Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №3**

по дисциплине «Базы данных»

«Язык SQL-DML»

Работу выполнила

студентка гр. 43501/3

К.В. Ардашова

Проверил преподаватель

А.В. Мяснов

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2016г

Санкт - Петербург

2016

**Цель работы**

Познакомиться с языком создания запросов управления данными SQL-DML.

**Программа работы**

1. Выполнить все стандартные запросы.
2. Реализовать SQL-запросы в соответствии с индивидуальным заданием.
3. Выполненные запросы SELECT сохранить в БД в виде представлений, запросы INSERT, UPDATE или DELETE - в виде ХП. Выложить скрипт в Subversion.

**Ход работы**

**1. Выполнить все стандартные запросы**

**1.1. Выборка всех данных из каждой таблицы**

SELECT \* FROM cinema;

SELECT \* FROM cinemacompany;

SELECT \* FROM MPAA\_rating;

SELECT \* FROM session;

SELECT \* FROM tickets\_sold;

SELECT \* FROM actor;

SELECT \* FROM age\_restriction;

SELECT \* FROM award;

**SELECT \* FROM country;**

SELECT \* FROM film;

SELECT \* FROM genre;

SELECT \* FROM link\_award\_film;

SELECT \* FROM link\_film\_actor;

SELECT \* FROM link\_film\_contry;

SELECT \* FROM link\_film\_genre;

SELECT \* FROM link\_film\_producer;

SELECT \* FROM link\_role\_award;

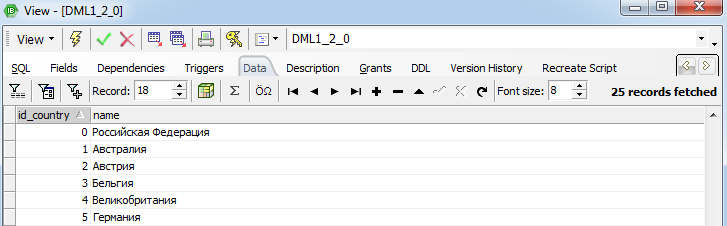
SELECT \* FROM producer;

SELECT \* FROM rating;

SELECT \* FROM stage\_director;

SELECT \* FROM user;

Пример выполнения запроса для таблицы country:

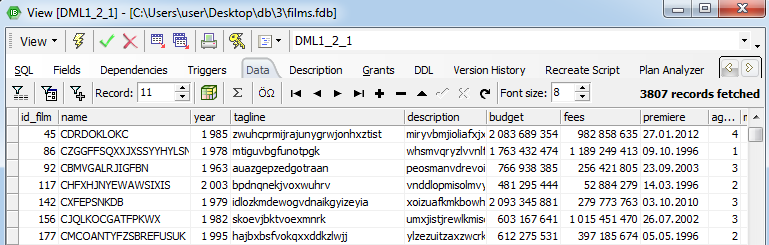


Выбраны все значения таблицы (25 стран).

**1.2. Выборка данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN.**

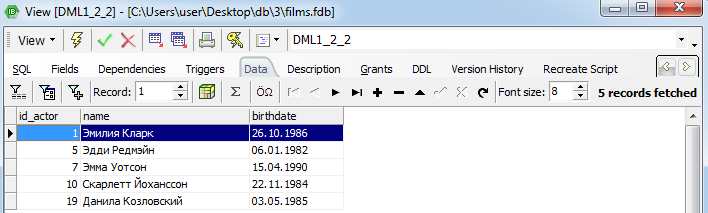
Выбрать все фильмы, название которых начинается с буквы «С»:

**SELECT \* from "film" where "film"."name" like 'C%';**



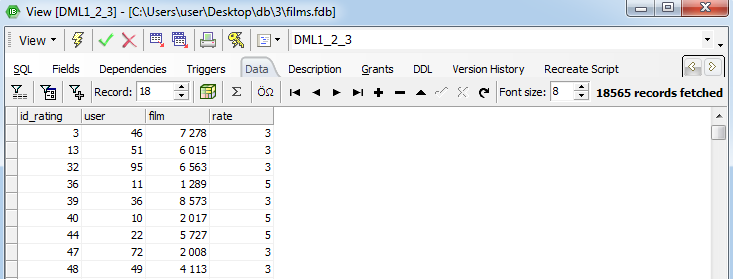
Выбрать всех актеров, родившихся между заданными датами:

**SELECT \* from "actor" where "actor"."birthdate" between '01.01.1980' and '01.01.2016';**



Вывести все оценки, имеющие значения 1, 3 или 5, но не меньше 2.

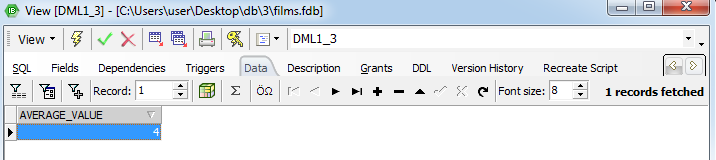
**SELECT \* from "rating" where "rating"."rate" IN(1, 3, 5) AND "rating"."rate">2;**



**1.3. Сделать в запросе вычисляемое поле.**

Подсчитывается среднее значение рейтингов:

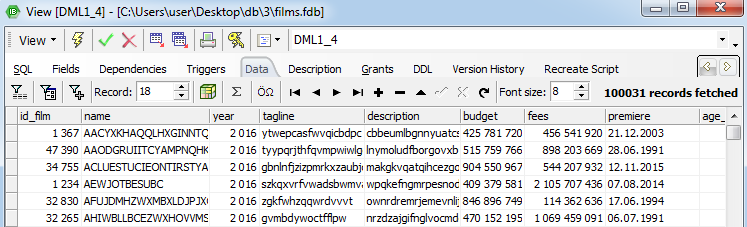
**SELECT sum("rating"."rate")/count("rating"."rate") as average\_value from "rating";**



**1.4. Выборка всех данных с сортировкой по нескольким полям.**

Отображение фильмов с сортировкой по годам и названиям:

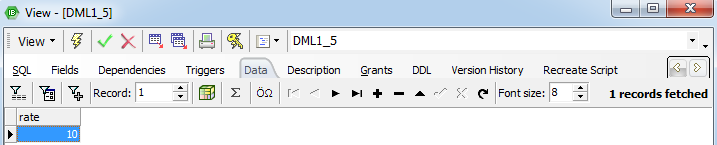
**SELECT**  **\* from "film" order by "film"."year" desc, "film"."name" ;**



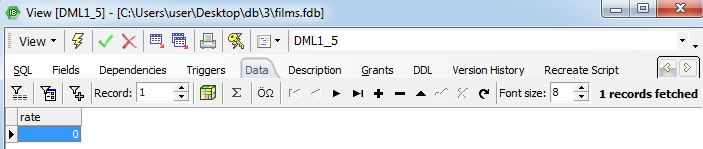
**1.5. Запрос, вычисляющий несколько совокупных характеристик таблиц.**

В пункте 1.3. осуществлено вычисление числа строк, вычисление суммы значений полей. Аналогично, можно найти максимальную и минимальную оценки, например:

**SELECT max("rating"."rate") from "rating";**



**SELECT min("rating"."rate") from "rating";**

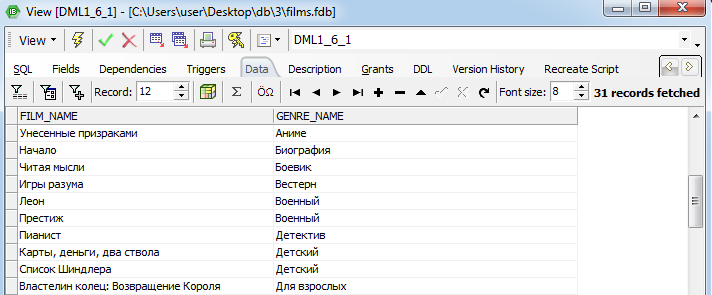


**1.6. Выборка данных из связанных таблиц (не менее двух примеров)**

Вывод жанров фильмов:

**SELECT "film"."name", "genre"."name" from "film", "link\_film\_genre", "genre"**

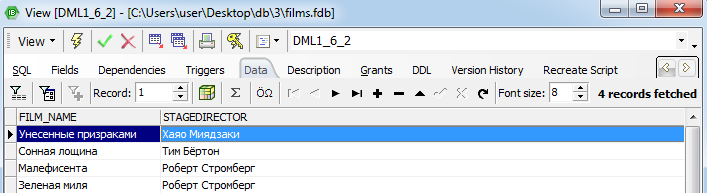
**where "film"."id\_film" = "link\_film\_genre"."film" and "link\_film\_genre"."genre" = "genre"."id\_genre" order by "genre"."name", "film"."name";**



Вывод режиссеров фильмов:

**SELECT "film"."name", "stage\_director"."name" from "film", "stage\_director"**

**where "film"."stage\_director" = "stage\_director"."id\_director";**



**1.7. Запрос, рассчитывающий совокупную характеристику с использованием группировки, наложить ограничение на результат группировки.**

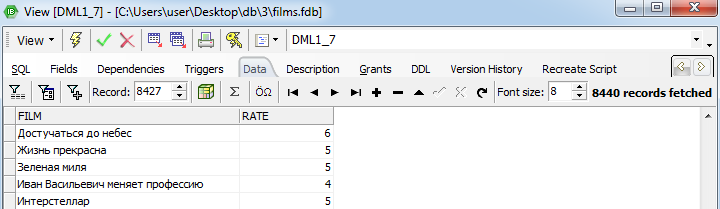
Вывести фильмы и их рейтинг, где рейтинг больше 3.

**SELECT "film"."name", avg("rating"."rate") from "film", "rating"**

**where "rating"."film" = "film"."id\_film"**

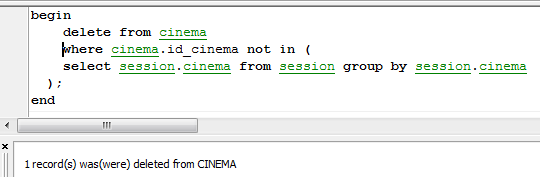
**group by "film"."name"**

**having avg("rating"."rate")>3;**



**1.8. Вложенный запрос.**

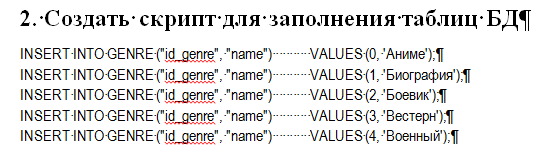
Удалить все кинотеатры, в которых нет ни одного сеанса:



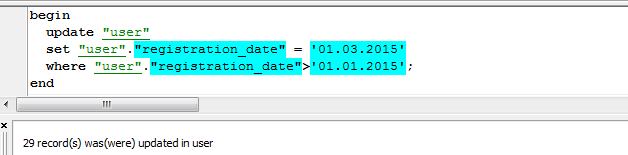
**1.9. С помощью оператора INSERT добавить в каждую таблицу по записи.**

Примеры приведены в отчете по лабораторной работе 2 в пункте 2.

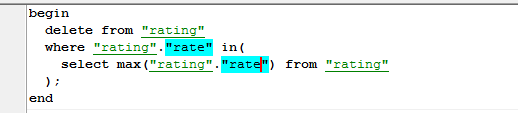
Например:



**1.10. С помощью оператора UPDATE изменить значения нескольких полей у всех записей, отвечающих заданному условию.**



**1.11. С помощью оператора DELETE удалить запись, имеющую минимальное значение некоторой совокупной характеристики**

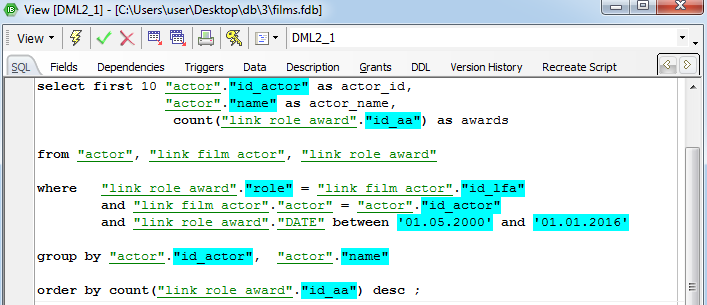


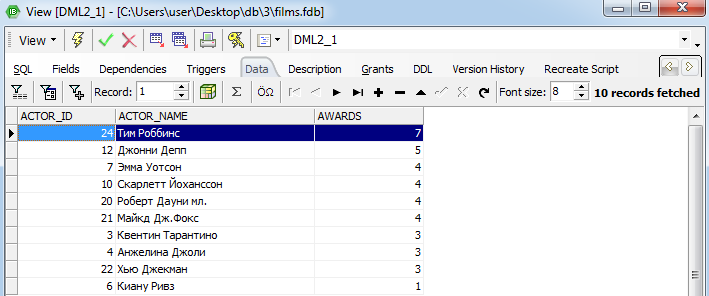
**1.12. С помощью оператора DELETE удалить записи в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос).**

Реализовано в пункте 1.8.

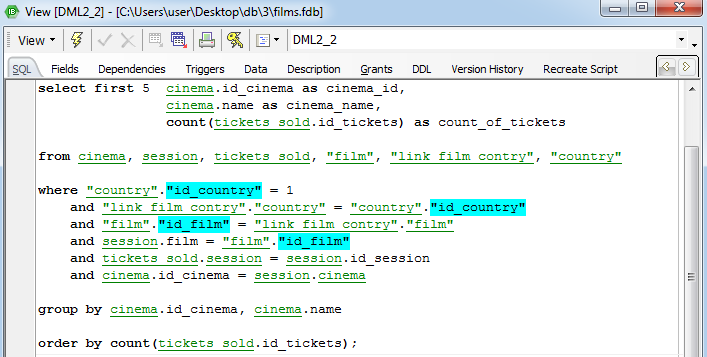
**2. Реализовать SQL-запросы в соответствии с заданием.**

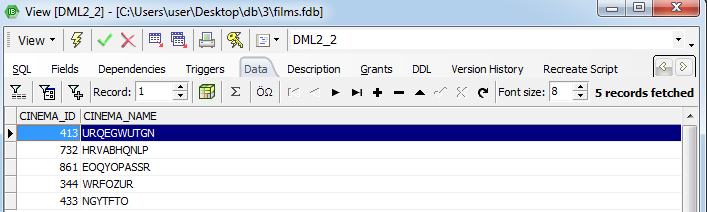
**2.1. Вывести 10 актеров с максимальным количеством наград за заданный промежуток времени.**



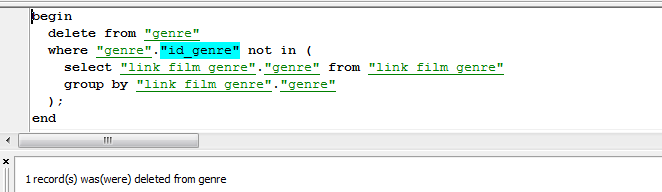


**2.2. Вывести 5 кинотеатров, в которых хуже всего продаются билеты на фильмы из заданной страны.**



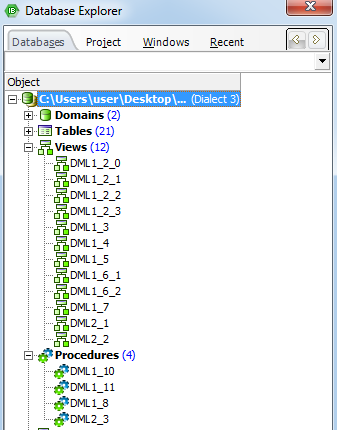


**2.3. Удалить неиспользуемые жанры.**



**3. Сохранение запросов**

Запросы сохранены в виде представлений и процедур (в зависимости от типа запроса согласно заданию).



**Выводы**

Язык SQL – язык структурированных запросов, определяющий два подмножества языка:

1. SQL-DDL – язык определения структур и ограничений целостности баз данных с командами создания и удаления баз данных и таблиц.
2. SQL-DML – язык манипулирования данными.

В данной работе использованы элементы языка SQL, предназначенные выбора, вставки, изменения и удаления данных с заданным параметрами.

Язык DML удобно использовать для составления запросов, в которых не нужно производить сложных (в много действий) вычислений над данными одновременно из разных таблиц.

Для более сложных запросов можно использовать представления и хранимые процедуры, хранящие и выполняющие сложные запросы на стороне базы. Кроме того, есть возможность предоставить пользователю доступ только к представлениям, а не ко всем данным базы, и тем самым разграничить доступ для разных пользователей.

Удобство представлений заключается в том, что к ним можно обращаться так же, как к таблицам базы, а хранимых процедур – в том, что они имеют сходства с уже знакомыми процедурными языками программирования.