) over (partition by staff_id order by payment_date, amount rows unbounded preceding)
as Нарастающий_итог_по_сотруднику,
sum (amount) over (partition by staff_id order by payment_date,
amount rows between unbounded preceding and unbounded following) as Сумма_продаж_за_август_2005,
sum (amount) over (partition by staff_id, date(payment_date) order by date(payment_date),
amount rows unbounded preceding ) as Нарастающий_итог_по_дате,
sum (amount) over (partition by staff_id, date(payment_date)) as Сумма_продаж_по_дням
from payment p
where payment_date between '2005-08-01' and '2005-09-01'
order by staff_id, date(payment_date);

```

	123 Айди_работника	Дата	123 Нарастающий_итог_по_сотруднику	123 Сумма_продаж_за_август_2005	123 Нарастающий_итог_по_дате	123 Сумма_продаж_по_дням
1	1	2005-08-01	1 135,2	11 853,65	0,99	
2	1	2005-08-01	767,12	11 853,65	1,98	
3	1	2005-08-01	66,84	11 853,65	2,97	
4	1	2005-08-01	786,08	11 853,65	3,96	
5	1	2005-08-01	224,48	11 853,65	4,95	
6	1	2005-08-01	77,81	11 853,65	5,94	
7	1	2005-08-01	1 039,46	11 853,65	6,93	
8	1	2005-08-01	161,63	11 853,65	7,92	
9	1	2005-08-01	854,92	11 853,65	8,91	
10	1	2005-08-01	860,9	11 853,65	9,9	
11	1	2005-08-01	1 027,49	11 853,65	10,89	
12	1	2005-08-01	1 012,52	11 853,65	11,88	

Рисунок В. 5 – Результат выполнения задания 5

\*/Задание 6. 20 августа 2005 года в магазинах проходила акция: покупатель каждого сотого платежа получал дополнительную скидку на следующую аренду. С помощью оконной функции выведите всех покупателей, которые в день проведения акции получили скидку\*/

```

with NumberedCustomers as (
select
*,
row_number () over (order by date(payment_date)) as numbered
from payment p
where date(payment_date) = '2005-08-20'
)
select
date(payment_date) as Дата,
numbered as Номер_покупателя,
customer_id as Айди_покупателя
from NumberedCustomers
where numbered % 100 = 0;

```

	Дата	123 Номер_покупателя	123 Айди_покупателя
1	2005-08-20	100	87
2	2005-08-20	200	189
3	2005-08-20	300	273
4	2005-08-20	400	372
5	2005-08-20	500	458
6	2005-08-20	600	569

Рисунок В. 6 – Результат выполнения задания 6

\*/Задание 7. Для каждой страны определите и выведите одним SQL-запросом покупателей, которые попадают под условия:

- покупатель, арендовавший наибольшее количество фильмов;
- покупатель, арендовавший фильмов на самую большую сумму;
- покупатель, который последним арендовал фильм.

```

with RankedCustomers as (
    select
        c.customer_id,
        c.first_name,
        c.last_name,
        c3.country,
        count(p.payment_id) over (partition by c.customer_id) as num_rentals,
        sum(p.amount) over (partition by c.customer_id) as total_amount,
        row_number() over (partition by c.customer_id order by p.payment_date desc) as rental_order,
        p.payment_date
    from customer c
    inner join payment p on p.customer_id = c.customer_id
    inner join rental r on r.rental_id = p.rental_id
    inner join address a on a.address_id = c.address_id
    inner join city c2 on c2.city_id = a.city_id
    inner join country c3 on c3.country_id = c2.country_id
),
MaxRentals as (
    select
        country,
        MAX(num_rentals) as max_num_rentals,
        MAX(total_amount) as max_total_amount,
        max(payment_date) as max_payment_date
    from RankedCustomers
    group by country
)
select
    rc.country,
    rc.customer_id,
    rc.first_name,
    rc.last_name,
    rc.num_rentals,
    rc.total_amount,
    case
        when rc.num_rentals = mr.max_num_rentals then rc.num_rentals
        else null
    end as customer_with_most_rentals,
    case
        when rc.payment_date = mr.max_payment_date then rc.payment_date
        else null
    end as customer_last_rental,
    case
        when rc.total_amount = mr.max_total_amount THEN rc.total_amount
        else null
    end as customer_with_highest_amount
from RankedCustomers rc
join MaxRentals mr on rc.country = mr.country
where rc.rental_order = 1 and not(rc.num_rentals != mr.max_num_rentals and rc.payment_date != mr.max_payment_date
and rc.total_amount != mr.max_total_amount)
order by rc.country, rc.rental_order desc;

```

	country(+)	customer_id	first_name	last_name	num_rentals	total_amount	customer_with_most_rentals	customer_last_rental
1	Afghanistan	218	VERA	MCCOY	18	67,82	18	2005-08-23 05:30:19.000
2	Algeria	176	JUNE	CARROLL	37	173,63	37	[NULL]
3	Algeria	441	MARIO	CHEATHAM	28	112,72	[NULL]	2006-02-14 15:16:03.000

Рисунок В. 7 – Результат выполнения задания 7



## Приложение Г

### Результат выполнения «Задания 4»

/\*Задание 1. Напишите SQL-запрос, который выводит всю информацию о фильмах со специальным атрибутом (поле special\_features) равным "Behind the Scenes"\*/

```
select
*
from film f
where special_features = '{"Behind the Scenes"}';
```

film 1

select \* from film f where special\_features = '{"Behind the Scenes"}'; Введите SQL выражение чтобы отфильтровать результаты

	film_id	title	description	release_year	language_id	original_language_id	rental_d
1	43	ATLANTIS CAUSE	A Thrilling Yarn of a Feminist And a Hunter who must Fight a Tec	2 006	1	[NULL]	
2	55	BARBARELLA STREETCAR	A Awe-Inspiring Story of a Feminist And a Cat who must Conque	2 006	1	[NULL]	
3	87	BOONDOCK BALLROOM	A Fateful Panorama of a Crocodile And a Boy who must Defeat a	2 006	1	[NULL]	
4	91	BOUND CHEAPER	A Thrilling Panorama of a Database Administrator And a Astronau	2 006	1	[NULL]	
5	101	BROTHERHOOD BLANKET	A Fateful Character Study of a Butler And a Technical Writer who	2 006	1	[NULL]	
6	107	BUNCH MINDS	A Emotional Story of a Feminist And a Feminist who must Escape	2 006	1	[NULL]	
7	109	BUTTERFLY CHOCOLAT	A Fateful Story of a Girl And a Composer who must Conquer a Hi	2 006	1	[NULL]	
8	115	CAMPUS REMEMBER	A Astounding Drama of a Crocodile And a Mad Cow who must B	2 006	1	[NULL]	
9	136	CHAPLIN LICENSE	A Boring Drama of a Dog And a Forensic Psychologist who must	2 006	1	[NULL]	

Рисунок Г. 1 – Результат выполнения задания 1

/\*Задание 2. Напишите ещё 2 варианта поиска фильмов с атрибутом "Behind the Scenes", используя другие функции или операторы языка SQL для поиска значения в массиве.\*/

```
select
*
from film f
where 'Behind the Scenes' = any (special_features); --Вариант через ANY

select
*
from film f
where 'Behind the Scenes' = all (special_features); --Через all

select
*
from film f
where special_features && array['Behind the Scenes']; --Через И
```

film 1

select \* from film f where special\_features && array['Behind the Scenes']; Введите SQL выражение чтобы отфильтровать результаты

	film_id	title	description	release_year	language_id	original_language_id	rental_d
1	1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist who must Battle	2 006	1	[NULL]	
2	4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbee And a Lumberjack who mus	2 006	1	[NULL]	
3	11	ALAMO VIDEOTAPE	A Boring Epistle of a Butler And a Cat who must Fight a Pastry Ch	2 006	1	[NULL]	
4	13	ALI FOREVER	A Action-Packed Drama of a Dentist And a Crocodile who must E	2 006	1	[NULL]	
5	14	ALICE FANTASIA	A Emotional Drama of a A Shark And a Database Administrator w	2 006	1	[NULL]	
6	15	ALIEN CENTER	A Brilliant Drama of a Cat And a Mad Scientist who must Battle a	2 006	1	[NULL]	
7	17	ALONE TRIP	A Fast-Paced Character Study of a Composer And a Dog who mu	2 006	1	[NULL]	

Рисунок Г. 2 – Результат выполнения задания 2

/\*Задание 3. Для каждого покупателя посчитайте, сколько он брал в аренду фильмов \* со специальным атрибутом "Behind the Scenes". Обязательное условие для выполнения задания: используйте запрос из задания 1, помещённый в CTE\*/

```

with customers_films as(
select distinct
r.customer_id as Айди_покупателя,
count(f.film_id) over (partition by r.customer_id) as films_count
from film f
inner join inventory i on i.film_id = f.film_id
inner join rental r on r.inventory_id = i.inventory_id
where special_features = '{"Behind the Scenes"}'
order by r.customer_id
)
select
*
from customers_films;

```

rental 1 X

with customers\_films as( select distinct r.customer\_id as Айди\_покупателя, count(f.film\_id) over (partition by r.customer\_id) as films\_count from film f inner join inventory i on i.film\_id = f.film\_id inner join rental r on r.inventory\_id = i.inventory\_id where special\_features = '{"Behind the Scenes"}' order by r.customer\_id; )

	123 Айди_покупателя	123 films_count
1	2	3
2	4	1
3	5	4
4	6	1
5	8	4
6	9	1
7	10	4
8	11	2
9	12	2
10	13	2

Рисунок Г. 3 – Результат выполнения задания 3

/\*Задание 4. Для каждого покупателя посчитайте, сколько он брал в аренду фильмов \* со специальным атрибутом "Behind the Scenes". Обязательное условие для выполнения задания: используйте запрос из задания 1, помещённый в подзапрос, который необходимо использовать для решения задания.\*/

```

select distinct
r.customer_id as Айди_покупателя,
count(f.film_id) over (partition by r.customer_id) as films_count
from film f
inner join inventory i on i.film_id = f.film_id
inner join rental r on r.inventory_id = i.inventory_id
where f.film_id in (select film_id from film f2 where special_features = '{"Behind the Scenes"}')
order by r.customer_id;

```

rental 1 X

select distinct r.customer\_id as Айди\_покупателя, count(f.film\_id) over (partition by r.customer\_id) as films\_count from film f inner join inventory i on i.film\_id = f.film\_id inner join rental r on r.inventory\_id = i.inventory\_id where f.film\_id in (select film\_id from film f2 where special\_features = '{"Behind the Scenes"}') order by r.customer\_id;

	123 Айди_покупателя	123 films_count
1	2	3
2	4	1
3	5	4
4	6	1
5	8	4
6	9	1
7	10	4
8	11	2
9	12	2
10	13	2
11	14	5

Рисунок Г. 4 – Результат выполнения задания 4

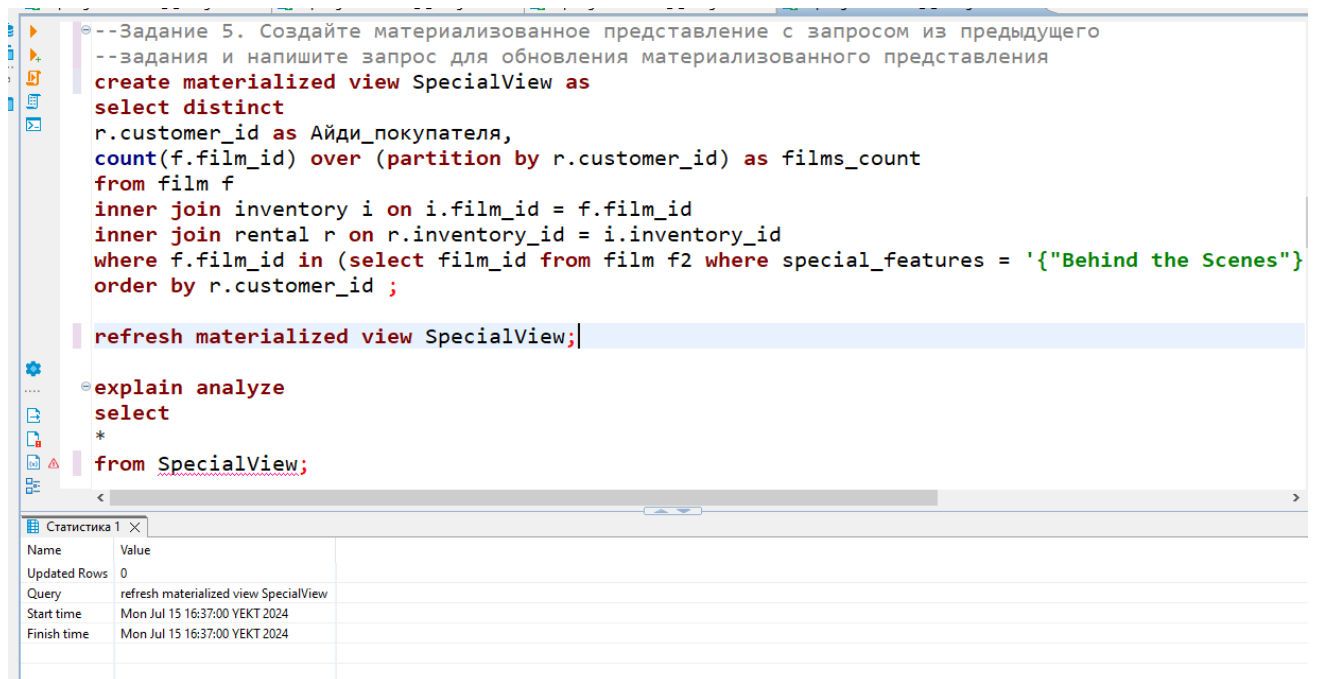


Рисунок Г. 5 – Результат выполнения задания 5

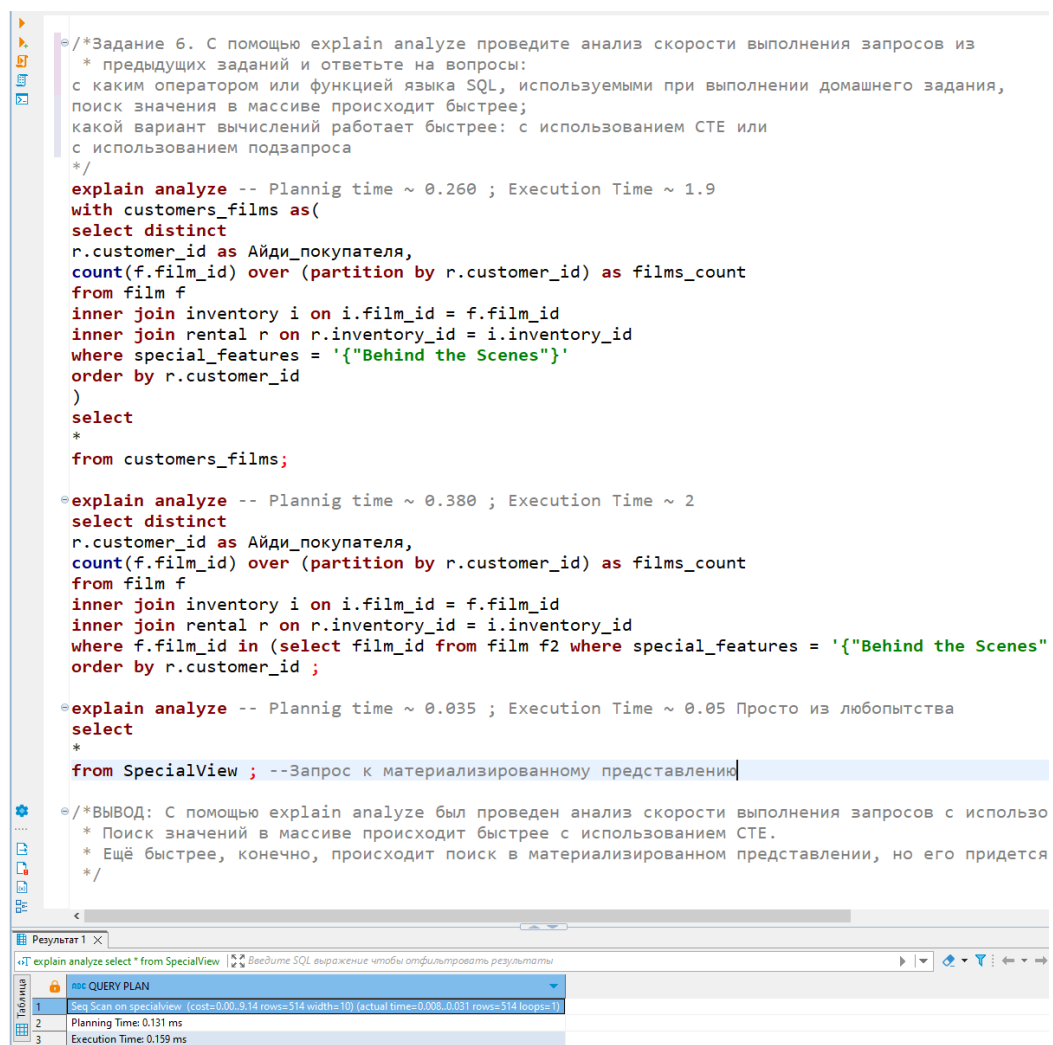


Рисунок Г. 6 – Результат выполнения задания 6

```
--Задание 7. Используя оконную функцию, выведите для каждого сотрудника сведения
--о первой его продаже.
with rental_info as(
select distinct
r.staff_id as Айди_продавца,
first_value(r.rental_date) over (partition by r.staff_id order by r.rental_date)
as first_rental
from rental r
)
select
*
from rental
inner join rental_info on rental_info.first_rental= rental.rental_date
```

rental_id	rental_date	inventory_id	customer_id	return_date	staff_id	last_update	Айди_продавца	first_rental
1	2005-05-24 22:53:30.000	367	130	2005-05-26 22:04:30.000	1	2006-02-15 21:30:53.000	1	2005-05-24 22:53:30.000
4	2005-05-24 23:04:41.000	2452	333	2005-06-03 01:43:41.000	2	2006-02-15 21:30:53.000	2	2005-05-24 23:04:41.000

Рисунок Г. 7 – Результат выполнения задания 7


```
/*Задание 8. Для каждого магазина определите и выведите одним SQL-запросом следующие
* аналитические показатели:
* день, в который арендовали больше всего фильмов (в формате год-месяц-день);
* количество фильмов, взятых в аренду в этот день;
* день, в который продали фильмов на наименьшую сумму (в формате год-месяц-день);
* сумму продажи в этот день.
*/
with shops_ranking as (
select distinct
date(r.rental_date) as day_rental_date,
count (r.rental_id) over (partition by date(r.rental_date)) as day_film_count,
sum (p.amount) over (partition by date(r.rental_date)) as day_amount_sum
from store s
inner join inventory i on i.store_id = s.store_id
inner join rental r on r.inventory_id = i.inventory_id
inner join payment p on p.rental_id = r.rental_id
), min_max as (
select
max(day_film_count) as max_rent,
min(day_amount_sum) as min_amount
from shops_ranking
)
select
*
from shops_ranking sr
join min_max mm on mm.min_amount = sr.day_amount_sum or mm.max_rent = sr.day_film_count
order by day_rental_date
```

day_rental_date	day_film_count	day_amount_sum	max_rent	min_amount
2005-05-24	12	38,88	679	38,88
2005-07-31	679	2 868,21	679	38,88

Рисунок Г. 8 – Результат выполнения задания 8

## Приложение Д


### Справка о проверке в системе «Антиплагиат.ру»

**АНТИПЛАГИАТ**  
ОБНАРУЖЕНИЕ ЗАИМСТВОВАНИЙ

**ТАРИФЫ**  
Демо ⚠  
[ИЗМЕНИТЬ](#)

**ПРОВЕРКИ**  
1 в 6 минут ⌚  
[ПРОВЕРИТЬ ДОКУМЕНТ](#)

**ПОЛЬЗОВАТЕЛЬ** 🔔  
tayoji1381@ikangou.com  
[ВОЙТИ В КАБИНЕТ](#)

**МЕНЮ** ru ▼

ГЛАВНАЯ / КАБИНЕТ

## Кабинет


[ПРОВЕРИТЬ ДОКУМЕНТ](#)  
[ПРОВЕРИТЬ ТЕКСТ](#)  
[ПРИМЕР ОТЧЕТА](#)

**ПАПКИ**  
Все документы

[УДАЛЕННЫЕ ДОКУМЕНТЫ](#) | < > 1/1 > >1

[ПЕРЕМЕСТИТЬ](#) [УДАЛИТЬ](#) [ИСТОРИЯ ОТЧЕТОВ](#) [ПРОВЕРИТЬ ПРАВОПИСАНИЕ](#) [ОФОРМИТЬ БИБЛИОГРАФИЮ](#)

<input type="checkbox"/>	Название ↕	Дата загрузки ↕	Оригинальность	
<input type="checkbox"/>	<a href="#">.PDF PostgreSQL Отчет_Елшанский_Владимир</a>	15 Июл 2024 15:32	97,32%	<a href="#">ПОСМОТРЕТЬ РЕЗУЛЬТАТЫ</a>

 Нужен результат, близкий к корпоративной версии системы Антиплагиат? Проверяйте документы по "Объединенной коллекции".

[КУПИТЬ ПРОВЕРКИ](#) 🔒

Рисунок Д.1 – Результаты проверки