МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Казанский национальный исследовательский технологический университет»
(ФГБОУ ВО «КНИТУ»)

Кафедра Автоматизированных систем сбора и обработки информации

Направление 09.03.01 «Информатика и вычислительная техника»

Профиль Автоматизированные системы обработки информации и управления

Группа 8191-21

КУРСОВАЯ РАБОТА

по дисциплине: «Базы данных»

на тему: «Проектирование БД для домашней медиатеки»

Исполнитель	04.06.21 (дата, подпись)	Шайманов Д.Н (Ф.И.О.)
Руководитель	04.06.21 (дата, подпись)	Томилова М.Н. (Ф.И.О.)
Нормоконтролер	04.06.21 (дата, подпись)	Томилова М.Н. (Ф.И.О.)

Работа защищена с оценкой отл.

Руководитель 04.06.21
(дата, подпись)

КАЗАНЬ, *2021*

ЗАДАНИЕ

На курсовую работу студенту кафедры: Шайманову Дмитрию Николаевичу, 8191-21.

Тема работы: Проектирование БД для домашней медиатеки.

Исходные данные к работе: Требуется разработать базу данных для хранения и обработки информации согласно предметной области, указанной в теме курсовой работы. Для реализации применяется реляционная СУБД MySQL. Проект должен включать не менее четырех таблиц, характерные запросы к базе данных, функции для автоматизации вычислений, процедуры для автоматизации рутинных действий пользователей базы данных, а также триггеры, позволяющие поддерживать целостность базы данных.

Содержание расчетно-пояснительной записки (включая перечень подлежащих разработке вопросов, включая вопросы стандартизации и контроля качества).

1. Изучение предметной области, постановка задачи.
2. Концептуальное, логическое и физическое моделирование базы данных.
3. Описание объектов базы данных: запросов, пользовательских функций, хранимых процедур и триггеров.

Перечень графического материала (схемной документации)

1. Модель «сущность-связь» базы данных в нотации П.Чена (1 рис.).
2. EER-модель базы данных в нотации IDEF1X (для объектов) и Crow's Foot (для связей) (1 рис.).

Консультанты по проекту (с указанием относящихся к ним разделов)

_____

_____

Дата выдачи задания «10» февраля 2021 г.

Руководитель работы _____ (Томилова М.Н.)

(подпись) (фамилия, инициалы)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 4

1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5

2. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ 7

3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 11

4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 16

5. ЗАПРОСЫ 29

6. ОПИСАНИЕ ОБЪЕКТОВ ПРОЦЕДУРНОГО SQL 32

6.1 Функции 32

6.2 Процедуры 35

7. ТРИГГЕРЫ 41

ЗАКЛЮЧЕНИЕ 43

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 44

ВВЕДЕНИЕ

В современном мире можно наблюдать тенденцию развития IT сферы, что означает, что системы баз данных тоже развиваются. Вместе с тем им находят всё больше применения даже в самых обыденных вещах, например, дома. Именно в такой области и развиваются базы данных — создание домашней медиатеки.

На данный момент есть множество онлайн сервисов предоставляющих доступ ко множеству фильмов, сериалов, музыке и картинкам, но есть множество таких фильмов, сериалов, музыки и картинок, которые нам и вовсе не нужны. Минусом также является постоянная потребность в стабильном интернет сигнале. Эти причины на фоне домашней медиатеки делают онлайн сервисы менее приятными в использовании. Так, в домашней медиатеке существует сервер, на котором хранится вся информация: фильмы, сериалы, музыка и картинки, но уже те, которые нам нужны. Из-за того, что у каждого зарегистрированного в медиатеке пользователя и его устройства, есть доступ к файлам этой самой медиатеки, следовательно они могут их просмотреть со своего устройства.

Отличным примером такой медиатеки служит Smart View от компании Samsung. В которой реализованы все функции, описанные выше, и даже больше.

В данной курсовой работе, будет создана база данных домашней медиатеки, а также реализованы некоторый её функционал.

1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Домашняя медиатека — это система, которая предоставляет доступ к файлам базы данных с любого устройства. Для начала, пользователю требуется зарегистрировать себя и своё устройство в медиатеке. Затем у него есть две опции: добавить или выбрать файл из медиатеки. При добавлении фильма или сериала пользователь должен обязательно указать название, тип — сериал это или же кино, и устройство, на котором оно находится, которое в свою очередь должно быть уже зарегистрировано в медиатеке; при добавлении музыки пользователь должен обязательно указать автора и название композиции и также, как в случае с фильмом или сериалом — устройство, на котором оно находится, которое в свою очередь также должно быть зарегистрировано в медиатеке. В случае выбора файла из медиатеки пользователь может воспользоваться поиском по системе и непосредственно выбрать файл и начать просмотр фильма или сериала или прослушать музыку.

Пользователю доступны следующие функции:

- Добавление новых фильмов, сериалов, музыки в медиатеку;
- Удаление существующих фильмов, сериалов, музыки из медиатеки;
- Изменение существующих данных фильмов, сериалов, музыки в медиатеке;
- Добавление новых пользователей в базу данных медиатеки;
- Удаление пользователей из базы данных медиатеки;
- Изменение данных пользователей в базе данных медиатеки;
- Добавление новых устройств в базу данных медиатеки;
- Удаление устройств из базы данных медиатеки;
- Изменение данных устройств в базе данных медиатеки;

В итоге, получается интерактивный каталог, состоящий из фильмов, сериалов и музыки, доступ к которому может получить любой зарегистрированный в базе данных пользователь и использовать файлы находящиеся не только на его устройстве, но и файлами находящимися на

других устройствах, благодаря тому, что они тоже будут зарегистрированы в медиатеке [1]. (ПК-4)

2. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

Проектируется БД домашней медиатеки, которая позволит организовать единую домашнюю систему фильмов и музыки с доступом с любого устройства, зарегистрированного в данной базе данных.

На основе информации, полученной в результате анализа предметной области были выделены следующие сущности: (ПК-4)

1) УСТРОЙСТВО;
2) ВЛАДЕЛЕЦ УСТРОЙСТВА;
3) ФИЛЬМЫ/СЕРИАЛЫ;
4) РЕЖИССЁРЫ;
5) АКТЁРЫ;
6) СТРАНЫ;
7) ЖАНРЫ(в кинематографе);
8) РЕЖИССЁР ФИЛЬМА/СЕРИАЛА;
9) АКТЁРЫ ФИЛЬМА/СЕРИАЛА;
10) СТРАНЫ ФИЛЬМА/СЕРИАЛА;
11) ЖАНРЫ ФИЛЬМА/СЕРИАЛА;
12) МУЗЫКА;
13) ЖАНРЫ(в музыке);
14) АВТОР;
15) ЖАНРЫ МУЗЫКИ;
16) АВТОРЫ МУЗЫКИ.

Сущность УСТРОЙСТВО содержит в себе информацию об устройствах: название устройства, дату добавления. Подробное описание атрибутов сущности приведено в таблице 1. (ПК-3)

Сущность ВЛАДЕЛЕЦ УСТРОЙСТВА содержит в себе информацию о владельцах устройств: имя, фамилию отчество. Связан с УСТРОЙСТВОМ. Одно УСТРОЙСТВО может принадлежать одному ВЛАДЕЛЬЦУ

УСТРОЙСТВА. У одного ВЛАДЕЛЬЦА УСТРОЙСТВА может быть несколько УСТРОЙСТВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 2. (ПК-3)

Сущность ФИЛЬМЫ/СЕРИАЛЫ содержит в себе информацию о фильмах и сериалах: название фильма или сериала, год выпуска, возрастное ограничение, продолжительность, тип — сериал это или фильм. Связан с УСТРОЙСТВОМ. У одного УСТРОЙСТВА может быть несколько ФИЛЬМОВ/ СЕРИАЛОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 3. (ПК-3)

Сущность РЕЖИССЁРЫ содержит в себе информацию о режиссёрах: имя, фамилию. Связан с ФИЛЬМЫ/СЕРИАЛЫ при помощи сущности РЕЖИССЁР ФИЛЬМА/СЕРИАЛА. У одного РЕЖИССЁРА может быть несколько ФИЛЬМОВ/СЕРИАЛОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 4. (ПК-3)

Сущность АКТЁРЫ содержит в себе информацию об актёрах: имя, фамилию. Связан с ФИЛЬМЫ/СЕРИАЛЫ при пони сущности АКТЁРЫ ФИЛЬМА/СЕРИАЛА. У одного АКТЁРА может быть несколько ФИЛЬМОВ/ СЕРИАЛОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 5. (ПК-3)

Сущность СТРАНЫ содержит в себе информацию о странах: название название страны. Связан с ФИЛЬМЫ/СЕРИАЛЫ при помощи сущности СТРАНЫ ФИЛЬМА/СЕРИАЛА. У одного ФИЛЬМА/СЕРИАЛА может быть несколько СТРАН — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 6. (ПК-3)

Сущность ЖАНРЫ(в кинематографе) содержит в себе информацию о жанрах: название жанра .Связан с ФИЛЬМЫ/СЕРИАЛЫ при помощи сущности ЖАНРЫ ФИЛЬМА/СЕРИАЛА. У одного ФИЛЬМА/СЕРИАЛА может быть несколько ЖАНРОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 7. (ПК-3)

Сущность РЕЖИССЁР ФИЛЬМА/СЕРИАЛА является связующей сущностью для двух других сущностей: РЕЖИССЁРЫ и ФИЛЬМЫ/СЕРИАЛЫ, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 8. (ПК-3)

Сущность АКТЁРЫ ФИЛЬМА/СЕРИАЛА является связующей сущностью для двух других сущностей: АКТЁРЫ и ФИЛЬМЫ/СЕРИАЛЫ, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 9. (ПК-3)

Сущность СТРАНЫ ФИЛЬМА/СЕРИАЛА является связующей сущностью для двух других сущностей: СТРАНЫ и ФИЛЬМЫ/СЕРИАЛЫ, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 10. (ПК-3)

Сущность ЖАНРЫ ФИЛЬМА/СЕРИАЛА является связующей сущностью для двух других сущностей: ЖАНРЫ(в кинематографе) и ФИЛЬМЫ/СЕРИАЛЫ, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 11. (ПК-3)

Сущность МУЗЫКА содержит в себе информацию о музыкальной композиции: название композиции, дата выхода, продолжительность. Связана с сущностью УСТРОЙСТВА. У одного УСТРОЙСТВА может быть несколько МУЗЫКИ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 12. (ПК-3)

Сущность ЖАНРЫ(в музыке) содержит в себе информацию о музыкальных жанрах: название жанра. Связан с МУЗЫКОЙ при помощи сущности ЖАНРЫ В МУЗЫКЕ. У одной МУЗЫКИ может быть несколько ЖАНРОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 13. (ПК-3)

Сущность АВТОР содержит в себе информацию об авторах музыкальных композиций: автор. Связан с МУЗЫКОЙ при помощи сущности АВТОРЫ МУЗЫКИ. У одного автора может быть несколько МУЗЫК, также иногда у

одной МУЗЫКИ может быть НЕСКОЛЬКО АВТОРОВ — один-ко-многим. Подробное описание атрибутов сущности приведено в таблице 14. (ПК-3)

Сущность ЖАНРЫ МУЗЫКИ является связующей сущностью для двух других сущностей: ЖАНРЫ(музыки) и МУЗЫКА, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 15. (ПК-3)

Сущность АВТОРЫ МУЗЫКИ является связующей сущностью для двух других сущностей: АВТОРЫ и МУЗЫКА, образуя связь многие-ко-многим. Подробное описание атрибутов сущности приведено в таблице 16 [2]. (ПК-3)

3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

Таблицы реляционной БД должны быть нормализованы. Таблица соответствует третьей нормальной форме (3НФ), если для нее определен первичный ключ, все значения полей неделимы (или могут быть принят за неделимые), все неключевые атрибуты полностью функционально зависят от первичного ключа, но взаимонезависимы между собой [3].

Рис. 2 иллюстрирует функциональные зависимости сущности УСТРОЙСТВО. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица УСТРОЙСТВО находится в 3НФ [4].

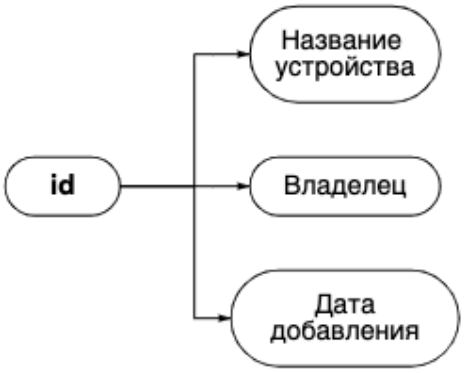

Рисунок 2 —Функциональные зависимости сущности УСТРОЙСТВО

Рис. 3 иллюстрирует функциональные зависимости сущности ВЛАДЕЛЕЦ УСТРОЙСТВА. Т.к. неключевые атрибуты взаимонезависимы,

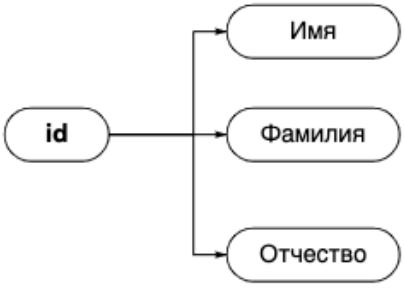

Рисунок 3 — Функциональные зависимости сущности ВЛАДЕЛЕЦ УСТРОЙСТВА

функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица ВЛАДЕЛЕЦ УСТРОЙСТВА находится в 3НФ [4].

Рис. 4 иллюстрирует функциональные зависимости сущности ФИЛЬМЫ/СЕРИАЛЫ. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица ФИЛЬМЫ/СЕРИАЛЫ находится в 3НФ [4].

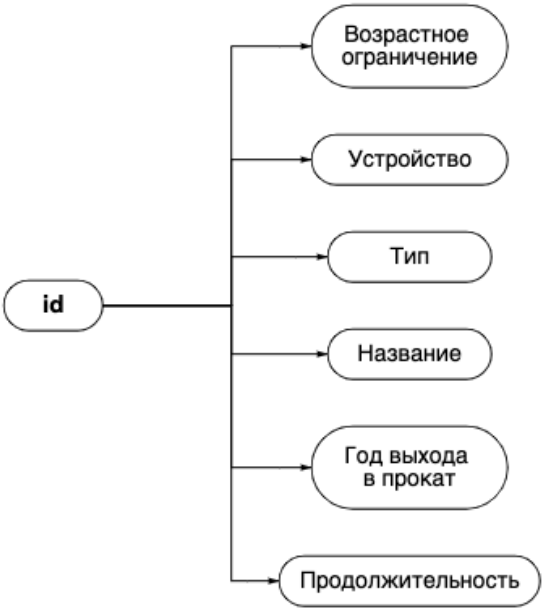

Рисунок 4 — Функциональные зависимости сущности ФИЛЬМЫ/СЕРИАЛЫ

Рис. 5 иллюстрирует функциональные зависимости сущности РЕЖИССЁРЫ. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица РЕЖИССЁРЫ находится в 3НФ [4].

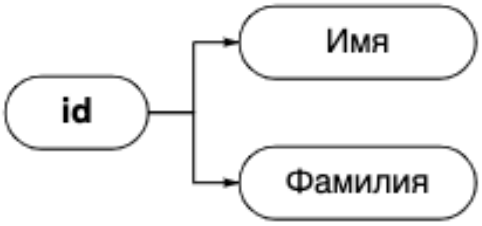

Рисунок 5 — Функциональные зависимости сущности РЕЖИССЁРЫ

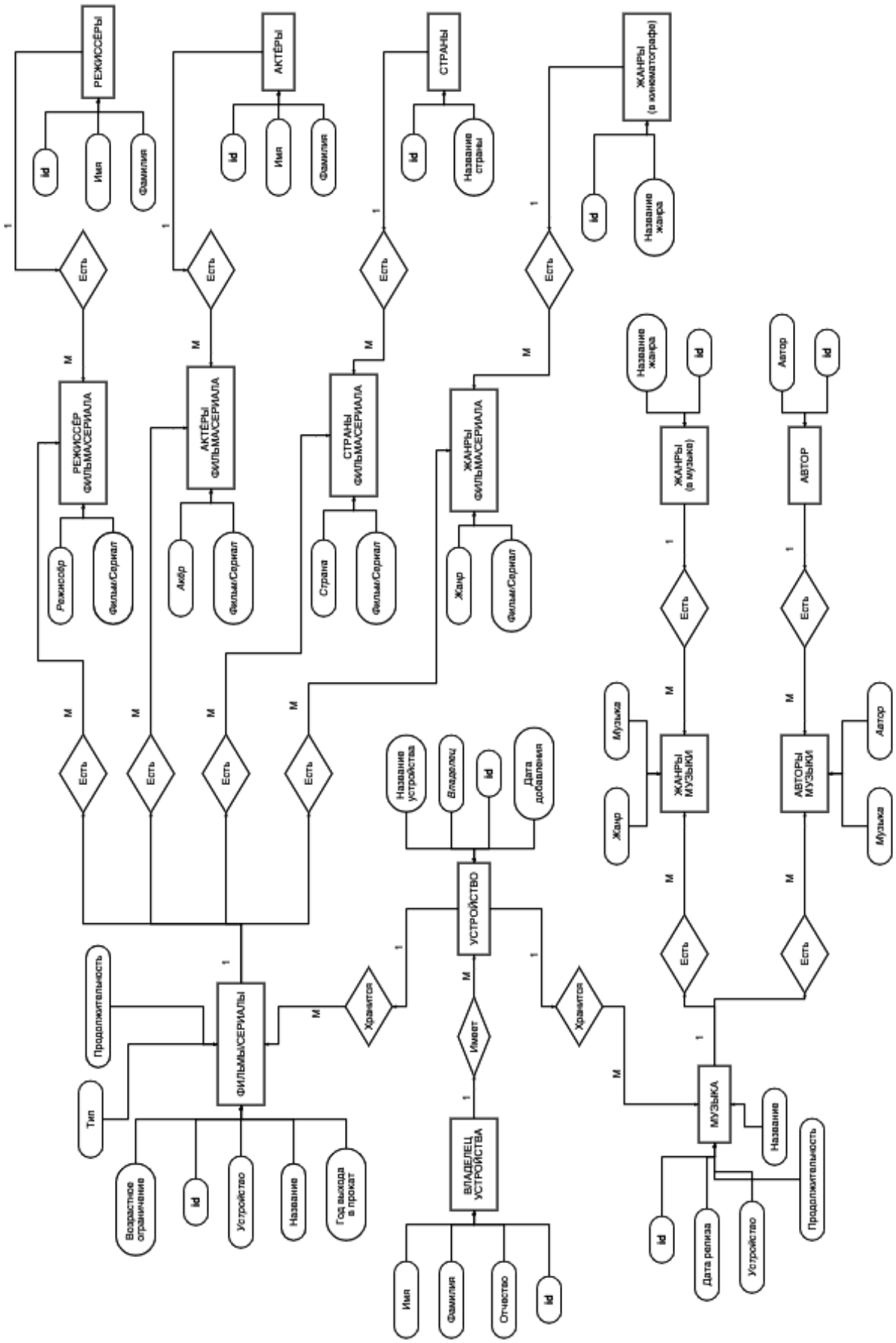

Рисунок 1 — Модель БД «сущность-связь» в нотации П.Чена

Рис. 6 иллюстрирует функциональные зависимости сущности АКТЁРЫ. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица АКТЁРЫ находится в 3НФ [4].

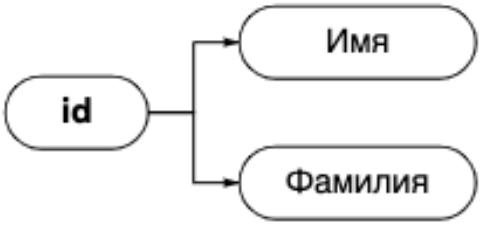

Рисунок 6 — Функциональные зависимости сущности АКТЁРЫ

Рис. 7 иллюстрирует функциональные зависимости сущности СТРАНЫ. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица СТРАНЫ находится в 3НФ [4].

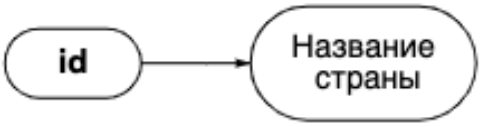

Рисунок 7 — Функциональные зависимости сущности СТРАНЫ

Рис. 8 иллюстрирует функциональные зависимости сущности ЖАНРЫ(в кинематографе). Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица ЖАНРЫ(в кинематографе) находится в 3НФ [4].

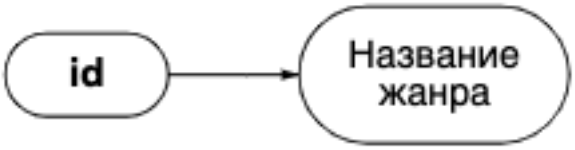

Рисунок 8 — Функциональные зависимости сущности ЖАНРЫ(в кинематографе)

Рис. 9 иллюстрирует функциональные зависимости сущности МУЗЫКА. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица МУЗЫКА находится в 3НФ [4].

Рисунок 9 — Функциональные зависимости сущности МУЗЫКА

Рис. 10 иллюстрирует функциональные зависимости сущности ЖАНРЫ(в музыке). Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица ЖАНРЫ(в музыке) находится в 3НФ [4].

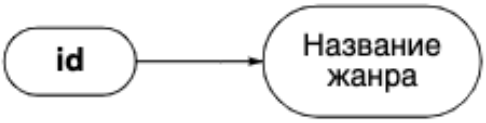

Рисунок 10 — Функциональные зависимости сущности ЖАНРЫ(в музыке)

Рис. 11 иллюстрирует функциональные зависимости сущности АВТОР. Т.к. неключевые атрибуты взаимонезависимы, функционально полностью зависят от первичного ключа, а значения полей неделимы, то таблица АВТОР находится в 3НФ [4].

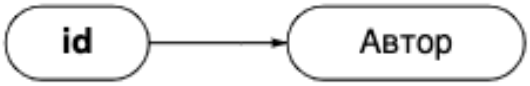

Рисунок 11 — Функциональные зависимости сущности АВТОР

4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

MySQL — свободная реляционная система управления базами данных. Как видно по концептуальному и логическому проектированиям, MySQL лучше, чем его аналоги, подходит для реализации базы данных домашней медиатеки. Во-первых, он прост в понимании, во-вторых, так как MySQL является реляционной базой данных, это позволяет упростить множество реализаций функций, процедур и триггеров.

Проанализировав концептуальное и логическое проектирования, были составлены таблицы с описанием атрибутов сущностей и примерами заполнения данных. (ПК-3)

Таблица 1. Сущность УСТРОЙСТВО (Device)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idDevice	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Name_of_device		Название или модель устройства	Строка переменной длины	Not null	Текст	
Date_add_to_base		Дата добавления	Дата	Not null	Даты формата: ГГГГ-ММ-ДД	
user_id	Внешний ключ для ВЛАДЕЛЕЦ УСТРОЙСТВА	Ссылка на владельца устройства	Целое число	Not null	Натуральные числа	

Таблица 2. Сущность ВЛАДЕЛЕЦ УСТРОЙСТВА (User_of_device)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idUser	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
first_name		Имя	Строка переменной длины	Not null	Только буквы русского алфавита	
second_name		Фамилия	Строка переменной длины	Not null	Только буквы русского алфавита	
surname		Отчество	Строка переменной длины		Только буквы русского алфавита	

Таблица 3. Сущность ФИЛЬМЫ/СЕРИАЛЫ (Movies_Serials)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idMovies_Serials	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Name_MS		Название фильма	Строка переменной длины	Not null	Текст	
Date_premiere		Год выхода фильма	Год		Дата в формате YYYY	
Age		Возрастное ограничение	Строка конечной длины		Возраст от 0-99 и знак «+»	

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Time_MS		Продолжительность	Время		Время в формате HH:MM:SS	
Type_MS		Фильм или сериал	Строка конечной длины	Not null	Кино, Сериал	
MS_device	Внешний ключ для УСТРОЙСТВА	Ссылка на устройство	Целое число	Not null	Натуральные числа	

Таблица 4. Сущность РЕЖИССЁРЫ(Director)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idDirector	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
first_name		Имя	Строка переменной длины	Not null	Только буквы русского алфавита	
second_name		Фамилия	Строка переменной длины	Not null	Только буквы русского алфавита	

Таблица 5. Сущность АКТЁРЫ (Actors)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idActors	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
first_name		Имя	Строка переменной длины	Not null	Только буквы русского алфавита	
second_name		Фамилия	Строка переменной длины	Not null	Только буквы русского алфавита	

Таблица 6. Сущность СТРАНЫ (Country)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idCountry	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Countrycol		Название страны	Строка переменной длины		Только буквы русского алфавита	

Таблица 7. Сущность ЖАНРЫ(в кинематографе) (Genre_of_Movie_Serials)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idGenre	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Genrecol		Название жанра	Строка переменной длины	Not null	Только буквы русского алфавита	

Таблица 8. Сущность РЕЖИССЁР ФИЛЬМА/СЕРИАЛА (Movies_Serials_has_Director)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Director_idDirector	Внешний ключ для РЕЖИССЁР	Ссылка на режиссёра	Целое число	Not null	Натуральные числа	
Movies_Serials_idMovies_Serials	Внешний ключ для ФИЛЬМЫ/СЕРИАЛЫ	Ссылка на фильм/сериал	Целое число	Not null	Натуральные числа	

Таблица 9. Сущность АКТЁРЫ ФИЛЬМА/СЕРИАЛА (Actors_has_Movies_Serials)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Actors_idActors	Внешний ключ для АКТЁРЫ	Ссылка на актёра	Целое число	Not null	Натуральные числа	
Movies_Serials_idMovies_Serials	Внешний ключ для ФИЛЬМЫ/СЕРИАЛЫ	Ссылка на фильм/сериал	Целое число	Not null	Натуральные числа	

Таблица 10. Сущность СТРАНЫ ФИЛЬМА/СЕРИАЛА (Movies_Serials_has_Country)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Movies_Serials_idMovies_Serials	Внешний ключ для ФИЛЬМЫ/СЕРИАЛЫ	Ссылка на фильм/сериал	Целое число	Not null	Натуральные числа	
Country_idCountry	Внешний ключ для СТРАНЫ	Ссылка на страну	Целое число	Not null	Натуральные числа	

Таблица 11. Сущность ЖАНРЫ ФИЛЬМА/СЕРИАЛА (Genre_has_Movies_Serials)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Genre_idGenre	Внешний ключ для ЖАНРА(в кинематографе)	Ссылка на жанр	Целое число	Not null	Натуральные числа	
Movies_Serials_idMovies_Serials	Внешний ключ для ФИЛЬМЫ/СЕРИАЛЫ	Ссылка на фильм/сериал	Целое число	Not null	Натуральные числа	

Таблица 12. Сущность МУЗЫКА (Music)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idMusic	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Name_M		Название композиции	Строка переменной длины	Not null	Текст	
Date_release		Дата релиза	Дата		Даты формата: ГГГГ-ММ-ДД	
M_device	Внешний ключ для УСТРОЙСТВА	Ссылка на устройство	Целое число	Not null	Натуральные числа	
Time_music		Продолжительность	Время		Время в формате HH:MM:SS	

Таблица 13. Сущность ЖАНРЫ(в музыке) (Genre_of_music)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idGenre_of_music	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Genre_of_musiccol		Название жанра	Строка переменной длины	Not null	Только буквы русского алфавита	

Таблица 14. Сущность АВТОР (Author)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
idAuthor	Первичный ключ	Идентификатор	Целое число	Not null Unique Auto Increment	Натуральные числа	
Authorcol		Автор композиции	Строка переменной длины	Not null	Только буквы русского алфавита	

Таблица 15. Сущность ЖАНРЫ МУЗЫКИ (Genre_of_music_has_Music)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Genre_of_music_idGenre_of_music	Внешний ключ для ЖАНРА(в музыке)	Ссылка на жанр	Целое число	Not null	Натуральные числа	
Music_id Music	Внешний ключ для МУЗЫКИ	Ссылка на музыку	Целое число	Not null	Натуральные числа	

Таблица 16. Сущность АВТОРЫ МУЗЫКИ (Author_has_Music)

Название	Ограничение	Описание	Тип данных	Ограничение на тип данных	Домен	Значение по умолчанию
Author_id Author	Внешний ключ для АВТОРА	Ссылка на автора	Целое число	Not null	Натуральные числа	
Music_id Music	Внешний ключ для МУЗЫКИ	Ссылка на музыку	Целое число	Not null	Натуральные числа	

Таблица 17. Пример данных для таблицы «Device»

idDevice	Name_of_device	Date_add_to_base	user_id
11	Macbook Air 2017	2019-09-01	1
12	Samsung Galaxy A9	2019-09-02	1
13	Honor Lite 20	2018-08-03	2
14	PC_Acer	2018-07-04	2
15	IPhone SE	2017-06-05	3
16	PC_MSI_NightBlade	2017-05-06	3

Таблица 18. Пример данных для таблицы «User_of_device»

idUser	first_name	second_name	surname
1	Дмитрий	Шайманов	Николаевич
2	Александр	Воронин	Владимирович
3	Михаил	Васильев	Ильич

Таблица 19. Пример данных для таблицы «Movies_Serials»

idMovies_Serials	Name_MS	Date_premiere	Age	Time_MS	Type_MS	MS_device
13	Учитель на замену	2011	16+	01:38:00	Кино	11
15	Во все тяжкие	2008	18+	00:47:00	Сериал	14

idMovies_Serials	Name_MS	Date_premiere	Age	Time_MS	Type_MS	MS_device
17	Карты, деньги, 2 ствола	1998	18+	01:47:00	Кино	16

Таблица 20. Пример данных для таблицы «Director»

idDirector	first_name	second_name
1	Тони	Кей
2	Мишель	МакЛарен
3	Гай	Ричи

Таблица 21. Пример данных для таблицы «Actors»

idActors	first_name	second_name
1	Винни	Джонс
2	Эдриан	Броуди
3	Сами	Гейл
4	Кристина	Хендрикс
5	Брайан	Крэнстон
6	Аарон	Пол
7	Боб	Оденкёрк
8	Джейсон	Стэйтхем

Таблица 22. Пример данных для таблицы «Country»

idCountry	Countrycol
1	США
2	Великобритания

Таблица 23. Пример данных для таблицы «Genre_of_Movie_Serials»

idGenre	Genrecol
1	Драма
2	Комедия
3	Триллер
4	Криминал
5	Боевик

Таблица 24. Пример данных для таблицы «Movies_Serials_has_Director»

Director_idDirector	Movies_Serials_idMovies_Serials
1	13
2	15
3	17

Таблица 25. Пример данных для таблицы «Actors_has_Movies_Serials»

Actors_idActors	Movies_Serials_idMovies_Serials
1	17
2	13
3	13
4	13
5	15
6	15
7	15
8	17

Таблица 26. Пример данных для таблицы «Movies_Serials_has_Country»

Movies_Serials_idMovies_Serials	Country_idCountry
13	1
15	1

Movies_Serials_idMovies_Serials	Country_idCountry
17	2

Таблица 27. Пример данных для таблицы «Genre_has_Movies_Serials»

Genre_idGenre	Movies_Serials_idMovies_Serials
1	13
1	15
2	17
3	15
4	15
4	17
5	17

Таблица 28. Пример данных для таблицы «Music»

idMusic	Name_M	Date_realese	M_device	Time_music
9	Take Five	1959-06-02	12	00:05:25
10	Шанс	2020-04-24	12	00:03:28
11	Look around	2012-01-25	13	00:03:28
12	Mr. Brightside	2003-09-29	13	00:03:42
13	Кукла колдуна	1999-03-01	15	00:03:23
14	Вокруг Шум	2008-05-15	15	00:03:33

Таблица 29. Пример данных для таблицы «Genre_of_music»

idGenre_of_music	Genre_of_musiccol
1	Джаз
2	Инди-рок
3	Альтернативный-рок
4	Панк
5	Хип-хоп

Таблица 30. Пример данных для таблицы «Author»

idAuthor	Authorcol
1	The Dave Brubeck Quartet
2	Дайте танк(!)
3	Red Hot Chili Peppers
4	The Killers
5	Король и Шут
6	Каста

Таблица 31. Пример данных для таблицы «Genre_of_music_has_Music»

Genre_of_music_idGenre_of_music	Music_idMusic
1	9
2	10
3	11
3	12
4	13
5	14

Таблица 32. Пример данных для таблицы «Author_has_Music»

Author_idAuthor	Music_idMusic
1	9
2	10
3	11
4	12
5	13
6	14

С помощью MySQL Workbench Model была создана ERR-модель БД домашней медиатеки в нотации IDEF1X для объектов БД и произведено прямое

проектирование БД на физическом уровне. Результат представлен на рисунке 18.

При помощи прямого инжиниринга были созданы, с соблюдением всех связей созданных в EER-модели ранее. Прямой инжиниринг — это технический термин, обозначающий процесс автоматизированного перевода логической модели БД в физическое представление [5]. (ПК-4)

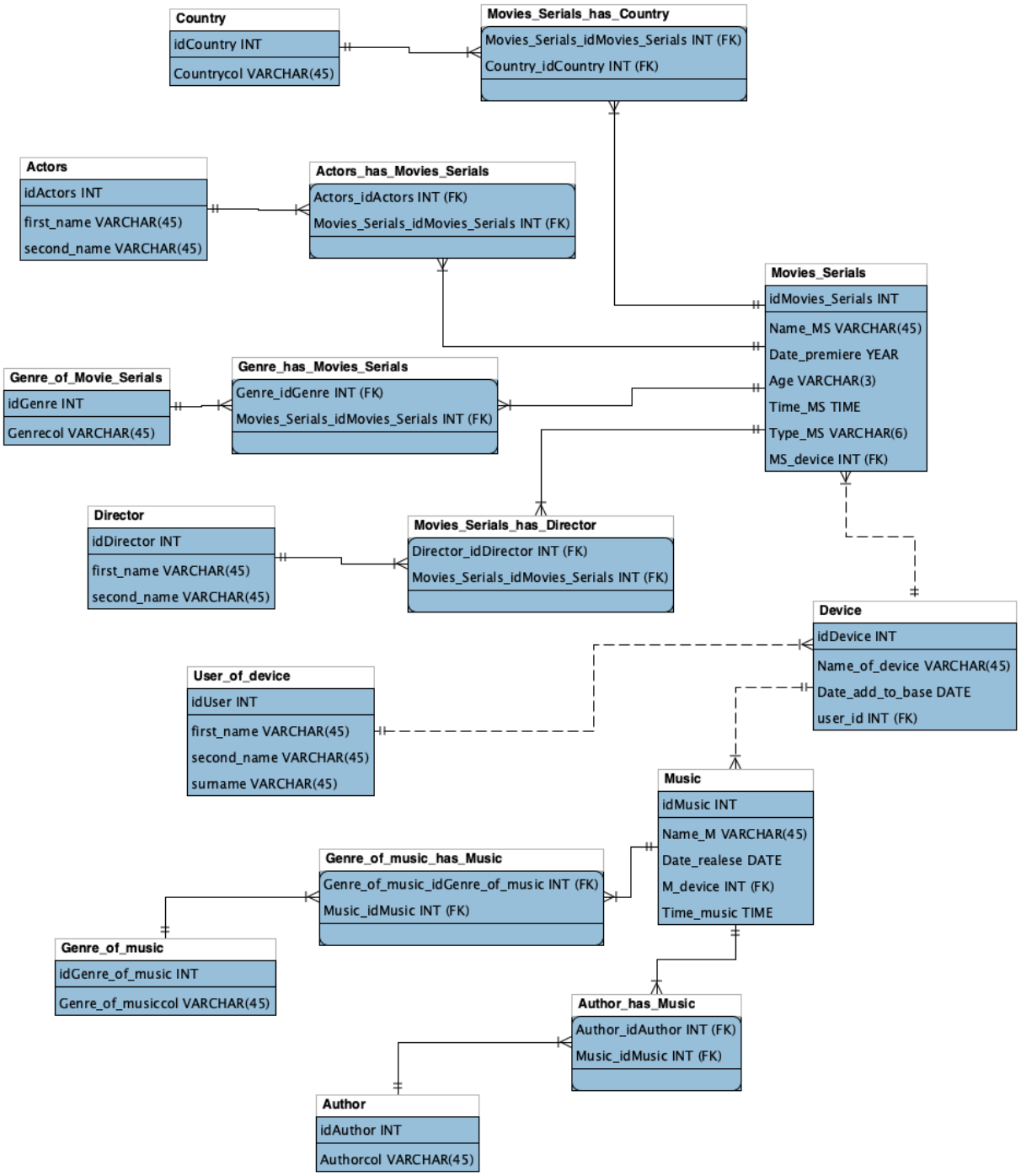

Рисунок 12 — EER-модель спроектированной БД

5. ЗАПРОСЫ

1. Фильмы, сделанные в США.

Запрос включает в себя ряд сущностей: ФИЛЬМЫ/СЕРИАЛЫ, АКТЁРЫ, АКТЁРЫ ФИЛЬМА/СЕРИАЛА, РЕЖИССЁРЫ, РЕЖИССЁР ФИЛЬМА/СЕРИАЛА, СТРАНЫ, СТРАНЫ ФИЛЬМА/СЕРИАЛА, ЖАНРЫ(в кинематографе), ЖАНРЫ ФИЛЬМА/СЕРИАЛА. В данном запросе используется встроенная функция group_concat(), которая соединяет значения из определённого столбца в одну строчку [6,7].

Код запроса

```
select    Name_MS as 'Название фильма',
                        Date_premiere as 'Год',
          Time_MS as 'Продолжительность фильма',
          group_concat(distinct Countrycol SEPARATOR ', ') as 'Страна',
          group_concat(distinct Genrecol SEPARATOR ', ') as 'Жанр',
                        group_concat(distinct concat(Actors.first_name, ' ', Actors.second_name) SEPARATOR ', ') as 'Актёр',
          concat(Director.first_name, ' ', Director.second_name) as 'Режиссёр',
          Age as 'Возрастное ограничение'
        from Movies_Serials, Actors, Actors_has_Movies_Serials, Director, Movies_Serials_has_Director, Country,
Movies_Serials_has_Country, Genre_of_Movie_Serials, Genre_has_Movies_Serials where Type_MS = 'Кино' and Countrycol = 'США' and
                                        (Actors.idActors = Actors_has_Movies_Serials.Actors_idActors and
Actors_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
                (Movies_Serials_has_Director.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials and Director.idDirector
= Movies_Serials_has_Director.Director_idDirector) and
                (Country.idCountry = Movies_Serials_has_Country.Country_idCountry and
Movies_Serials_has_Country.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
                (Genre_of_Movie_Serials.idGenre = Genre_has_Movies_Serials.Genre_idGenre and
Genre_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials)
                group by Countrycol, `Режиссёр`, idMovies_Serials;
```

2. Фильмы в порядке убывания их продолжительности.

Запрос включает в себя ряд сущностей: ФИЛЬМЫ/СЕРИАЛЫ, АКТЁРЫ, АКТЁРЫ ФИЛЬМА/СЕРИАЛА, РЕЖИССЁРЫ, РЕЖИССЁР ФИЛЬМА/СЕРИАЛА, СТРАНЫ, СТРАНЫ ФИЛЬМА/СЕРИАЛА, ЖАНРЫ(в кинематографе), ЖАНРЫ ФИЛЬМА/СЕРИАЛА. В данном запросе используется встроенная функция group_concat(), которая соединяет значения из определённого столбца в одну строчку [6,7].

Код запроса

```
select    Name_MS as 'Название фильма',
                        Date_premiere as 'Год',
          Time_MS as 'Продолжительность фильма',
          Countrycol as 'Страна',
          group_concat(distinct genrecol SEPARATOR ', ') as 'Жанр',
```

```
                    group_concat(distinct concat(Actors.first_name, ' ', Actors.second_name) SEPARATOR ', ') as 'Актёр',
        concat(Director.first_name, ' ', Director.second_name) as 'Режиссёр',
        Age as 'Возрастное ограничение'
    from Movies_Serials, Actors, Actors_has_Movies_Serials, Director, Movies_Serials_has_Director, Country,
Movies_Serials_has_Country, Genre_of_Movie_Serials, Genre_has_Movies_Serials where Type_MS = 'Кино' and
                                        (Actors.idActors = Actors_has_Movies_Serials.Actors_idActors and
Actors_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
            (Movies_Serials_has_Director.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials and Director.idDirector
= Movies_Serials_has_Director.Director_idDirector) and
            (Country.idCountry = Movies_Serials_has_Country.Country_idCountry and
Movies_Serials_has_Country.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
            (Genre_of_Movie_Serials.idGenre = Genre_has_Movies_Serials.Genre_idGenre and
Genre_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials)
            group by Countrycol, `Режиссёр`, idMovies_Serials
            order by Time_MS desc;
```

3. Фильмы и сериалы, у которых больше одного жанра.

Запрос включает в себя ряд сущностей: ФИЛЬМЫ/СЕРИАЛЫ, АКТЁРЫ, АКТЁРЫ ФИЛЬМА/СЕРИАЛА, РЕЖИССЁРЫ, РЕЖИССЁР ФИЛЬМА/СЕРИАЛА, СТРАНЫ, СТРАНЫ ФИЛЬМА/СЕРИАЛА, ЖАНРЫ(в кинематографе), ЖАНРЫ ФИЛЬМА/СЕРИАЛА. В данном запросе используется встроенная функция group_concat(), которая соединяет значения из определённого столбца в одну строчку [6,7].

Код запроса

```
select      Name_MS as 'Название фильма',
                    Date_premiere as 'Год',
    Time_MS as 'Продолжительность фильма',
    Countrycol as 'Страна',
    group_concat(distinct genrecol SEPARATOR ', ') as 'Жанр',
                    group_concat(distinct concat(Actors.first_name, ' ', Actors.second_name) SEPARATOR ', ') as 'Актёр',
    concat(Director.first_name, ' ', Director.second_name) as 'Режиссёр',
    Age as 'Возрастное ограничение'
    from Movies_Serials, Actors, Actors_has_Movies_Serials, Director, Movies_Serials_has_Director, Country,
Movies_Serials_has_Country, Genre_of_Movie_Serials, Genre_has_Movies_Serials where
                                        (Actors.idActors = Actors_has_Movies_Serials.Actors_idActors and
Actors_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
            (Movies_Serials_has_Director.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials and Director.idDirector
= Movies_Serials_has_Director.Director_idDirector) and
            (Country.idCountry = Movies_Serials_has_Country.Country_idCountry and
Movies_Serials_has_Country.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials) and
            (Genre_of_Movie_Serials.idGenre = Genre_has_Movies_Serials.Genre_idGenre and
Genre_has_Movies_Serials.Movies_Serials_idMovies_Serials = Movies_Serials.idMovies_Serials)
            group by Countrycol, `Режиссёр`, idMovies_Serials
            having (select count(*) from Genre_has_Movies_Serials where Movies_Serials_idMovies_Serials = idMovies_Serials) >=
2;
```

4. Музыка, начиная с самого позднего релиза.

Запрос включает в себя такие сущности как МУЗЫКА, АВТОР, АВТОРЫ МУЗЫКИ, ЖАНРЫ(в музыке), ЖАНРЫ МУЗЫКИ. В данном запросе

используется встроенная функция group_concat(), которая соединяет значения из определённого столбца в одну строчку [6,7].

Код запроса

```
select	Name_M as 'Название трека',
		group_concat(distinct Authorcol SEPARATOR ', ') as 'Автор',
	group_concat(distinct Genre_of_musiccol SEPARATOR ', ') as 'Жанр',
	Date_realese as 'Дата выхода'
		from Music, Author, Author_has_Music, Genre_of_music, Genre_of_music_has_Music where
		(Author.idAuthor = Author_has_Music.Author_idAuthor and Author_has_Music.Music_idMusic = idMusic) and
	(Genre_of_music.idGenre_of_music = Genre_of_music_has_Music.Genre_of_music_idGenre_of_music and
Genre_of_music_has_Music.Music_idMusic = idMusic)
	group by idMusic, Genre_of_musiccol, Authorcol
	order by Date_realese;
```

5. Музыка, где в жанре имеется слово «рок».

Запрос включает в себя такие сущности как МУЗЫКА, АВТОР, АВТОРЫ МУЗЫКИ, ЖАНРЫ(в музыке), ЖАНРЫ МУЗЫКИ. В данном запросе используется встроенная функция group_concat(), которая соединяет значения из определённого столбца в одну строчку [6,7].

Код запроса

```
select	Name_M as 'Название трека',
		group_concat(distinct Authorcol SEPARATOR ', ') as 'Автор',
	group_concat(distinct Genre_of_musiccol SEPARATOR ', ') as 'Жанр',
	Date_realese as 'Дата выхода'
		from Music, Author, Author_has_Music, Genre_of_music, Genre_of_music_has_Music where Genre_of_musiccol like
'%рок%' and
		(Author.idAuthor = Author_has_Music.Author_idAuthor and Author_has_Music.Music_idMusic = idMusic) and
	(Genre_of_music.idGenre_of_music = Genre_of_music_has_Music.Genre_of_music_idGenre_of_music and
Genre_of_music_has_Music.Music_idMusic = idMusic)
	group by idMusic, Genre_of_musiccol, Authorcol;
```

6. ОПИСАНИЕ ОБЪЕКТОВ ПРОЦЕДУРНОГО SQL

6.1 Функции

1. Функция для подсчёта в скольких фильмах снялся актёр.

Синтаксис функции

count_actor_in_movies(Actor_name)

Аргументы функции

Actor_name — полное имя актёра, тип данных — VARCHAR(45).

Результат

Значение количества фильмов, в которых снялся актёр, тип данных — INT.

Пример результатов выполнения

Пример выполнения приведён на рис. 13.

Код

```
CREATE DEFINER=`root`@`localhost` FUNCTION `count_actor_in_movies`(Actor_name varchar(45)) RETURNS int
    DETERMINISTIC
BEGIN
        declare count_of_actors int;
  declare id_actor int;
  select idActors into id_actor from Actors where concat(Actors.first_name, ' ', Actors.second_name) = Actor_name;
  select count(*) into count_of_actors from Actors_has_Movies_Serials where Actors_idActors = id_actor;
RETURN count_of_actors;
END
```

Описание алгоритма

1) Находится ID актёра с введёнными именем и фамилией;

```
1 •  select time_in_device(13);
2 •  select time_in_device(14);
3 •  select time_in_device(15);
```

2)

time_in_device(13)	time_in_device(14)	time_in_device(15)
00:07:10	00:47:00	00:06:56

Считается

Рисунок 14 —Пример выполнения функции time_in_device(id)

количество фильмов по ID актёра в связанной таблице;

Для того, чтобы найти ID актёра по введённым имени и фамилии, был произведён запрос с выбором только тех имён и фамилий актёров, которые совпадают с введёнными значениями. Так как в таблице Actors имена и фамилии записаны в разных колонках, то была использована встроенная функця concat(), которая соединит эти два значения. После того, как ID актёра было найдено, производоится следующий запрос по связанной таблице Actors_has_Movies_Serials с выбором только тех фильмов, где снимался актёр с найденным ID, производя их подсчёт при помощи встроенной функции count() [8].

2. Функция для подсчёта общего времени продолжительности времени файлов на устройстве.

Синтаксис функции

time_in_device(id)

Аргументы функции

id — id устройства, тип данных — VARCHAR(45).

Результат

Значение количества времени на устройстве, тип данных — TIME.

Пример результатов выполнения

Пример выполнения приведён на рис. 14.

Код

```
CREATE DEFINER=`root`@`localhost` FUNCTION `time_in_device`(id int) RETURNS time
    DETERMINISTIC
BEGIN
        declare sum_of_movie time;
    declare sum_of_music time;
```

```
  declare result time;

  select SEC_TO_TIME(sum(TIME_TO_SEC(Time_MS))) into sum_of_movie from Movies_Serials where id = MS_device;
  select SEC_TO_TIME(sum(TIME_TO_SEC(Time_music))) into sum_of_music from Music where id = M_device;

  if sum_of_movie is null then set sum_of_movie = '00:00:00'; end if;
  if sum_of_music is null then set sum_of_music = '00:00:00'; end if;
  set result = sum_of_movie + sum_of_music;
RETURN result;
END
```

Описание алгоритма

1) Находится время в таблице Movies_Serials;

2) Находится время в таблице Music;

3) Проводится проверка: если одно из найденных времён будет NULL, то присвоить ему значение ’00:00:00’;

4) Суммируются полученные времена.

Для того, чтобы найти время в таблице Movies_Serials, был произведён по ней запрос, в котором используется встроенная функцию sum(), которая суммирует значения в выбранном столбце, с условием равенства ID устройства с введённым значением. Но так просто сложить значения типа TIME не получится, поэтому используем две встроенные функции: TIME_TO_SEC() и SEC_TO_TIME(). Первая переводит значение типа TIME в секунды, а вторая

```
1 •  select count_actor_in_movies('Винни Джонс');
2 •  select count_actor_in_movies('Эдриан Броуди');
3 •  select count_actor_in_movies('Аарон Пол');
```

наоборот — из секунд в значение типа TIME. Аналогично с таблицей Music. Так как не на всех устройствах может быть фильм/сериал или музыка, следовательно одна из

```
count_actor_in_movies('Винни Джонс')
▶ 1

count_actor_in_movies('Эдриан Броуди')
▶ 1

count_actor_in_movies('Аарон Пол')
▶ 1
```

Рисунок 13 — Пример выполнения функции count_actor_in_movies(Actor_name)

полученных сумм может принять значение NULL — пустое, в результате чего

конечный ответ будет тоже NULL, поэтому была произведена проверка при помощи конструкции IF … THEN, в которой проверяется каждая из сумм на равенство NULL. Если значение равно NULL, то сумме присваиваем значение ’00:00:00’. Так как значение «нулевое», то на конечный ответ это не отразиться. Полученные в ходе запросов суммы и получаем ответ [8, 9].

6.2 Процедуры

1. Добавление пользователя и его устройства.

Синтаксис

add_user_with_device(new_first_name, new_second_name, new_surname, new_Name_of_device)

Входные параметры

new_first_name - имя пользователя, тип данных VARCHAR(45);

new_second_name — фамилия пользователя, тип данных VARCHAR(45);

new_surname — отчество пользователя, тип данных VARCHAR(45);

new_Name_of_device — наименование устройства, тип данных VARCHAR(45);

Пример вызова процедуры

Пример выполнения приведён на рис. 15.

Код

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `add_user_with_device`(new_first_name varchar(45), new_second_name varchar(45), new_surname varchar(45), new_Name_of_device varchar(45))
	BEGIN
		declare buf int;
		insert into User_of_device(first_name, second_name, surname) values (new_first_name, new_second_name, new_surname);
		select idUser into buf from User_of_device where first_name = new_first_name and second_name = new_second_name and surname = new_surname;
		insert into Device(Name_of_device, Date_add_to_base, user_id) values (new_Name_of_device, CURRENT_DATE(), buf);
	END
```

```
1 •  call Mediateka.add_user_with_device('Алекнсадр', 'Александров', 'Александрович', 'Macbook Pro');
```

idDevice	Name_of_device	Date_add_t...	user_id
11	Macbook Air 2017	2019-09-01	1
12	Samsung Galaxy A9	2019-09-02	1
13	Honor Lite 20	2018-08-03	2
14	PC_Acer	2018-07-04	2
15	IPhone SE	2017-06-05	3
16	PC_MSI_NightBlade	2017-05-06	3
18	Macbook Pro	2021-06-03	5

idUser	first_name	second_name	surname
1	Дмитрий	Шайманов	Николаевич
2	Александр	Воронин	Владимирович
3	Михаил	Васильев	Ильич
5	Алекнсадр	Александров	Александрович
NULL	NULL	NULL	NULL

Рисунок 15 —Пример вызова процедуры add_user_with_device(new_first_name varchar(45), new_second_name varchar(45), new_surname varchar(45), new_Name_of_device varchar(45))

Описание алгоритма

1) Добавляется имя, фамилия, отчество в таблицу User_of_device;

2) Записывается ID нового пользователя в буферную переменную;

3) Добавляется название устройства, дату добавления и ID нового пользователя в таблицу Device.

При помощи команды insert были добавлены введённые значения имени, фамилии и отчества в таблицу User_of_device. Затем при помощи запроса было найдено ID нового пользователя, оно не было введено так как у него по умолчанию стоит autoincrement. Потом снова при помощи команды insert было добавлено значения названия устройства, даты добавления, при помощи встроенной функции now(), которая передаёт дату и время на данный момент времени и ID нового пользователя [9, 10]. (ПК-3)

2. Подсчёт сколько всего времени занимают файлы в медиатеке.

Синтаксис

all_time_in_mediateka()

Входные параметры

Нет

Пример вызова процедуры

Пример выполнения приведён на рис. 16.

Код

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `all_time_in_mediateka`()
	BEGIN
		select sec_to_time(sum(TIME_TO_SEC(time_in_device(idDevice)))) as 'Продолжительность фильмов \nи музыки во всей медиатеке' from Device;
	END
```

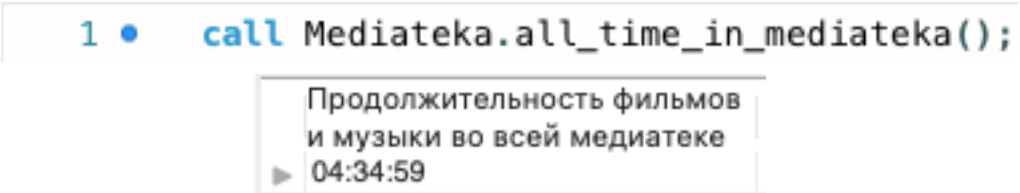

Рисунок 16 — Пример вызова процедуры all_time_in_mediateka()

Описание алгоритма

1) Производится запрос, в которой используется функция time_in_device,

по таблице Device.

Производится запрос по таблице Device, в которой используется функция time_in_device, которой передаётся значение ID устройства, к которой применяется встроенную функцию sum(). Но чтобы правильно суммировать значения типа TIME, нужно сначала перевести их в секунды, а потом всю сумму перевести обратно в значение типа TIME. Делается это при помощи встроенных функций TIME_TO_SEC() и SEC_TO_TIME() соответсвенно [9].

3. Удаление пользователя и связанных с ним устройств.

Синтаксис

delete_user_with_device(id)

Входные параметры

id — id пользователя, тип данных — INT.

Пример вызова процедуры

Пример выполнения приведён на рис. 17.

Код

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `delete_user_with_device`(id int)
	BEGIN
		delete from device where user_id = id;
	 delete from User_of_device where idUser = id;
	END
```

idDevice	Name_of_device	Date_add_t...	user_id
11	Macbook Air 2017	2019-09-01	1
12	Samsung Galaxy A9	2019-09-02	1
13	Honor Lite 20	2018-08-03	2
14	PC_Acer	2018-07-04	2
15	IPhone SE	2017-06-05	3
16	PC_MSI_NightBlade	2017-05-06	3
18	Macbook Pro	2021-06-03	5

idUser	first_name	second_name	surname
1	Дмитрий	Шайманов	Николаевич
2	Александр	Воронин	Владимирович
3	Михаил	Васильев	Ильич
5	Алекнсадр	Александров	Александрович
NULL	NULL	NULL	NULL

```
1 •  call Mediateka.delete_user_with_device(5);
```

idUser	first_name	second_name	surname
1	Дмитрий	Шайманов	Николаевич
2	Александр	Воронин	Владимирович
3	Михаил	Васильев	Ильич

idDevice	Name_of_device	Date_add_t...	user_id
11	Macbook Air 2017	2019-09-01	1
12	Samsung Galaxy A9	2019-09-02	1
13	Honor Lite 20	2018-08-03	2
14	PC_Acer	2018-07-04	2
15	IPhone SE	2017-06-05	3
16	PC_MSI_NightBlade	2017-05-06	3

Рисунок 17 — Пример вызова процедуры delete_user_with_device(id int)

Описание алгоритма

1) Удаляется запись по ID пользователя из таблицы Device;

2) Удаляется запись по ID пользователя из таблицы User_of_device.

Сначала удаляется запись в таблице Device при помощи команды delete по ведённому id пользователя. Так как, таблица Device связана с таблицей User_of_device при помощи внешнего ключа user_id, значит можно при помощи команды delete удалить запись в таблице User_of_device также при помощи введённого id пользователя.

4. Подсчёт количества фильмов/сериалов, в которых снялись актёры.

Синтаксис

how_many_actors_have_films()

Входные параметры

Нет

Пример вызова процедуры

Пример выполнения приведён на рис. 18.

Код

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `how_many_actors_have_films`()
	BEGIN
		select	concat(first_name,' ',second_name) as 'Актёр',
			count_actor_in_movies(concat(first_name,' ',second_name)) as 'Количество фильмов'
			from Actors;
	END
```

```
1 •	call Mediateka.how_many_actors_have_films();
```

Актёр	Количество фильмов
Винни Джонс	1
Эдриан Броуди	1
Сами Гейл	1
Кристина Хендрикс	1
Брайан Крэнстон	1
Аарон Пол	1
Боб Оденкёрк	1
Джейсон Стэйтхем	1

Рисунок 18 — Пример вызова процедуры how_many_actors_have_films()

Описание алгоритма

1) Производится запрос, в которой используется функция count_actor_in_movie, по таблице Actors;

Производится запрос по таблице Actors в котором, сперва при помощи встроенной функции concat() объединяются имена и фамилии актёров, затем используется функция count_actor_in_movie(), в которую в качестве аргумента передаётся объединённая строка, при помощи встроенной функции concat(), имени и фамилии актёра.

5. Подсчёт сколько всего времени занимают файлы на каждом устройстве.

Синтаксис

time_on_every_devices()

Входные параметры

Нет

Пример вызова процедуры

Пример выполнения приведён на рис. 19.

Код

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `time_on_every_devices`()
        BEGIN
         select    concat(first_name,' ', second_name, ' ',surname) as 'Владелец',
                                    Name_of_device as 'Устройство',
            time_in_device(idDevice) as 'Время'
                        from User_of_device, Device where user_id=idUser;
        END
```

```
1 •   call Mediateka.time_on_every_devices();
```

Владелец	Устройство	Время
Дмитрий Шайманов Николаевич	Macbook Air 2017	01:38:00
Дмитрий Шайманов Николаевич	Samsung Galaxy A9	00:08:53
Александр Воронин Владимирович	Honor Lite 20	00:07:10
Александр Воронин Владимирович	PC_Acer	00:47:00
Михаил Васильев Ильич	IPhone SE	00:06:56
Михаил Васильев Ильич	PC_MSI_NightBlade	01:47:00

Рисунок 19 — Пример вызова процедуры time_on_every_devices()

Описание алгоритма

1) Производится запрос, в котором используется функция time_in_device,

по таблице Device;

Был произведён запрос по таблице Device, используется функция time_in_device, которой передаётся значение ID устройства.

7. ТРИГГЕРЫ

1. Type_MS_before_insert

Триггер для проверки ввода данных в колонку Type_MS в таблицу Movies_Serials.

Алгоритм выполнения

1) Проверка введённого значения на соответствие словам Кино или Сериал;

2) Если введённые значения не Кино и не Сериал, то выводит ошибку;

Данный триггер предназначен для проверки ввода данных в колонку Type_MS таблицы Movies_Serials, чтобы в дальнейшем было легче производить поиск либо же по фильмам, либо же по сериалам.

Код

```
CREATE DEFINER = CURRENT_USER TRIGGER `Mediateka`.`Movies_Serials` BEFORE INSERT ON `Movies_Serials` FOR EACH ROW
BEGIN
	if new.Type_MS != 'Кино' or new.Type_MS != 'Сериал' then
		SIGNAL SQLSTATE '45000'
  SET MESSAGE_TEXT = 'The value in the Type_MS field must be "Кино" or "Сериал"';
	end if;
END
```

2. Date_realese_before_insert

Триггер для проверки ввода данных в колонку Date_realese в таблицу Music.

Алгоритм выполнения

1) Сравнение введённой даты с сегодняшней;

2) Если введённые значения больше, чем сегодняшняя дата, то выводит ошибку;

Данный триггер предназначен для проверки ввода данных в колонку Date_realese таблицы Music. Если введённая дата больше, чем сегодняшняя, то выводится ошибка, так как дата релиза музыкальной композиции не может быть больше, чем сегодняшняя.

Код

```
CREATE DEFINER = CURRENT_USER TRIGGER `Mediateka`.`Movies_Serials` BEFORE INSERT ON `Movies_Serials` FOR EACH ROW
BEGIN
        if new.Type_MS != 'Кино' or new.Type_MS != 'Сериал' then
                SIGNAL SQLSTATE '45000'
  SET MESSAGE_TEXT = 'The value in the Type_MS field must be "Кино" or "Сериал"';
        end if;
END
```

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы была спроектирована функционирующая база данных домашней медиатеки. Данная база данных имеет частичный функционал: функции для подсчёта в скольких фильмах снялся актёр, для подсчёта общего времени продолжительности файлов на устройстве, процедуры добавления пользователя и его устройства, подсчёта общей времени продолжительности файлов в медиатеке, удаление пользователя и связанных с ним устройств, подсчёта общей продолжительности файлов на каждом устройстве, подсчёта количества фильмов/сериалов, в которых снялись актёры.

В рамках курсовой работы было создано пять процедур, ориентированных на функционал домашней медиатеки, две функции, два триггера, которые используются в процедурах, и одно представление. В процессе работы над базой данных были закреплены навыки разработки реляционных баз данных: написание функций, процедур, триггеров, представлений, приведение БД к 1NF, 2NF, 3NF, выявлять функциональные связи атрибутов таблиц, а также были приобретены навыки работы с MySQL Workbench и MySQL Model.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Стружкин, Н. П. Базы данных: проектирование : учебник для вузов / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2020. — 477 с. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/450165 (дата обращения: 04.06.2021).
2. Осипов, Д. Л. Технологии проектирования баз данных / Д. Л. Осипов. — Москва : ДМК Пресс, 2019. — 498 с. — ISBN 978-5-97060-737-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131692 (дата обращения: 04.06.2021). — Режим доступа: для авториз. Пользователей.
3. Стружкин, Н. П. Базы данных: проектирование. Практикум : учебное пособие для вузов / Н. П. Стружкин, В. В. Годин. — Москва : Издательство Юрайт, 2020. — 291 с. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/451246 (дата обращения: 04.06.2021).
4. Ульман, Л. MySQL / Л. Ульман. — Москва : ДМК Пресс, 2008. — 352 с. — ISBN 5-94074-229-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/1241 (дата обращения: 04.06.2021). — Режим доступа: для авториз. пользователей.
5. Мартишин, С. А. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench : учеб. пособие / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. — М. : ИД «ФОРУМ» : ИНФРА-М, 2019. – 160 с. – Текст : электронный. – URL: https://znanium.com/catalog/product/1007949 (дата обращения: 04.06.2021). – Режим доступа: по подписке.
6. Мартишин, С. А. Базы данных. Практическое применение СУБД SQL- и NoSQL-типа для проектирования информационных систем : учебное пособие / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. — Москва : ФОРУМ : ИНФРА-М, 2021. — 368 с. — Текст : электронный. – URL: https://znanium.com/catalog/product/1243192 (дата обращения: 04.06.2021). – Режим доступа: по подписке.

7. GROUP_CONCAT( ) в Mysql. // WeBinstruments [сайт]. — URL: https://webi.ru/webi_articles/8_14_f.html (дата обращения: 04.06.2021).
8. MySQL 8.0 Reference Manual // MySQL [сайт]. – URL: https://dev.mysql.com/doc/refman/8.0/en/ (дата обращения: 04.06.2021).
9. Date and Time Functions. // MySQL [сайт]. – URL: https://dev.mysql.com/doc/refman/8.0/en/date-and-time-functions.html (дата обращения: 04.06.2021).
10. MySQL 8 для больших данных / Ш. Чаллавала, Д. Лакхатария, Ч. Мехта, К. Патель ; перевод с английского А. В. Логунова. — Москва : ДМК Пресс, 2018. — 226 с. — ISBN 978-5-97060-653-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131684 (дата обращения: 04.06.2021). — Режим доступа: для авториз. пользователей.