Белорусский Государственный Университет

Информатики и Радиоэлектроники

Кафедра ЭВМ

Отчет по лабораторной работе № 2

Тема: «Создание реляционной модели данных по ER-модели»

Выполнили: Проверила:

Куприянова Д.В

ст. гр. 950503

Зарубо Д. Ю

Ященко В.П

Минск 2022

1. **ЦЕЛЬ РАБОТЫ**

Познакомиться с реляционной моделью данных. Для указанного варианта

задания преобразовать ER-диаграмму в реляционную модель данных.

Реализовать полученную реляционную модель данных в среде целевой СУБД.

1. **ЗАДАНИЕ**

1) Проверить ER-диаграмму, созданную в лабораторной работе No1.

2) Выполнить преобразование ER-диаграммы в реляционную модель в двух вариантах:

- вид «бумажного» варианта преобразования.

- «автоматизированный».

3) Сравнить полученные диаграммы и, если есть расхождения в полученных

реляционных диаграммах, найти несоответствия и устранить их.

4) Оформить отчет.

Порядок получения реляционной модели из ER-диаграммы:

Алгоритм преобразования ER-диаграммы в реляционную модель (схему)

состоит из следующих шагов:

Шаг 1. Каждый объект на ER-диаграмме превращается в таблицу. Имя

объекта становится именем таблицы.

Шаг 2. Каждый атрибут объекта становится возможным столбцом с тем

же именем; при этом может выбираться более точный формат данных.

Столбцы, соответствующие необязательным атрибутам, могут содержать

неопределенные значения; столбцы, соответствующие обязательным

атрибутам, – не могут.

Шаг 3. Уникальные (ключевые) атрибуты объекта превращаются в

первичный ключ таблицы. Если имеется несколько возможных уникальных

идентификаторов, то выбирается наиболее подходящий для использования.

Шаг 4. Связи «один-ко-многим» (в том числе и связи «один-к-одному»)

становятся внешними ключами. Внешний ключ добавляется в виде столбца

(столбцов) в таблицу, соответствующую объекту со стороны «многие» связи.

Необязательные связи соответствуют столбцам, допускающим неопределенные

значения; обязательные связи – столбцам, не допускающим неопределенные

значения.

Шаг 5. Связи «многие-ко-многим» реализуются через промежуточную

таблицу. Эта таблица будет содержать как минимум столбцы внешних ключей

на соответствующие объекты. Первичный ключ промежуточной таблицы

должен включать в себя все внешние ключи на объекты, участвующие в связи.

Шаг 6. Если связь имеет дополнительные атрибуты, то, как и в случае

атрибутов объектов, они становятся возможным столбцом таблицы:

· для связей «один-ко-многим» – в таблице со стороны «многие» (вместе с

внешним ключом);

· для связей «многие-ко-многим» – в промежуточной таблице (при этом

атрибуты, расширяющие комбинацию в связи (например «дата»), также

должны войти в состав первичного ключа промежуточной таблицы).

1. **УСТАНОВКА POSTGRESQL И PGADMIN**

PostgreSQL — это бесплатная объектно-реляционная СУБД с мощным функционалом, который позволяет конкурировать с платными базами данных, такими как Microsoft SQL, Oracle. PostgreSQL поддерживает пользовательские данные, функции, операции, домены и индексы. В данной статье мы рассмотрим установку и краткий обзор по управлению базой данных PostgreSQL. Мы установим СУБД PostgreSQL в Windows 10, создадим новую базу, добавим в неё таблицы и настроим доступа для пользователей. Также мы рассмотрим основы управления PostgreSQL с помощью SQL shell и визуальной системы управления PgAdmin. Надеюсь эта статья станет хорошей отправной точкой для обучения работы с PostgreSQL и использованию ее в разработке и тестовых проектах.

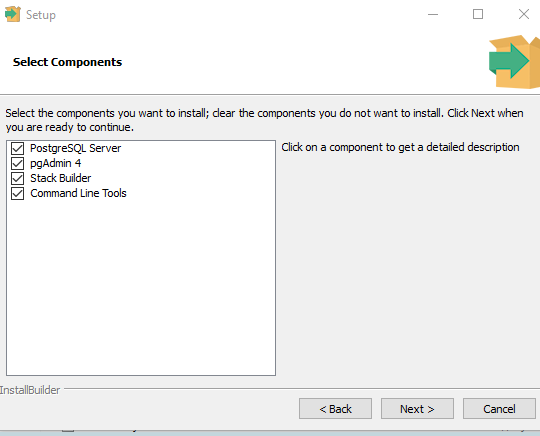
Установка PostgreSQL 11 в Windows 10

Для установки PostgreSQL перейдите на сайт [https://www.postgresql.org](https://www.postgresql.org/) и скачайте последнюю версию дистрибутива для Windows, на сегодняшний день это версия PostgreSQL 11 (в 11 версии PostgreSQL поддерживаются только 64-х битные редакции Windows). После загрузки запустите инсталлятор.

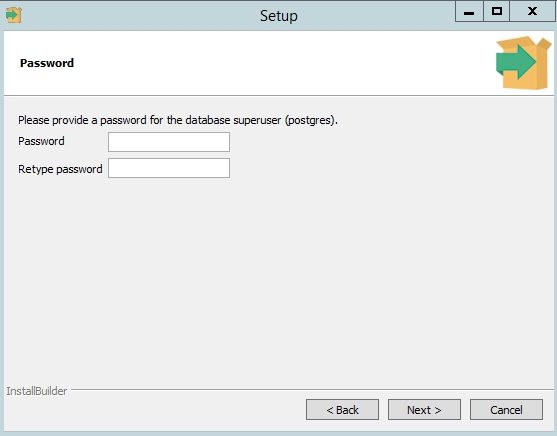


В процессе установки установите галочки на пунктах:

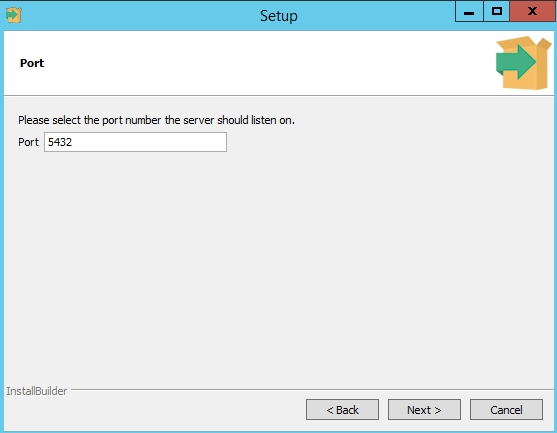
* PostgreSQL Server – сам сервер СУБД
* PgAdmin 4 – визуальный редактор SQL
* Stack Builder – дополнительные инструменты для разработки (возможно вам они понадобятся в будущем)
* Command Line Tools – инструменты командной строки



Установите пароль для пользователя postgres (он создается по умолчанию и имеет права суперпользователя).



По умолчание СУБД слушает на порту 5432, который нужно будет добавить в исключения в правилах фаерволла.



Нажимаете Далее, Далее, на этом установка PostgreSQL завершена.

Доступ к PostgreSQL по сети, правила файерволла

Чтобы разрешить сетевой доступ к вашему экземпляру PostgreSQL с других компьютеров, вам нужно создать правила в файерволе. Вы можете создать правило через командную строку или PowerShell.

Запустите командную строку от имени администратора. Введите команду:

netsh advfirewall firewall add rule name="Postgre Port" dir=in action=allow protocol=TCP localport=5432

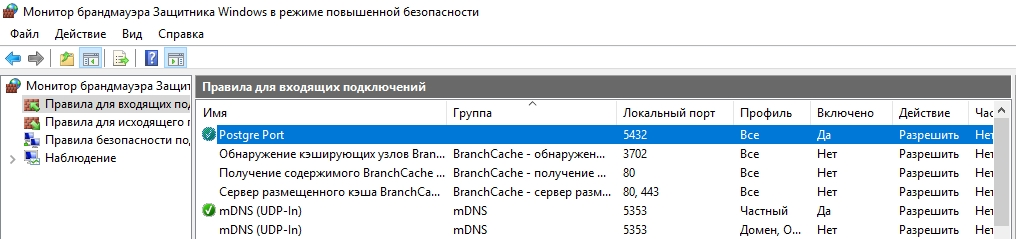
Где rule name – имя правила

Localport – разрешенный порт

Либо вы можете [создать правило](https://winitpro.ru/index.php/2019/09/25/upravlenie-windows-firewall-powershell/), разрешающее TCP/IP доступ к экземпляру PostgreSQL на порту 5432 с помощью PowerShell:

New-NetFirewallRule -Name 'POSTGRESQL-In-TCP' -DisplayName 'PostgreSQL (TCP-In)' -Direction Inbound -Enabled True -Protocol TCP -LocalPort 5432

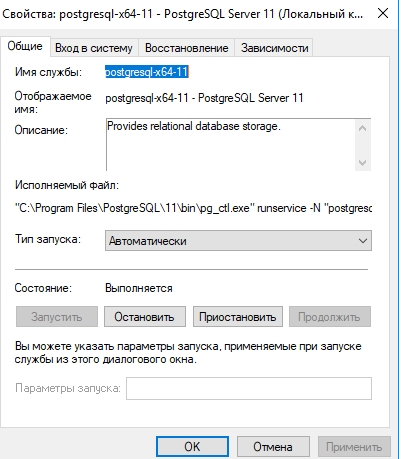
После применения команды в брандмауэре Windows появится новое разрешающее правило для порта Postgres.



Совет. Для изменения порта в установленной PostgreSQL отредактируйте файл postgresql.conf по пути C:\Program Files\PostgreSQL\11\data.

Измените значение в пункте port = 5432 . Перезапустите службу сервера postgresql-x64-11 после изменений. Можно перезапустить службу с помощью PowerShell:

Restart-Service -Name postgresql-x64-11



Более подробно о настройке параметров в конфигурационном файле postgresql.conf с помощью тюнеров смотрите в [статье](https://winitpro.ru/index.php/2019/09/26/ustanovka-postgresql-db-centos/).

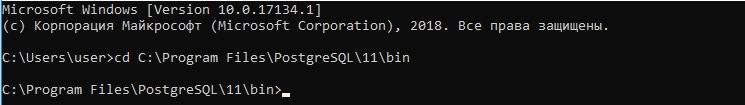
Утилиты управления PostgreSQL через командную строку

Рассмотрим управление и основные операции, которые можно выполнять с PostgreSQL через командную строку с помощью нескольких утилит. Основные инструменты управления PostgreSQL находятся в папке bin, потому все команды будем выполнять из данного каталога.

Запустите командную строку.

Совет. Перед запуском СУБД, смените кодировку для нормального отображения в русской Windows 10. В командной строке выполните: chcp 1251

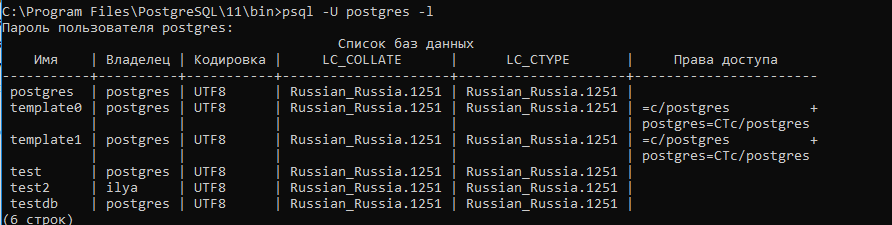
Перейдите в каталог bin выполнив команду: CD C:\Program Files\PostgreSQL\11\bin



Основные команды PostgreSQL:

Проверка установленной версии СУБД: psql –V

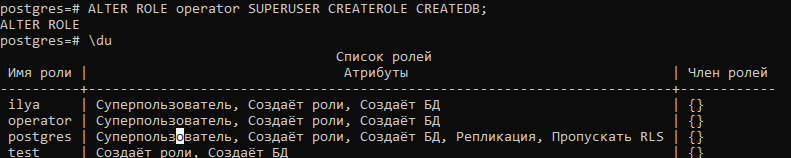
Для создания новой базы данных воспользуйтесь утилитой createdb: createdb -U postgres testdb (где postgres суперпользователь, testdb новая база данных)Введите пароль суперпользователя.createdb -U postgres 

Проверить список активных баз: Psql -U postgres –l (пароль)

С помощью инструмента createuser cоздадим нового пользователя: createuser –U postgres operator (где operator -имя нового пользователя)createuser –U postgres - создать пользователя 

Предоставим пользователю привилегии суперпользователя (на практике этого делать не надо). Запустите интерактивную командную оболочку управления PostgreSQL (shell): psql –U postgres . С помощью SQL команды ALTER ROLE предоставим нужные права нашему пользователю: ALTER ROLE operator SUPERUSER CREATEROLE CREATEDB; . Мы предоставили пользователю права суперпользователя, права на создание ролей и баз данных.

Для выводы списка пользователей и ролей в СУБД выполните команду: \du

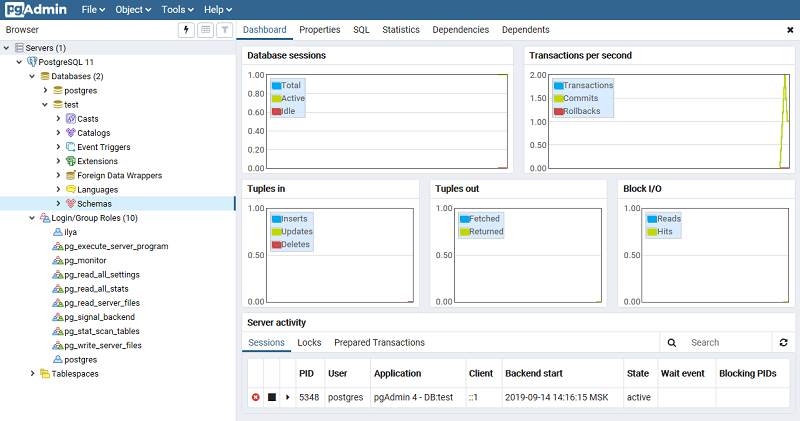


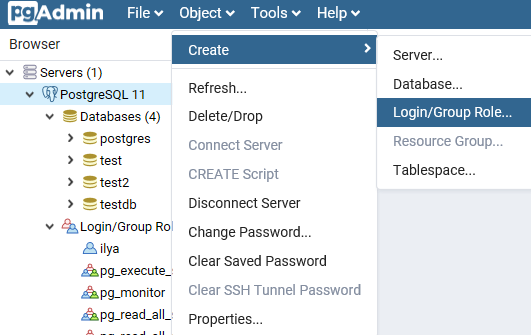
PgAdmin: Визуальный редактор для PostgresSQL

Редактор PgAdmin служит для упрощения управления базой данных PostgresSQL в понятном визуальном режиме.

Для запуска редактора запустите PgAdmin 4 в меню Пуск

Для доступа нужно ввести пароль суперпользователя postgres

В панели Servers вы можете раскрыть список активных БД.  


В панели управления возможно быстро создать нового пользователя и группу, предоставить ему права. Для этого Откройте меню Object -> Create -> Create Login/Group.

Для создания новой базы данных достаточно выбрать: Database в меню Object -> Create. В новом поле указать имя базы и владельца.

По умолчанию все созданные базы хранятся в каталоге base по пути C:\Program Files\PostgreSQL\11\data\base.

Для каждой БД существует подкаталог внутри PGDATA/base, названный по OID базы данных в pg\_database. Этот подкаталог по умолчанию является местом хранения файлов базы данных; в частности, там хранятся её системные каталоги. Каждая таблица и индекс хранятся в отдельном файле.

Для резервного копирования и восстановления лучше использовать инструмент Backup в панели инструментов Tools. Для автоматизации бэкапа PostgreSQL из командной строки используйте утилиту pg\_dump.exe.

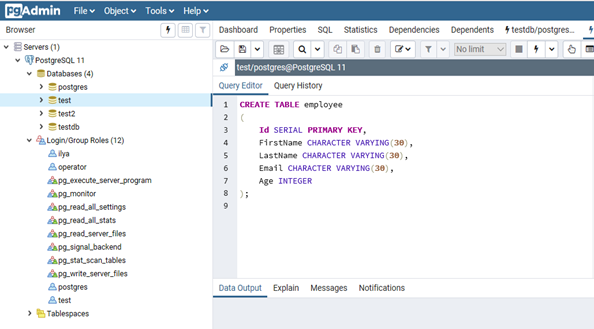
Query Tool: использование SQL запросов в PostgreSQL

Для написания SQL запросов в удобном графическом редакторе используется встроенный в pgAdmin инструмент Query Tool. Например, вы хотите создать новую таблицу в базе данных через инструмент Query Tool.

Выберите базу данных, в панели Tools откройте Query Tool

Создадим таблицу сотрудников:

CREATE TABLE employee  
(  
Id SERIAL PRIMARY KEY,  
FirstName CHARACTER VARYING(30),  
LastName CHARACTER VARYING(30),  
Email CHARACTER VARYING(30),  
Age INTEGER  
);

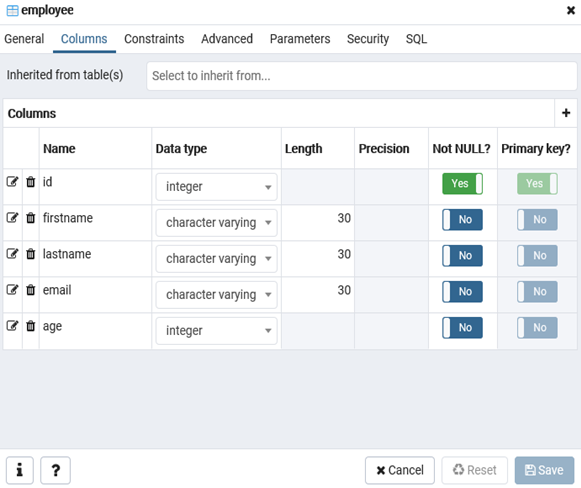


Id — номер сотрудника, которому присвоен ключ SERIAL. Данная строка будет хранить числовое значение 1, 2, 3 и т.д., которое для каждой новой строки будет автоматически увеличиваться на единицу. В следующих строках записаны имя, фамилия сотрудника и его электронный адрес, которые имеют тип CHARACTER VARYING(30), то есть представляют строку длиной не более 30 символов. В строке — Age записан возраст, имеет тип INTEGER, т.к. хранит числа.

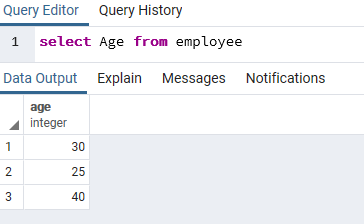
После того, как написали код SQL запроса в Query Tool, нажмите клавишу F5 и в базе будет создана новая таблица employee.

Для заполнения полей в свойствах таблицы выберите таблицу employee в разделе Schemas -> Tables. Откройте меню Object инструмент View/Edit Data.

Здесь вы можете заполнить данные в таблице.



После заполнения данных выполним инструментом Query простой запрос на выборку:  
select Age from employee;



1. **РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ**

Готовая ER диаграмма, отображающая базу данных проката видеодисков, приведена на рисунке 1.

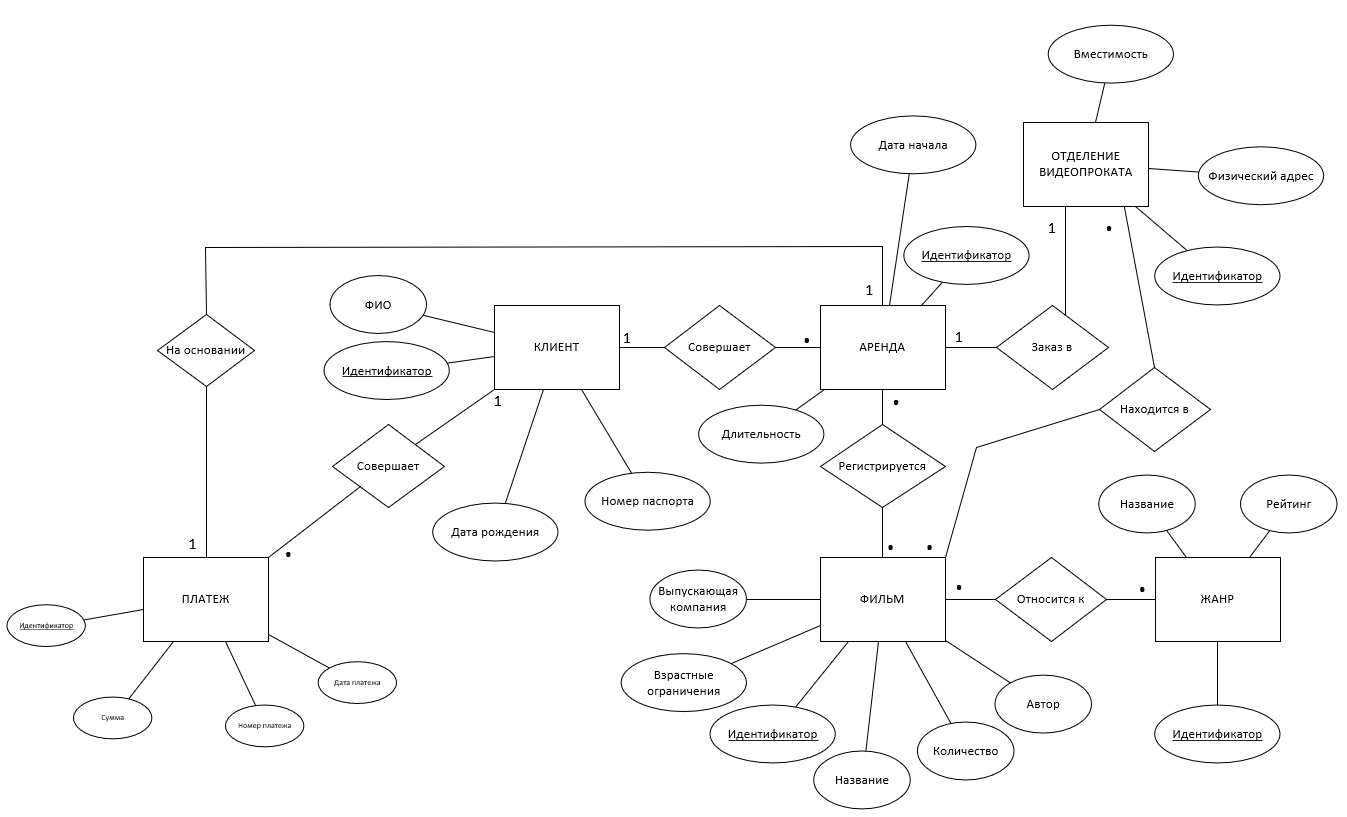


Рисунок 1 – ER диаграмма проката видеодисков

Выполнение шагов 1 – 3 (Рисунок 2):

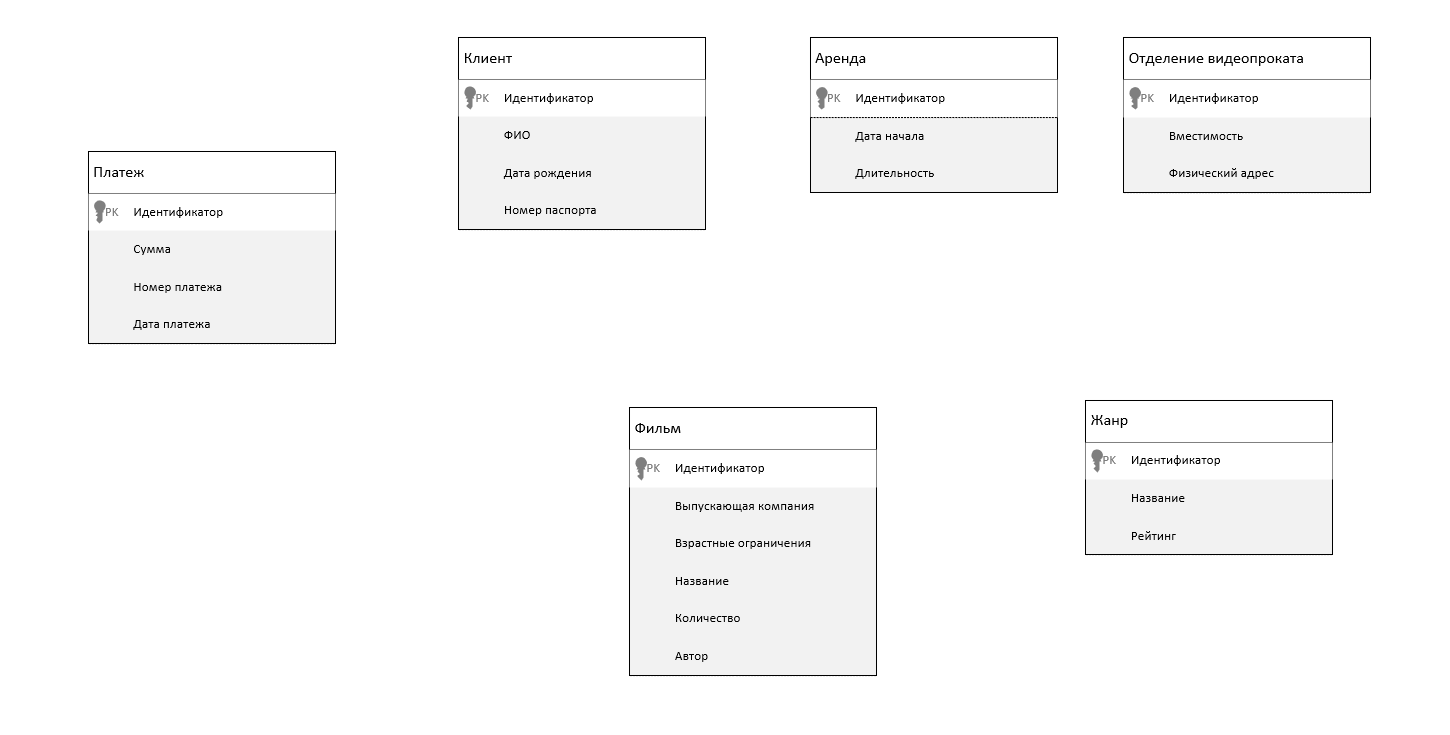


Рисунок 2 – превращение ER-диаграммы в реляционную схему

Выполнение шага 4 (Рисунок 3):

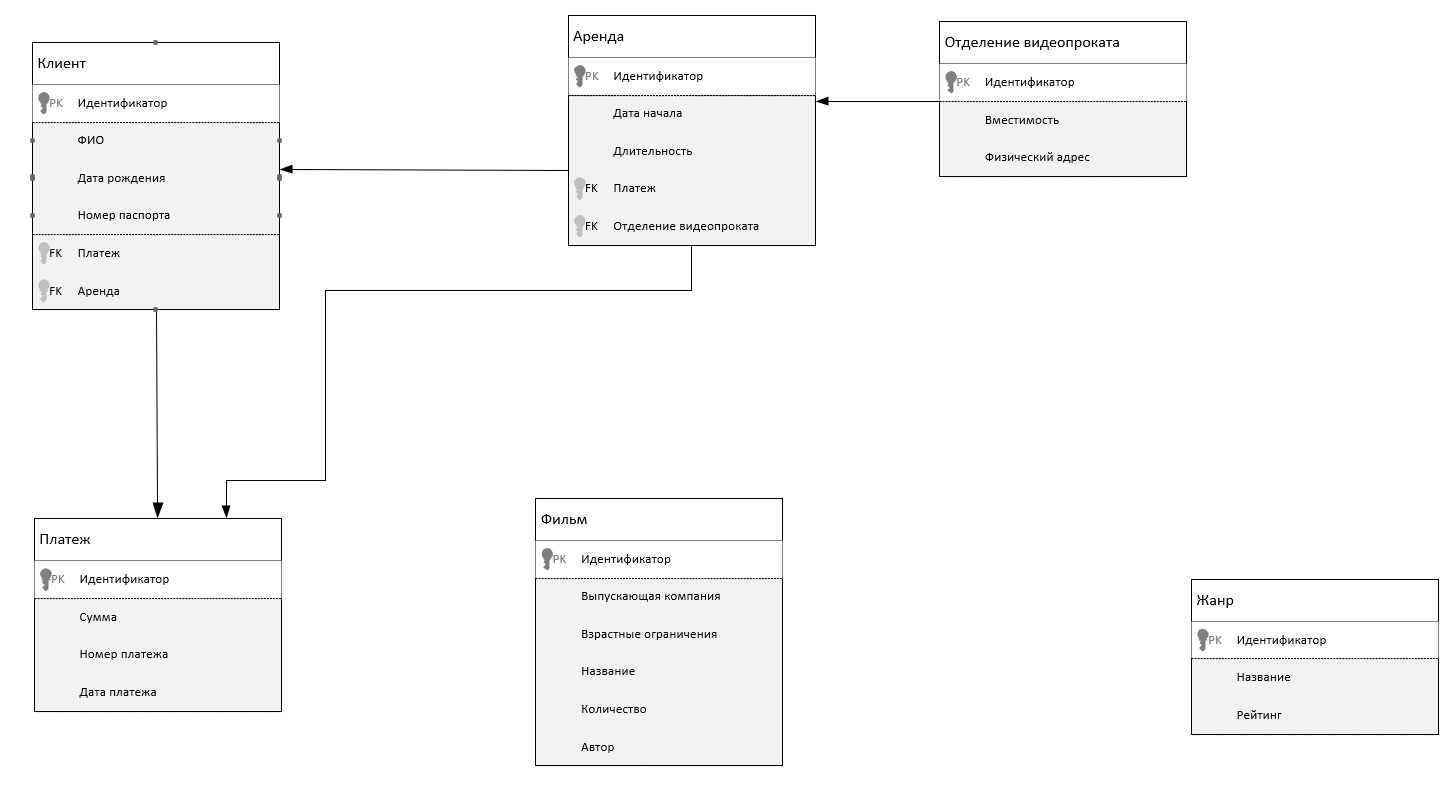


Рисунок 3 – связи один-ко-многим

Выполнение шага 5 (Рисунок 4):

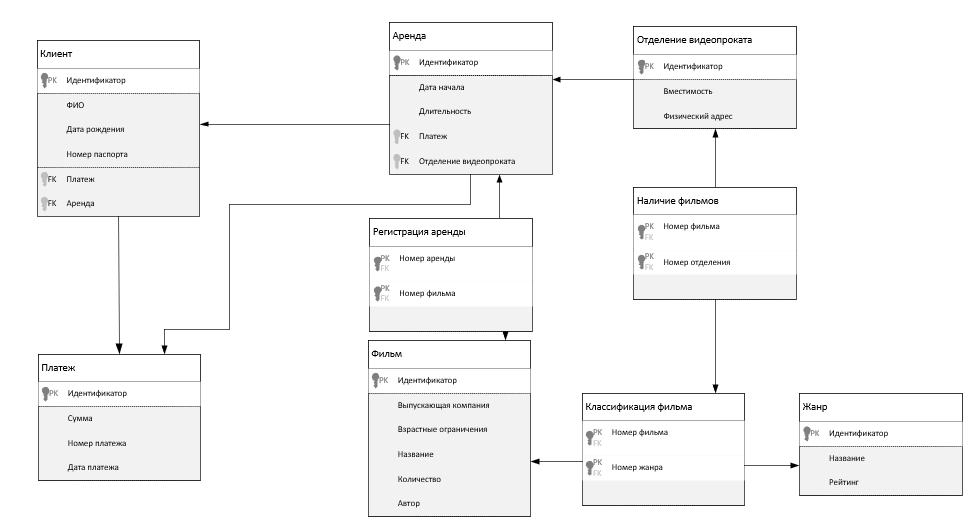


Рисунок 4 – связи многие-ко-многим

**Преобразование реляционной модели в целевой СУБД**

1. **ВЫВОД**

В ходе выполнения лабораторной работы было выполнено преобразование ER-диаграммы в реляционную модель данных, а также реляционная модель данных была реализована в среде целевой СУБД (Postgres).