Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и

радиоэлектроники»

Кафедра электронных вычислительных машин

Лабораторная работа №4

«Реализация SQL-запросов на простую выборку данных» (Видеопрокат)

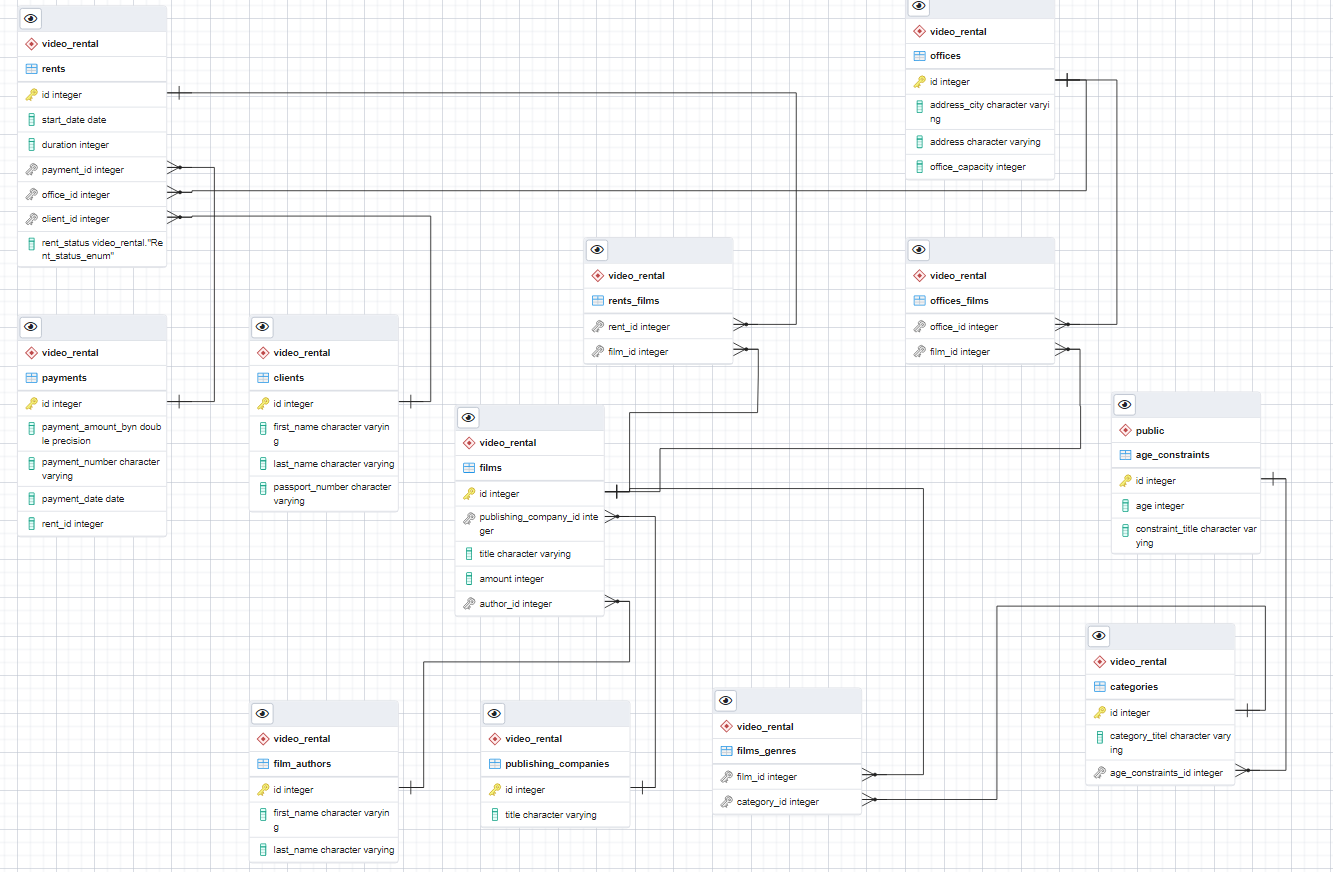
Выполнили Проверила:

студенты группы 950503: Куприянова Д.В.

Зарубо Д. Ю.

Ященко В. П.

Минск 2022

**ER-диаграмма «Видеопрокат»**

# Оператор SELECT

Оператор языка SQL SELECT - основная строительная конструкция для  
создания любого, простого или сложного запроса к базе данных. Без него,  
как и без фундамента для постройки, невозможно получить ни одну выборку  
данных из базы.  
Оператор SELECT состоит из нескольких предложений (разделов),  
рассмотрим основной из них:  
• SELECT определяет список возвращаемых столбцов (как существующих,  
так и вычисляемых), их имена, ограничения на уникальность строк в  
возвращаемом наборе, ограничения на количество строк в возвращаемом  
наборе, реализует операцию проекции, то есть указание подмножества  
столбцов из таблиц табличного выражения, а также операцию  
переименования столбцов и операцию добавления новых вычислимых  
столбцов.

Можно выбрать все данные из таблицы:



*Рисунок 1. Выборка всех данных из таблицы*

**

*Рисунок 2. Результат выполнения команды*

Можно выбрать только интересующие нас столбцы:

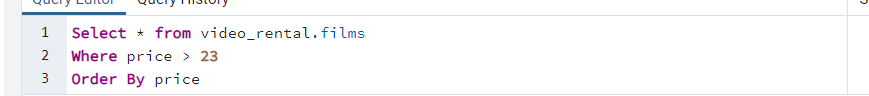


*Рисунок 3. Выборка конкретных данных из таблицы*



*Рисунок 4. Результат выполнения команды*

Теперь сделаем выборку из таблицы films, содержащую все поля. Критерий выборки – цена аренды фильма price превышает 23. Записи отсортированы по полю price

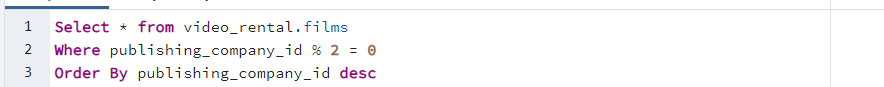


*Рисунок 5. Выборка конкретных данных из таблицы*

**

*Рисунок 6. Результат выполнения команды*

В данном примере мы сделаем выборку из таблицы “films” записей, у которых номер компании-производителя четный.

**

*Рисунок 7. Выборка конкретных данных из таблицы*

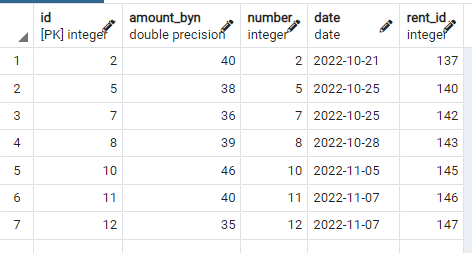


*Рисунок 8. Результат выполнения команды*

Тут выполним выборку из таблицы payments записей, где сумма аренды больше 30 и меньше 50.



*Рисунок 9. Выборка конкретных данных из таблицы*

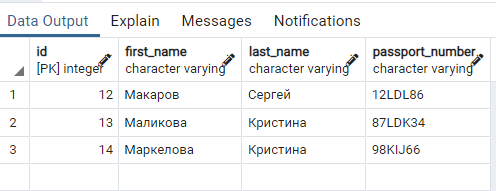
**

*Рисунок 10. Результат выполнения команды*

В данном примере мы сделаем выборку из таблицы “cliets” записей, где имя клиента начинается с буквы М.

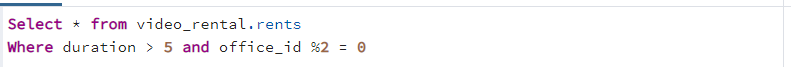


*Рисунок 11. Выборка конкретных данных из таблицы*

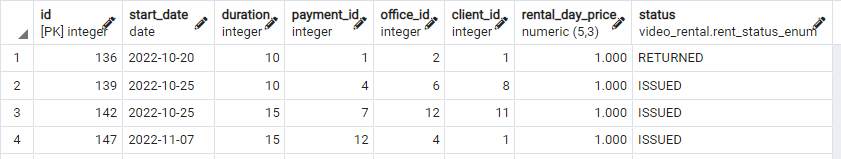


*Рисунок 12. Результат выполнения команды*

Тут сделаем выборку из таблицы rents, где аренда длится больше 5 дней и фильм был выдан в отделении с четным номером.

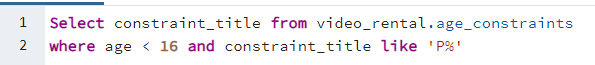


*Рисунок 13. Выборка конкретных данных из таблицы*

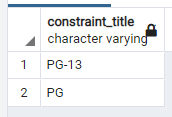


*Рисунок 14. Результат выполнения команды*

Выберем из таблицы age\_constraints названия возрастных ограничений, разрешенный возраст у которых ниже 16, которые начинаются на букву ‘P’ и выведем в обратном порядке.



*Рисунок 15. Выборка конкретных данных из таблицы*

**

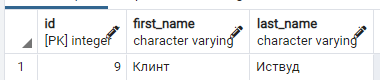
*Рисунок 16. Результат выполнения команды*

# Предложение WHERE

Оператор SQL WHERE служит для задания дополнительного условия  
выборки, операций вставки, редактирования и удаления записей.  
Предложение [[WHERE (SQL)|WHERE]] используется для определения,  
какие строки должны быть выбраны из табличного выражения в  
предложении FROM. Мы можем описывать несколько условий выборки:



*Рисунок 17. Выборка конкретных данных из таблицы*



*Рисунок 18. Результат выполнения команды*

Мы выполнили выборку из таблицы режиссеров фильмов, причем искали режиссера с конкретным именем и фамилией.

# Предложение ORDER BY

ORDER BY — необязательное предложение операторов SELECT и UNION ,  
который означает что операторы SELECT , UNION возвращают набор строк,  
отсортированных по значениям одного или более столбцов. Его можно  
применять как к числовым столбцам, так и к строковым. В последнем случае,  
сортировка будет происходить по алфавиту.  
Использование предложения ORDER BY является единственным способом  
отсортировать результирующий набор строк. Без этого предложения СУБД  
может вернуть строки в любом порядке. Если упорядочение  
необходимо, ORDER BY должен присутствовать в SELECT , UNION .  
Сортировка может производиться как по возрастанию, так и по убыванию  
значений.  
• Параметр ASC (по умолчанию) устанавливает порядок сортировки по  
возрастанию, от меньших значений к большим.  
• Параметр DESC устанавливает порядок сортировки по убыванию, от  
больших значений к меньшим

Выберем клиентов из таблицы clients по возрастанию их имен, но по убыванию фамилий



*Рисунок 17. Выборка конкретных данных из таблицы*



*Рисунок 18. Результат выполнения команды*

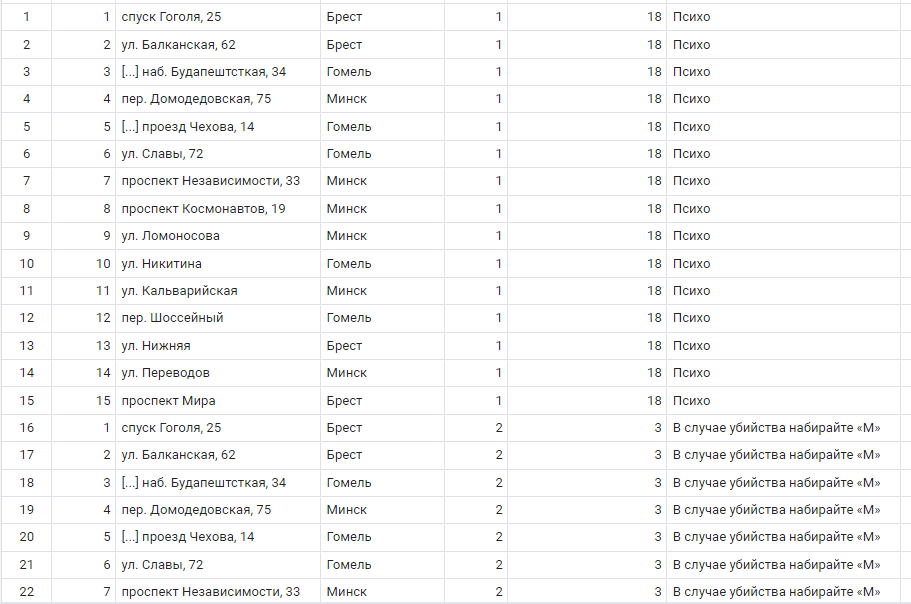
# CROSS JOIN

Оператор SQL CROSS JOIN формирует таблицу перекрестным  
соединением (декартовым произведением) двух таблиц. При использовании  
оператора SQL CROSS JOIN каждая строка левой таблицы сцепляется с  
каждой строкой правой таблицы. В результате получается таблица со всеми  
возможными сочетаниями строк обеих таблиц.

Сделаем пересечение таблиц offices и films



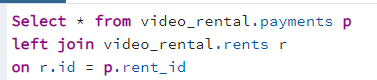
*Рисунок 19. Выборка конкретных данных из таблицы*



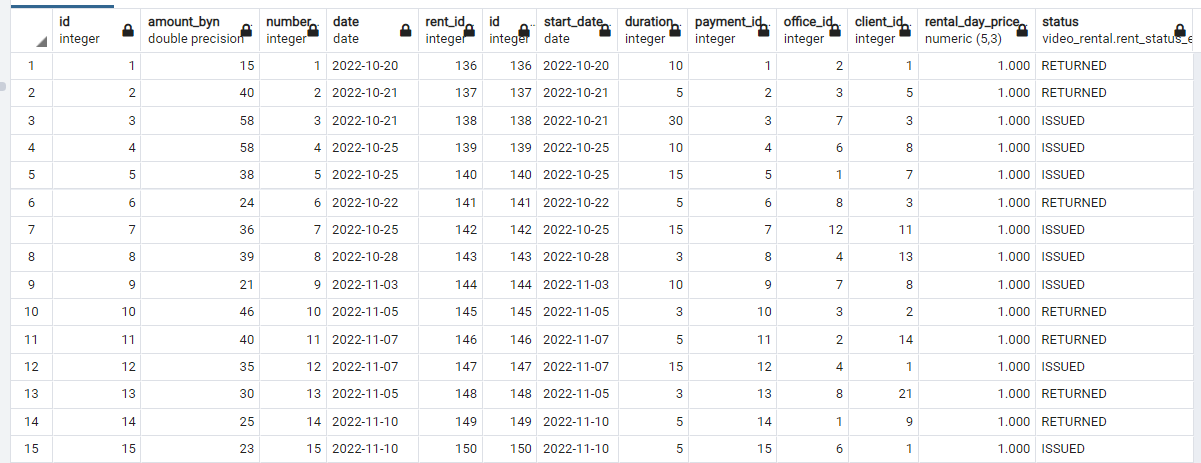
# Рисунок 20. Часть результирующей таблицы

# INNER JOIN

Объединяет записи из двух таблиц, если в связующих полях этих  
таблиц содержатся одинаковые значения.  
Операцию INNER JOIN можно использовать в любом  
предложении FROM. Это самый распространенный тип объединения.  
С его помощью происходит объединение записей из двух таблиц по  
связующему полю, если оно содержит одинаковые значения в обеих  
таблицах.



*Рисунок 21. Выборка конкретных данных из таблицы*

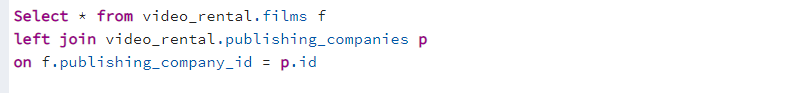
**

*Рисунок 22. Результат выполнения команды Inner Join*

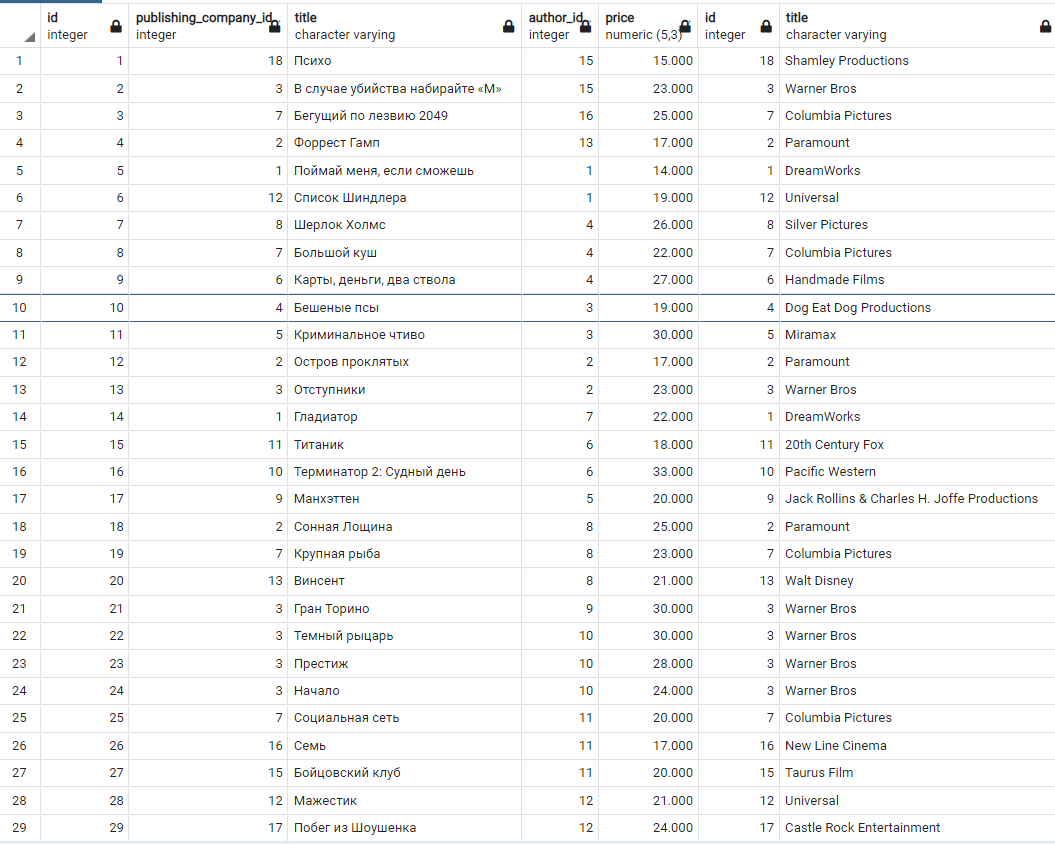
# OUTER JOIN

Внешние соединения - это соединения, которые возвращают совпадающие  
значения и несопоставимые значения из одной или обеих таблиц. Существует  
несколько типов внешних соединений:  
• левое внешнее соединение (LEFT OUTER JOIN) - выборка будет  
содержать все строки из первой или левой таблицы;  
• правое внешнее соединение (RIGHT OUTER JOIN) - выборка будет  
содержать все строки из второй или правой таблицы;  
• полное внешнее соединение (FULL OUTER JOIN) - выборка будет  
содержать все строки из обеих таблиц.  
Для примера воспользуемся FULL OUTER JOIN:

Cделаем left join над таблицами films и publishing\_companies



. *Рисунок 23. Выборка конкретных данных из таблицы*

**

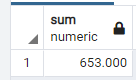
*Рисунок 24. Результат выполнения команды Left Join*

# СКАЛЯРНЫЕ ФУНКЦИИ

Скалярные функции Transact-SQL используются в создании скалярных  
выражений. (Скалярная функция выполняет вычисления над одним  
значением или списком значений, тогда как агрегатная функция выполняет  
вычисления над группой значений из нескольких строк.) Скалярные функции  
можно разбить на следующие категории:  
• числовые функции;  
• функции даты;  
• строковые функции;  
• системные функции;  
• функции метаданных.

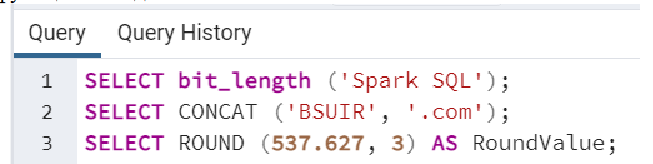
Выполним подсчет общей стоимости аренды всех фильмов

*Рисунок 25. Выборка конкретных данных из таблицы*

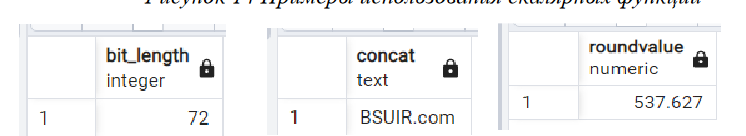
**

*Рисунок 26. Результат подсчета суммы*

Некоторые другие функции



*Рисунок 27. Пример функций*



*Рисунок 27. Результат функций*