**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №2**

**по дисциплине «Базы данных»**

**Тема: Реализация базы данных в СУБД MySQL**

| Студент гр. 9381 |  | Матвеев А. Н. |
| --- | --- | --- |
| Преподаватель |  | Заславский М. М. |

Санкт-Петербург

2021

## Цель работы.

Реализация базы данных в СУБД MySQL.

**Задание.**

Вариант 10.

Пусть требуется создать программную систему, предназначенную для работников справочной службы кинотеатров города. Такая система должна обеспечивать хранение сведений о кинотеатрах города, о фильмах, которые в них демонстрируются, о сеансах и билетах на эти сеансы. Сведения о кинотеатре — это его название, район города, где расположен кинотеатр, категория, вместимость. Сведения о фильме — это название фильма, режиссер, оператор, актеры, сыгравшие главные роли, жанр; производство, наличие призов кинофестивалей, продолжительность сеанса, кадр из фильма для рекламы. Кроме того, должна храниться информация о репертуаре кинотеатров на месяц, то есть о том какие фильмы, когда и где демонстрируются, о ценах на билеты и о количестве свободных мест на тот или иной сеанс. На разных сеансах в одном кинотеатре могут идти разные фильмы, а если в кинотеатре несколько залов, то и на одном. Кинотеатр может ввести новый фильм в репертуар или убрать фильм из репертуара. Работник справочной службы может корректировать перечень фильмов, находящихся в прокате – добавлять новые фильмы и снимать с проката, а также перечень кинотеатров, поскольку кинотеатры могут открываться или закрываться, причем иногда временно, например, на ремонт. Цена билета определяется прокатной стоимостью копии фильма, сеансом и категорией кинотеатра. Справочной службе могут потребоваться следующие сведения о текущем состоянии проката фильмов в городе:

* + - * Репертуар кинотеатра?
      * Адрес и район кинотеатра ?
      * Число свободных мест на данный сеанс в указанном кинотеатре?
      * Цена билетов на данный сеанс в указанном кинотеатре?
      * Жанр, производство и режиссер данного фильма ?
      * Какие фильмы имеют награды, когда и в каких кинотеатрах они демонстрируются?
      * В каких кинотеатрах в указанный день на указанных сеансах демонстрируется комедия?

**Выполнение работы.**

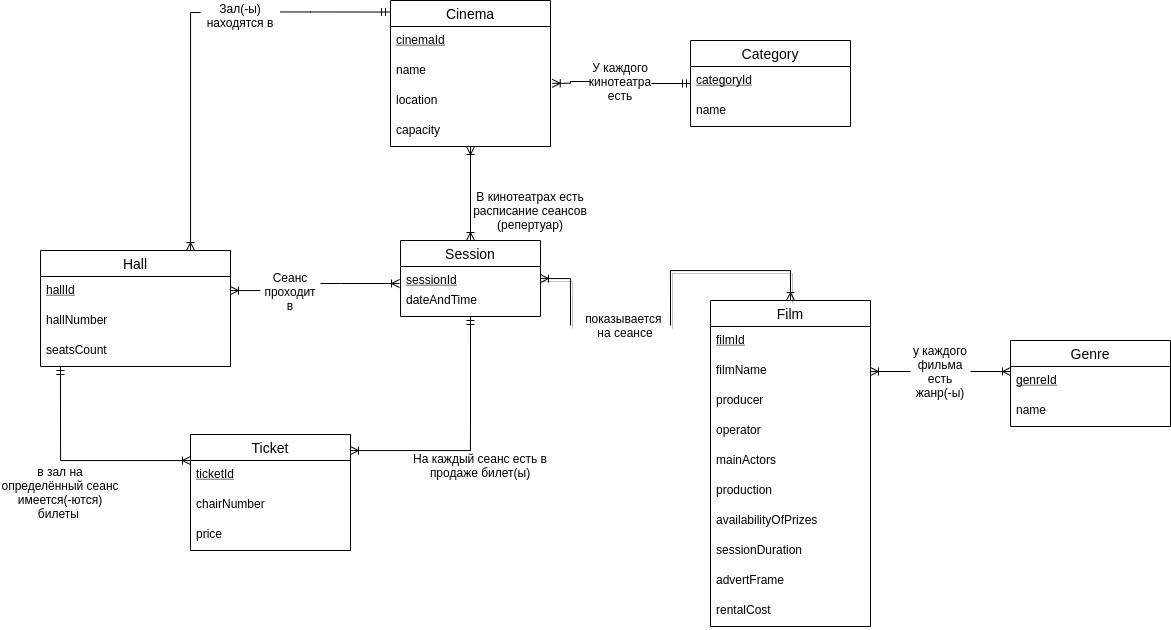


Рис. 1. ER-диаграмма.

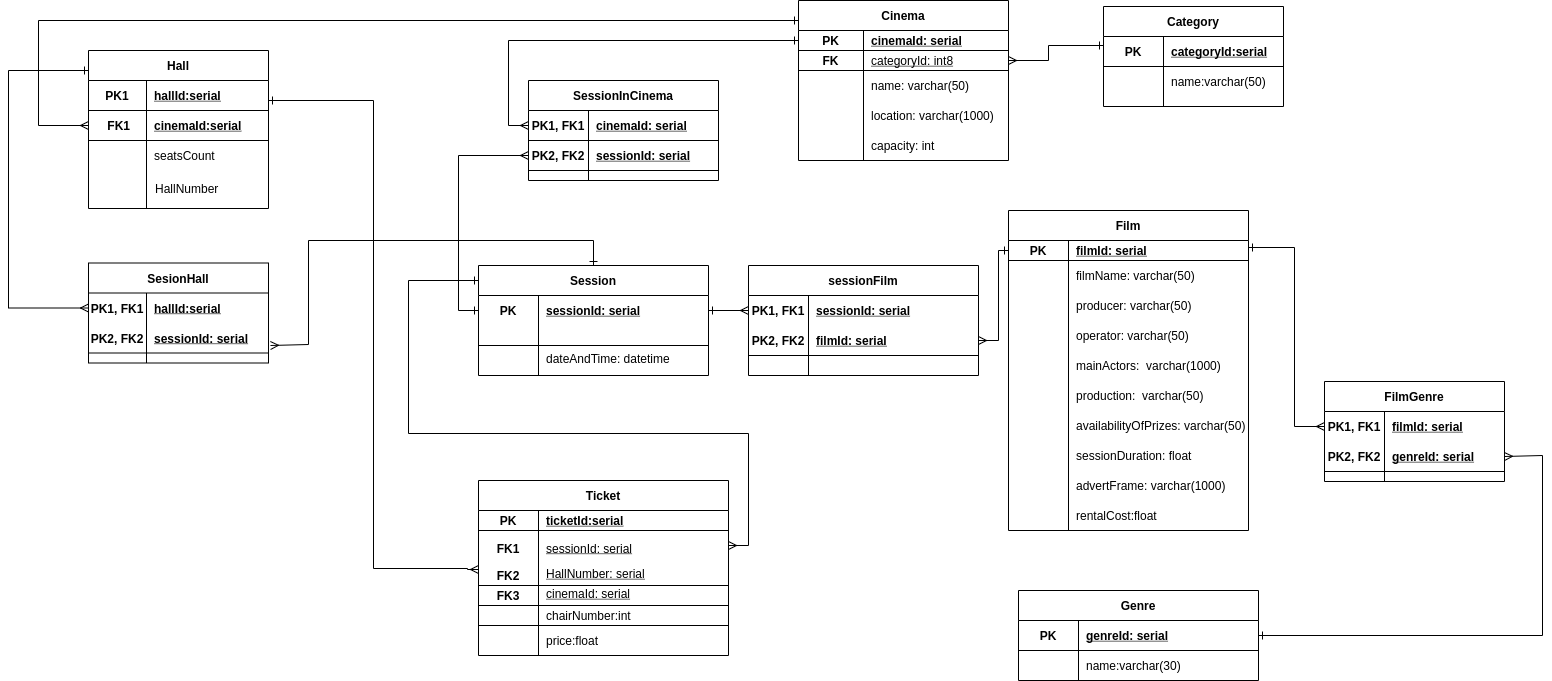
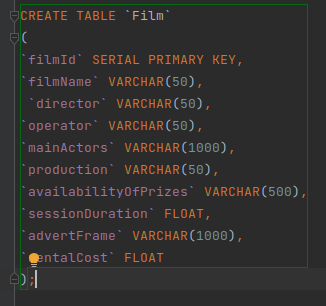
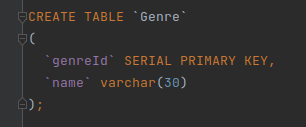
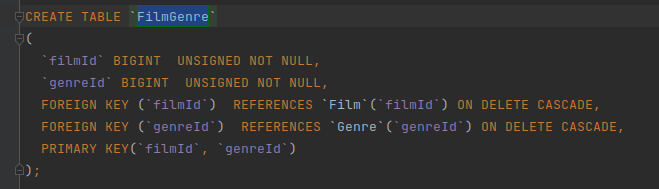


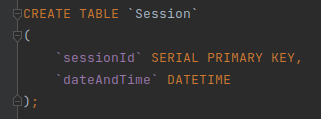
Рис. 2. Структура БД.

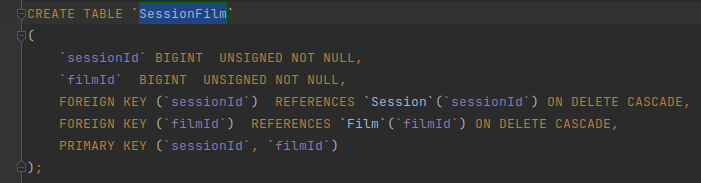
1. Локально развернут MySQL-сервер. Созданы таблицы.

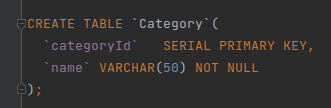


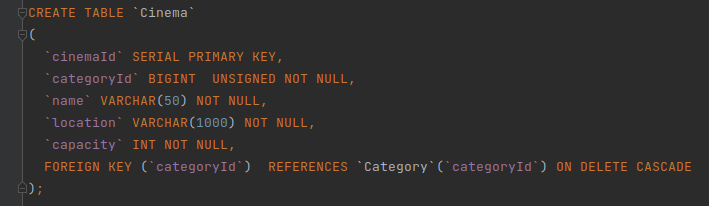


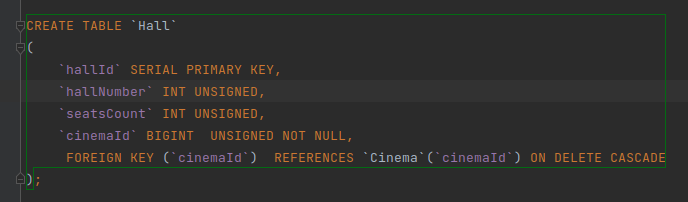


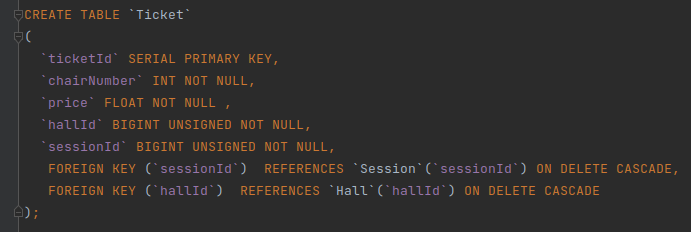


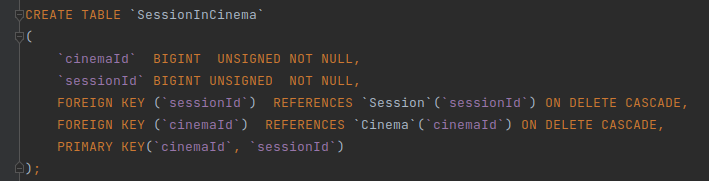


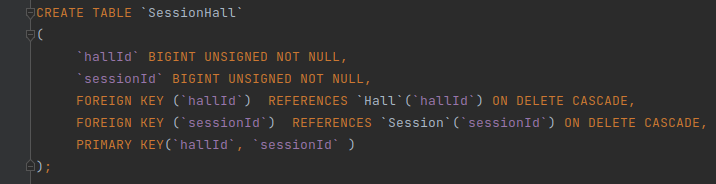




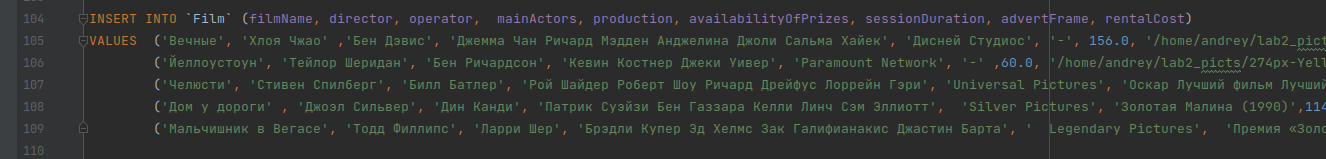


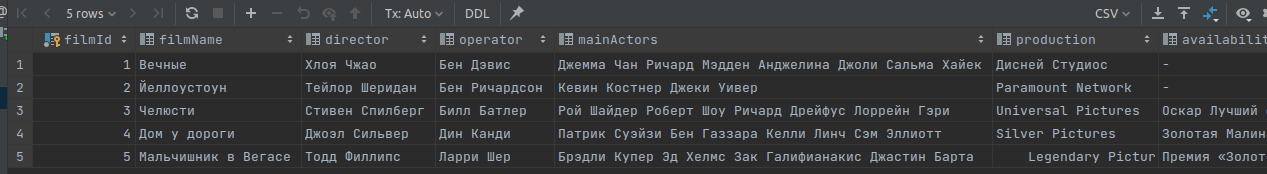


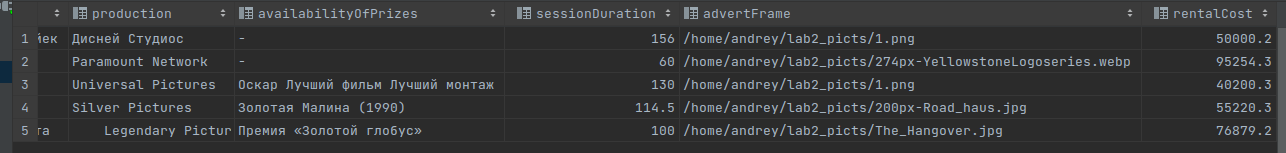


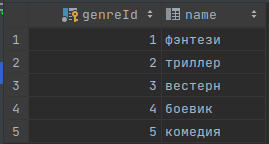
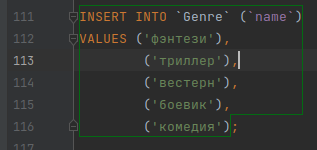


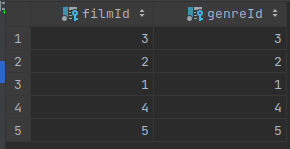
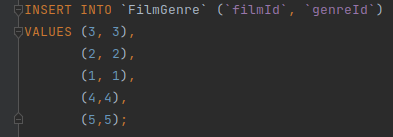
1. Таблицы были заполнены данными.

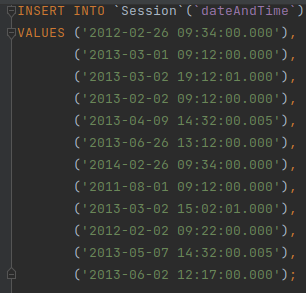
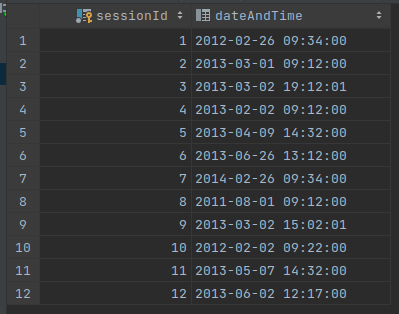


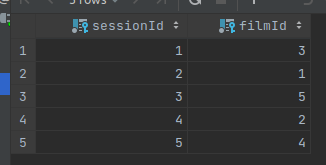
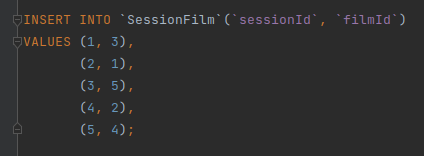


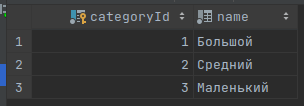
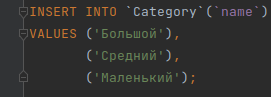


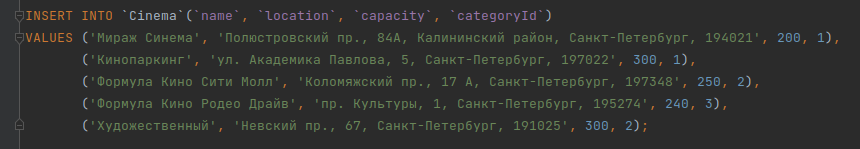


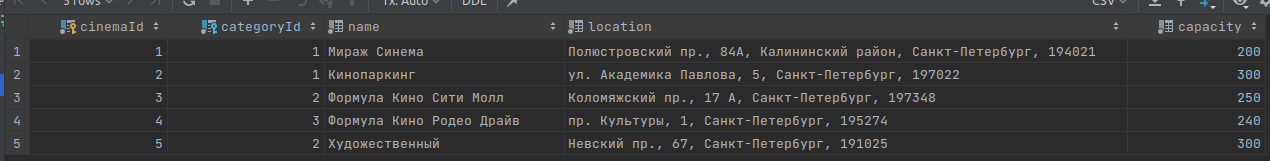


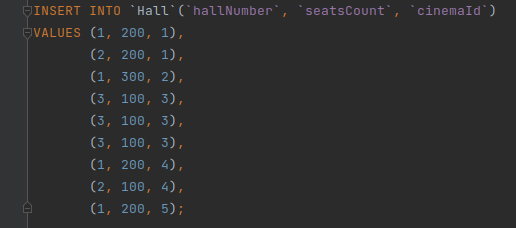
 

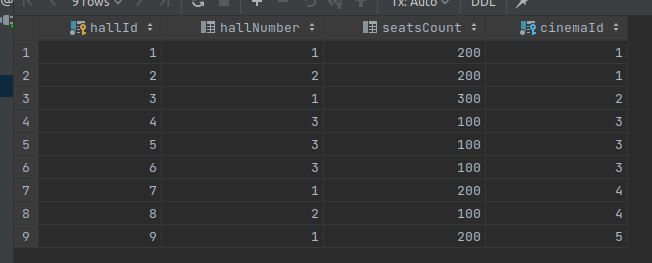


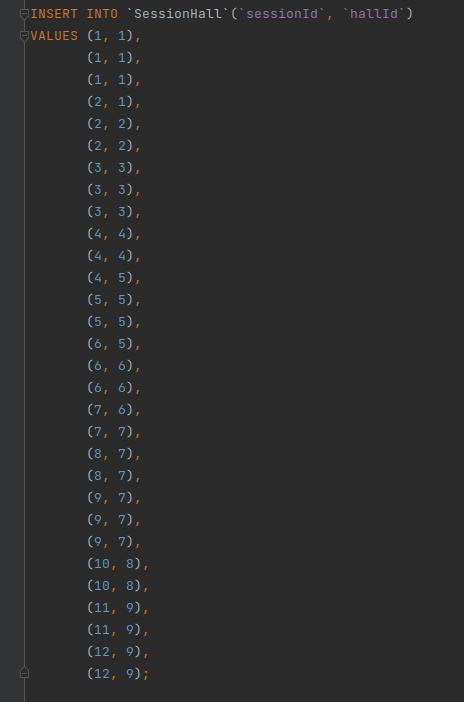


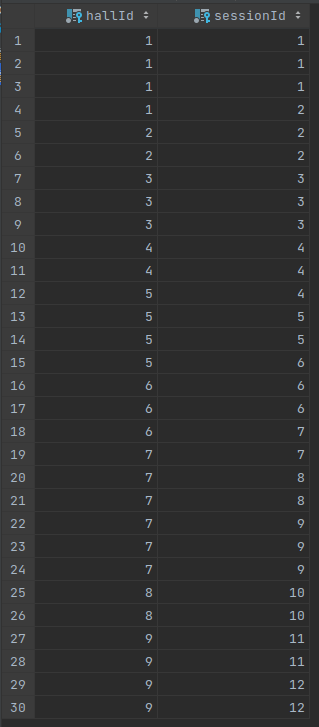


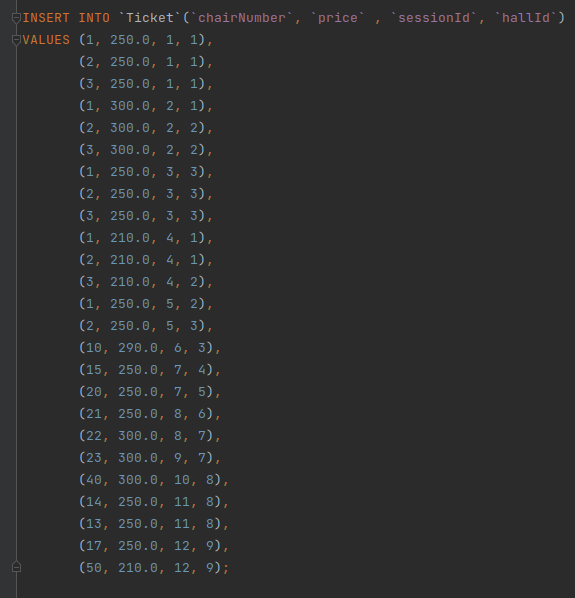


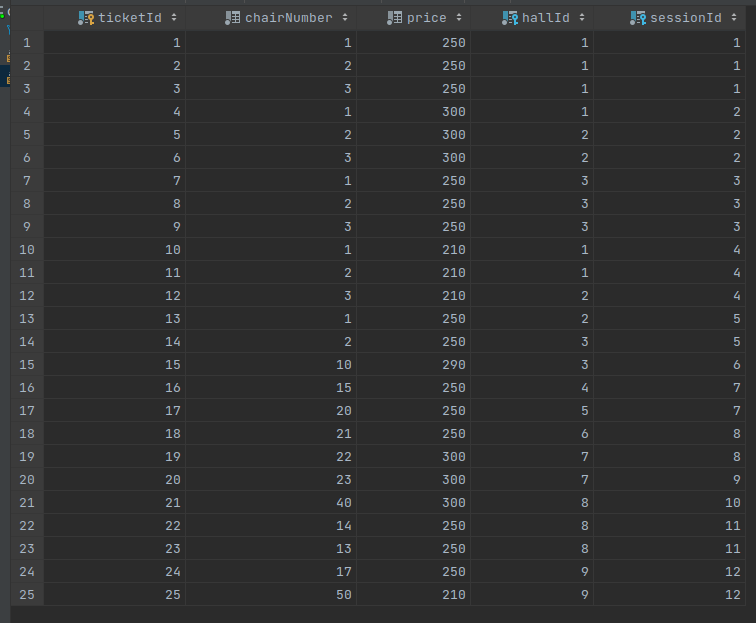


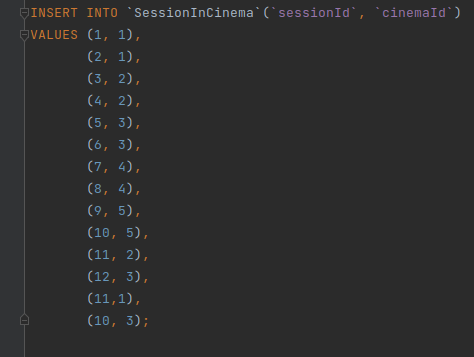


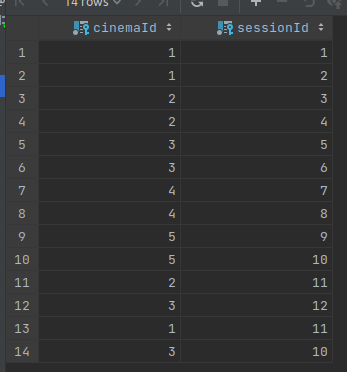






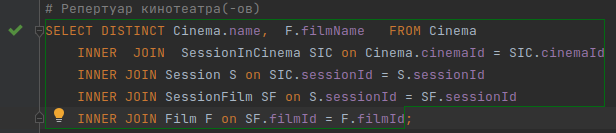


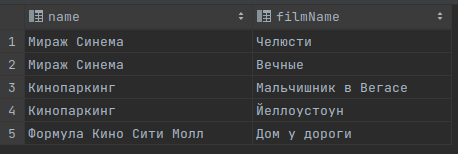




1. После заполнения таблиц были написаны требуемые запросы на выборку.

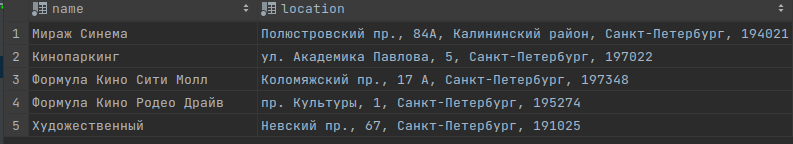
3.1. Репертуар кинотеатра(-ов).



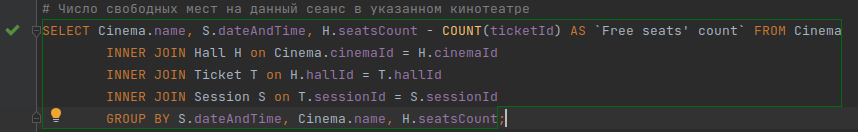


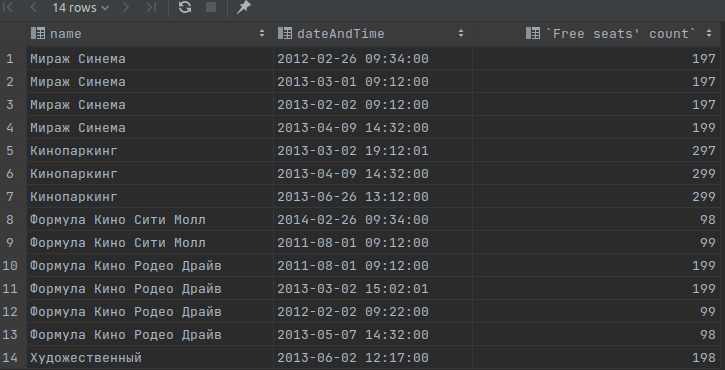
3.2. Адрес и район кинотеатра.



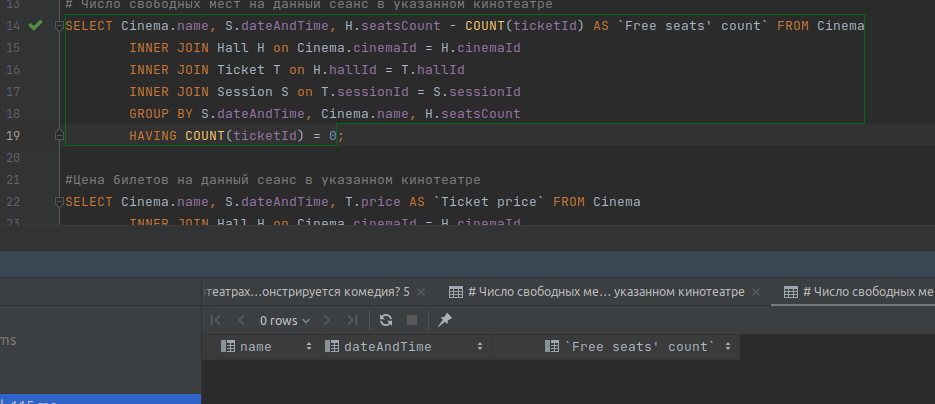


3.3. Число свободных мест на данный сеанс в указанном кинотеатре.



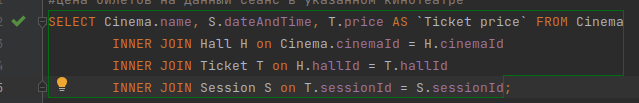


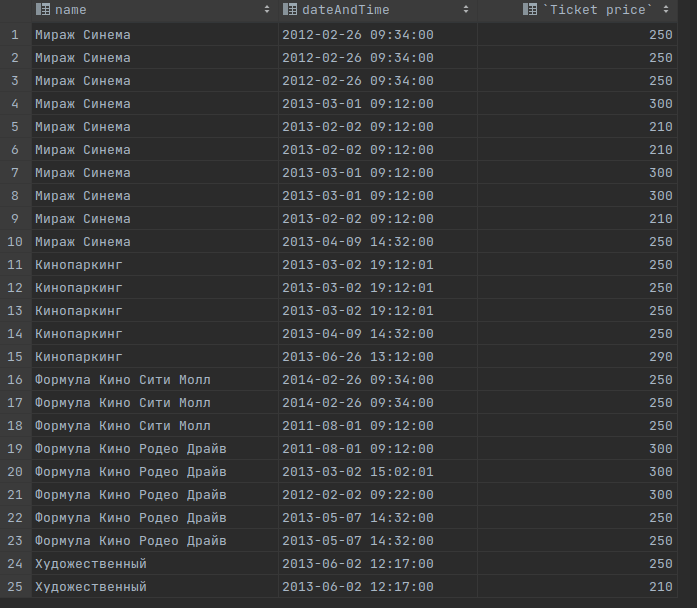
3.3.1. Реализация комментария преподавателя: “Запрос "Число свободных мест на данный сеанс в указанном кинотеатре" - проверьте, что будет, если на какой-нибудь сеанс вообще не будет куплено билетов.” Выполнена проверка:



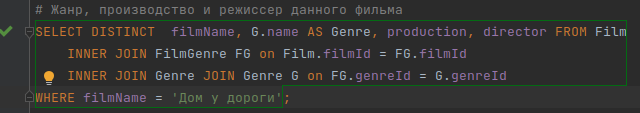
Проверка показала, что если на сеанс не куплено билетов, то ничего не выведется. Это вызвано тем, что таблицы были заполнены таким образом, что на каждый сеанс был куплен хотя бы один билет.

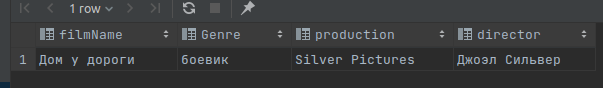
3.4. Цена билетов на данный сеанс в указанном кинотеатре.



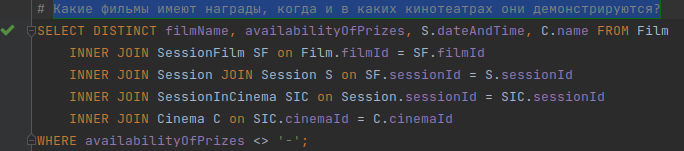


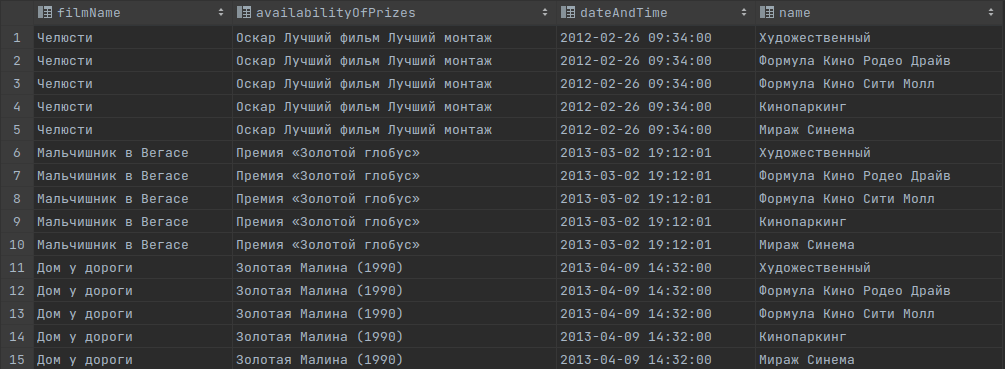
3.5. Жанр, производство и режиссер данного фильма.



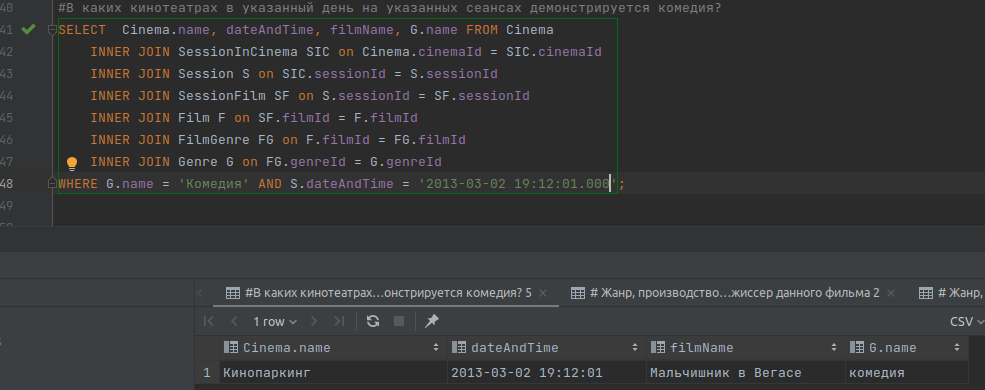


3.6. Какие фильмы имеют награды, когда и в каких кинотеатрах они демонстрируются.





3.7. В каких кинотеатрах в указанный день на указанных сеансах демонстрируется комедия.



**Выводы.**

Реализована база данных в СУБД MySQL. Написаны запросы на создание таблиц, вставку данных, и их выборку. Получены навыки работы с СУБД MySQL.

**ПРИЛОЖЕНИЕ A**

Ссылка на DB-FIDDLE:

<https://www.db-fiddle.com/f/p6eSLikKenwsg26kNsKCkT/2>

Ссылка на PR: <https://github.com/moevm/sql-2021-9381/pull/20>

**ПРИЛОЖЕНИЕ B**

**ИСХОДНЫЙ КОД ЗАПРОСОВ**

Файл lab2.sql:

CREATE DATABASE IF NOT EXISTS lab2;

USE lab2;

DROP TABLE IF EXISTS `SessionFilm`;

DROP TABLE IF EXISTS `FilmGenre`;

DROP TABLE IF EXISTS `Film`;

DROP TABLE IF EXISTS `Ticket`;

DROP TABLE IF EXISTS `SessionHall`;

DROP TABLE IF EXISTS `Hall`;

DROP TABLE IF EXISTS `SessionInCinema`;

DROP TABLE IF EXISTS `Cinema`;

DROP TABLE IF EXISTS `Category`;

DROP TABLE IF EXISTS `Genre`;

DROP TABLE IF EXISTS `Session`;

CREATE TABLE `Film`

(

`filmId` SERIAL PRIMARY KEY,

`filmName` VARCHAR(50),

`director` VARCHAR(50),

`operator` VARCHAR(50),

`mainActors` VARCHAR(1000),

`production` VARCHAR(50),

`availabilityOfPrizes` VARCHAR(500),

`sessionDuration` FLOAT,

`advertFrame` VARCHAR(1000),

`rentalCost` FLOAT

);

CREATE TABLE `Genre`

(

`genreId` SERIAL PRIMARY KEY,

`name` varchar(30)

);

CREATE TABLE `FilmGenre`

(

`filmId` BIGINT UNSIGNED NOT NULL,

`genreId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`filmId`) REFERENCES `Film`(`filmId`) ON DELETE CASCADE,

FOREIGN KEY (`genreId`) REFERENCES `Genre`(`genreId`) ON DELETE CASCADE,

PRIMARY KEY(`filmId`, `genreId`)

);

CREATE TABLE `Session`

(

`sessionId` SERIAL PRIMARY KEY,

`dateAndTime` DATETIME

);

CREATE TABLE `SessionFilm`

(

`sessionId` BIGINT UNSIGNED NOT NULL,

`filmId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`sessionId`) REFERENCES `Session`(`sessionId`) ON DELETE CASCADE,

FOREIGN KEY (`filmId`) REFERENCES `Film`(`filmId`) ON DELETE CASCADE,

PRIMARY KEY (`sessionId`, `filmId`)

);

CREATE TABLE `Category`(

`categoryId` SERIAL PRIMARY KEY,

`name` VARCHAR(50) NOT NULL

);

CREATE TABLE `Cinema`

(

`cinemaId` SERIAL PRIMARY KEY,

`categoryId` BIGINT UNSIGNED NOT NULL,

`name` VARCHAR(50) NOT NULL,

`location` VARCHAR(1000) NOT NULL,

`capacity` INT NOT NULL,

FOREIGN KEY (`categoryId`) REFERENCES `Category`(`categoryId`) ON DELETE CASCADE

);

CREATE TABLE `Hall`

(

`hallId` SERIAL PRIMARY KEY,

`hallNumber` INT UNSIGNED,

`seatsCount` INT UNSIGNED,

`cinemaId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`cinemaId`) REFERENCES `Cinema`(`cinemaId`) ON DELETE CASCADE

);

CREATE TABLE `Ticket`

(

`ticketId` SERIAL PRIMARY KEY,

`chairNumber` INT NOT NULL,

`price` FLOAT NOT NULL ,

`hallId` BIGINT UNSIGNED NOT NULL,

`sessionId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`sessionId`) REFERENCES `Session`(`sessionId`) ON DELETE CASCADE,

FOREIGN KEY (`hallId`) REFERENCES `Hall`(`hallId`) ON DELETE CASCADE

);

CREATE TABLE `SessionInCinema`

(

`cinemaId` BIGINT UNSIGNED NOT NULL,

`sessionId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`sessionId`) REFERENCES `Session`(`sessionId`) ON DELETE CASCADE,

FOREIGN KEY (`cinemaId`) REFERENCES `Cinema`(`cinemaId`) ON DELETE CASCADE,

PRIMARY KEY(`cinemaId`, `sessionId`)

);

CREATE TABLE `SessionHall`

(

`hallId` BIGINT UNSIGNED NOT NULL,

`sessionId` BIGINT UNSIGNED NOT NULL,

FOREIGN KEY (`hallId`) REFERENCES `Hall`(`hallId`) ON DELETE CASCADE,

FOREIGN KEY (`sessionId`) REFERENCES `Session`(`sessionId`) ON DELETE CASCADE,

PRIMARY KEY(`hallId`, `sessionId` )

);

INSERT INTO `Film` (`filmName`, `director`, `operator`, `mainActors`, `production`, `availabilityOfPrizes`, `sessionDuration`, `advertFrame`, `rentalCost`)

VALUES ('Вечные', 'Хлоя Чжао' ,'Бен Дэвис', 'Джемма Чан Ричард Мэдден Анджелина Джоли Сальма Хайек', 'Дисней Студиос', '-', 156.0, '/home/andrey/lab2\_picts/1.png', 50000.2),

('Йеллоустоун', 'Тейлор Шеридан', 'Бен Ричардсон', 'Кевин Костнер Джеки Уивер', 'Paramount Network', '-' ,60.0, '/home/andrey/lab2\_picts/274px-YellowstoneLogoseries.webp', 95254.3),

('Челюсти', 'Стивен Спилберг', 'Билл Батлер', 'Рой Шайдер Роберт Шоу Ричард Дрейфус Лоррейн Гэри', 'Universal Pictures', 'Оскар Лучший фильм Лучший монтаж', 130.0, '/home/andrey/lab2\_picts/1.png', 40200.3),

('Дом у дороги' , 'Джоэл Сильвер', 'Дин Канди', 'Патрик Суэйзи Бен Газзара Келли Линч Сэм Эллиотт', 'Silver Pictures', 'Золотая Малина (1990)',114.5, '/home/andrey/lab2\_picts/200px-Road\_haus.jpg', 55220.3),

('Мальчишник в Вегасе', 'Тодд Филлипс', 'Ларри Шер', 'Брэдли Купер Эд Хелмс Зак Галифианакис Джастин Барта', ' Legendary Pictures', 'Премия «Золотой глобус»', 100, '/home/andrey/lab2\_picts/The\_Hangover.jpg', 76879.2);

INSERT INTO `Genre` (`name`)

VALUES ('фэнтези'),

('триллер'),

('вестерн'),

('боевик'),

('комедия');

INSERT INTO `FilmGenre` (`filmId`, `genreId`)

VALUES (3, 3),

(2, 2),

(1, 1),

(4,4),

(5,5);

INSERT INTO `Session`(`dateAndTime`)

VALUES ('2012-02-26 09:34:00.000'),

('2013-03-01 09:12:00.000'),

('2013-03-02 19:12:01.000'),

('2013-02-02 09:12:00.000'),

('2013-04-09 14:32:00.005'),

('2013-06-26 13:12:00.000'),

('2014-02-26 09:34:00.000'),

('2011-08-01 09:12:00.000'),

('2013-03-02 15:02:01.000'),

('2012-02-02 09:22:00.000'),

('2013-05-07 14:32:00.005'),

('2013-06-02 12:17:00.000');

INSERT INTO `SessionFilm`(`sessionId`, `filmId`)

VALUES (1, 3),

(2, 1),

(3, 5),

(4, 2),

(5, 4);

INSERT INTO `Category`(`name`)

VALUES ('Большой'),

('Средний'),

('Маленький');

INSERT INTO `Cinema`(`name`, `location`, `capacity`, `categoryId`)

VALUES ('Мираж Синема', 'Полюстровский пр., 84А, Калининский район, Санкт-Петербург, 194021', 200, 1),

('Кинопаркинг', 'ул. Академика Павлова, 5, Санкт-Петербург, 197022', 300, 1),

('Формула Кино Сити Молл', 'Коломяжский пр., 17 А, Санкт-Петербург, 197348', 250, 2),

('Формула Кино Родео Драйв', 'пр. Культуры, 1, Санкт-Петербург, 195274', 240, 3),

('Художественный', 'Невский пр., 67, Санкт-Петербург, 191025', 300, 2);

INSERT INTO `Hall`(`hallNumber`, `seatsCount`, `cinemaId`)

VALUES (1, 200, 1),

(2, 200, 1),

(1, 300, 2),

(3, 100, 3),

(3, 100, 3),

(3, 100, 3),

(1, 200, 4),

(2, 100, 4),

(1, 200, 5);

INSERT INTO `SessionHall`(`sessionId`, `hallId`)

VALUES (1, 1),

(2, 1),

(2, 2),

(3, 3),

(4, 4),

(4, 5),

(5, 5),

(6, 5),

(6, 6),

(7, 6),

(7, 7),

(8, 7),

(9, 7),

(10, 8),

(11, 9),

(12, 9);

INSERT INTO `Ticket`(`chairNumber`, `price` , `sessionId`, `hallId`)

VALUES (1, 250.0, 1, 1),

(2, 250.0, 1, 1),

(3, 250.0, 1, 1),

(1, 300.0, 2, 1),

(2, 300.0, 2, 2),

(3, 300.0, 2, 2),

(1, 250.0, 3, 3),

(2, 250.0, 3, 3),

(3, 250.0, 3, 3),

(1, 210.0, 4, 1),

(2, 210.0, 4, 1),

(3, 210.0, 4, 2),

(1, 250.0, 5, 2),

(2, 250.0, 5, 3),

(10, 290.0, 6, 3),

(15, 250.0, 7, 4),

(20, 250.0, 7, 5),

(21, 250.0, 8, 6),

(22, 300.0, 8, 7),

(23, 300.0, 9, 7),

(40, 300.0, 10, 8),

(14, 250.0, 11, 8),

(13, 250.0, 11, 8),

(17, 250.0, 12, 9),

(50, 210.0, 12, 9);

INSERT INTO `SessionInCinema`(`sessionId`, `cinemaId`)

VALUES (1, 1),

(2, 1),

(3, 2),

(4, 2),

(5, 3),

(6, 3),

(7, 4),

(8, 4),

(9, 5),

(10, 5),

(11, 2),

(12, 3),

(11,1),

(10, 3);

Файл queries.sql:

use lab2;

# Репертуар кинотеатра(-ов)

SELECT DISTINCT Cinema.name, F.filmName FROM Cinema

INNER JOIN SessionInCinema SIC on Cinema.cinemaId = SIC.cinemaId

INNER JOIN Session S on SIC.sessionId = S.sessionId

INNER JOIN SessionFilm SF on S.sessionId = SF.sessionId

INNER JOIN Film F on SF.filmId = F.filmId;

# Адрес и район кинотеатра

SELECT name, location FROM Cinema;

# Число свободных мест на данный сеанс в указанном кинотеатре

SELECT Cinema.name, S.dateAndTime, H.seatsCount - COUNT(ticketId) AS `Free seats' count` FROM Cinema

INNER JOIN Hall H on Cinema.cinemaId = H.cinemaId

INNER JOIN Ticket T on H.hallId = T.hallId

INNER JOIN Session S on T.sessionId = S.sessionId

GROUP BY S.dateAndTime, Cinema.name, H.seatsCount

HAVING COUNT(ticketId) = 0;

#Цена билетов на данный сеанс в указанном кинотеатре

SELECT Cinema.name, S.dateAndTime, T.price AS `Ticket price` FROM Cinema

INNER JOIN Hall H on Cinema.cinemaId = H.cinemaId

INNER JOIN Ticket T on H.hallId = T.hallId

INNER JOIN Session S on T.sessionId = S.sessionId;

# Жанр, производство и режиссер данного фильма

SELECT DISTINCT filmName, G.name AS Genre, production, director FROM Film

INNER JOIN FilmGenre FG on Film.filmId = FG.filmId

INNER JOIN Genre JOIN Genre G on FG.genreId = G.genreId

WHERE filmName = 'Дом у дороги';

# Какие фильмы имеют награды, когда и в каких кинотеатрах они демонстрируются?

SELECT DISTINCT filmName, availabilityOfPrizes, S.dateAndTime, C.name FROM Film

INNER JOIN SessionFilm SF on Film.filmId = SF.filmId

INNER JOIN Session JOIN Session S on SF.sessionId = S.sessionId

INNER JOIN SessionInCinema SIC on Session.sessionId = SIC.sessionId

INNER JOIN Cinema C on SIC.cinemaId = C.cinemaId

WHERE availabilityOfPrizes <> '-';

#В каких кинотеатрах в указанный день на указанных сеансах демонстрируется комедия?

SELECT Cinema.name, dateAndTime, filmName, G.name FROM Cinema

INNER JOIN SessionInCinema SIC on Cinema.cinemaId = SIC.cinemaId

INNER JOIN Session S on SIC.sessionId = S.sessionId

INNER JOIN SessionFilm SF on S.sessionId = SF.sessionId

INNER JOIN Film F on SF.filmId = F.filmId

INNER JOIN FilmGenre FG on F.filmId = FG.filmId

INNER JOIN Genre G on FG.genreId = G.genreId

WHERE G.name = 'Комедия' AND S.dateAndTime = '2013-03-02 19:12:01.000';