Государственный Университет Молдовы   
Факультет Математики и Информатики  
Департамент Информатики

**Индивидуальная работа**

По курсу «Базы данных»

тема : База данных для сайта агрегатора оценок фильмов

Выполнил студент группа I2302:

Проверил преподаватель:  
Карчева Н.Ф.

Кишинёв, 2020

**Цель работы :**

Создание реляционной базы данных для сайта агрегатора оценок фильмов.

**Этапы создания базы данных :**

1. Анализ необходимых моделей
2. Создание схемы базы данных
3. Нормализация базы

**Содержание работы :**

1) Схема базы данных

2) Описание запросов к базе

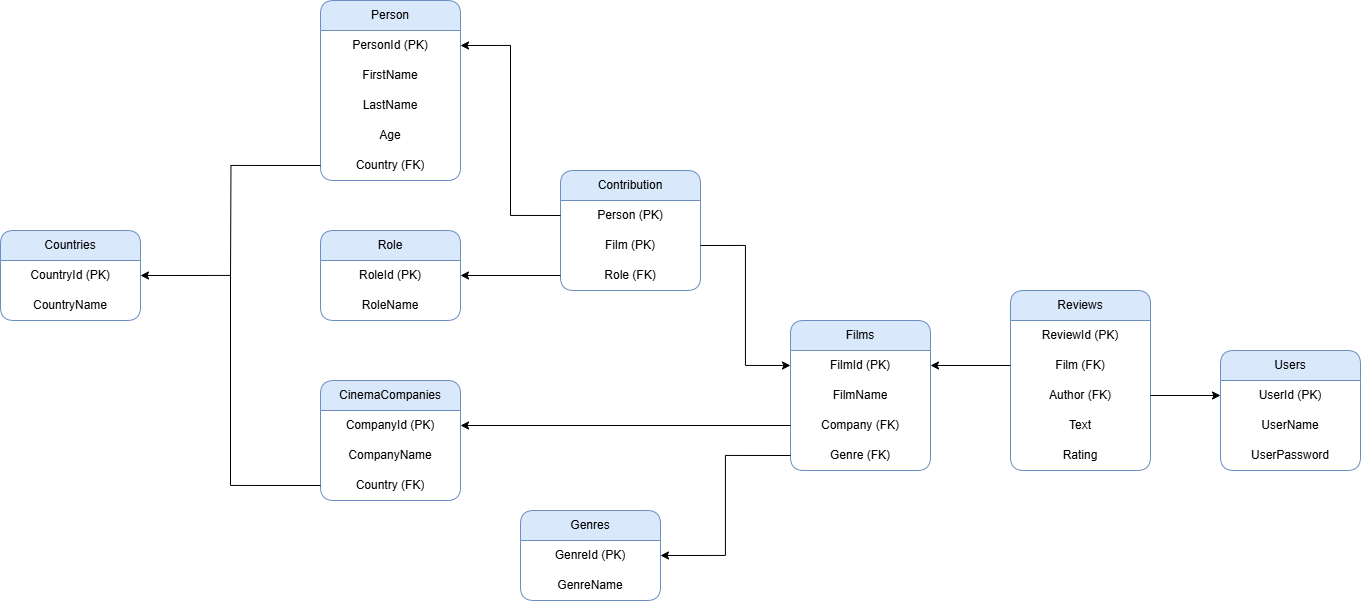
4) Описание представлений

**Схема данных включает в себя следующие таблицы :**

1. *Users*
2. Films
3. Genres
4. Reviews
5. Role
6. Contributions
7. Person
8. Countries
9. CinemaCompanies

**Отношения таблиц :**

1. *Users – Films: Many to many*
2. Films – Genres: Many to one
3. Films – Person: Many to many
4. Contribution – Role: Many to one
5. Person – Countries: Many to one
6. Countries – CinemaCompanies: One to many
7. CinemaCompanies – Films: One to many



**Создание селектов:**

1) Вывести все фильмы, в создании которых участвовал ‘Christopher Nolan’

select distinct \*

from Films

where FilmId in

(select Film

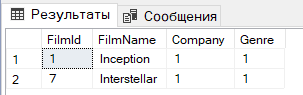
from Contributions

where Person =

(select PersonId

from Persons

where CONCAT(FirstName, ' ', LastName) = 'Christopher Nolan'))



2) Выбрать все рецензии на фильм ‘Inception’

select \*

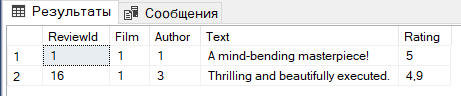
from Reviews

where Film =

(select FilmId

from Films

where FilmName = 'Inception')



3) Выбрать все фильмы с количеством рецензий на них и поле со значениями HIGH, LOW, MEDIUM в зависимости от их количества

select

FilmName,

COUNT(ReviewId) AS ReviewCount,

case

when AVG(Rating) < 2.5 then 'LOW'

when AVG(Rating) < 4 then 'MEDIUM'

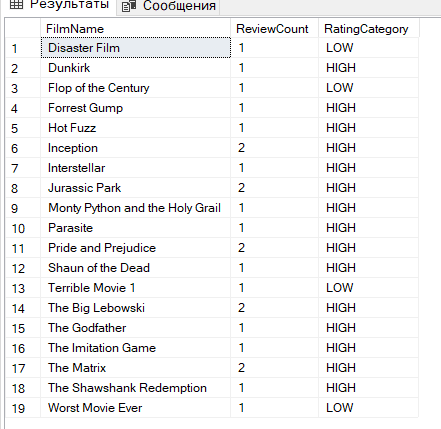
else 'HIGH'

end as RatingCategory

from Reviews

join Films on Film = FilmId

group by FilmName



4) Подсчитать фильмы по странам

select

CountryName,

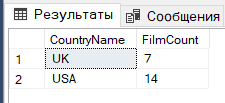
COUNT(Films.FilmId) as FilmCount

from Countries

join CinemaCompanies on Countries.CountryId = CinemaCompanies.Country

join Films on CinemaCompanies.CompanyId = Films.Company

group by Countries.CountryName



5) Выбрать 5 самых высоко оценённых фильмов в США

select top 5

Films.FilmName,

avg(Reviews.Rating)

from Films

join CinemaCompanies on CinemaCompanies.CompanyId = Films.Company

join Countries on Countries.CountryId = CinemaCompanies.Country

join Reviews on Reviews.Film = Films.FilmId

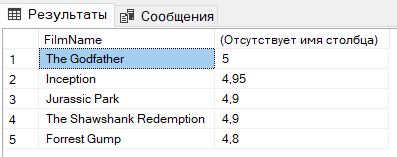
where Countries.CountryName = 'USA'

group by

Films.FilmName,

Countries.CountryName

order by AVG(Rating) desc



6) Выбрать все фильмы, где было больше одного режиссёра

select

FilmName,

COUNT(Contributions.Person) as Directors

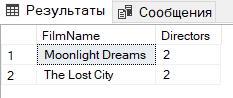
from Films

join Contributions on Contributions.Film = Films.FilmId

where Contributions.Role = 1

group by Films.FilmName

having count (Contributions.Person) > 1



7) Выбрать топ 5 пользователей по количеству написанных рецензий

select top 5

Users.UserName,

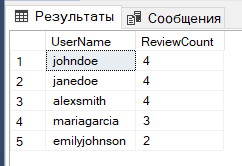
COUNT(Reviews.ReviewId) as ReviewCount

from Users

join Reviews on Reviews.Author = Users.UserId

group by Users.UserName

order by ReviewCount desc



8) Выбрать фильмы вместе с подсчётом уникальных стран, из которых происходят актёры

select

Films.FilmName,

count(distinct Persons.Country) as CountryCount

from Films

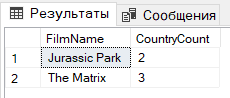
join Contributions on Contributions.Film = Films.FilmId

join Persons on Persons.PersonId = Contributions.Person

where Contributions.Role = 2

group by Films.FilmName

having count(distinct Persons.Country) > 1



9) Подсчитать фильмы по жанрам

select

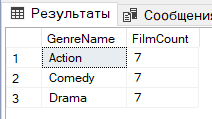
Genres.GenreName,

COUNT(Films.FilmId) as FilmCount

from Films

join Genres on Genres.GenreId = Films.Genre

group by Genres.GenreName



10) Выбрать средний рейтинг фильмов по жанрам

select

Genres.GenreName,

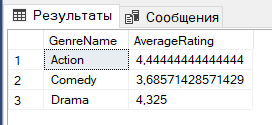
AVG(Reviews.Rating) as AverageRating

from Films

join Reviews on Films.FilmId = Reviews.Film

join Genres on Films.Genre = Genres.GenreId

group by Genres.GenreName



**Создание представлений:**

1) Представление, включающее данные о фильме, кинокомпании и жанре фильма

create view FilmDetails as

select

f.FilmId,

f.FilmName,

c.CompanyName,

g.GenreName

from Films f

join CinemaCompanies c on f.Company = c.CompanyId

join Genres g on f.Genre = g.GenreId



2) Представление, включающее данные о рецензии, фильме, авторе

create view ReviewDetails as

select

r.ReviewId,

f.FilmName,

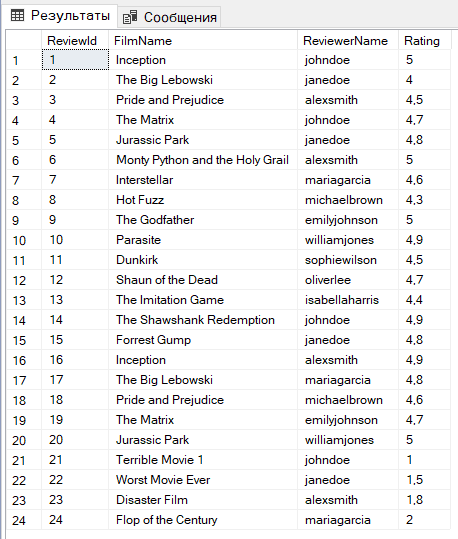
u.UserName as ReviewerName,

r.Rating

from Reviews r

join Films f on r.Film = f.FilmId

join Users u on r.Author = u.UserId



3) Представление, включающее данные о персоне, участвовавшей в создании фильма и фильме

create view PersonContributions as

select

p.FirstName,

p.LastName,

f.FilmName,

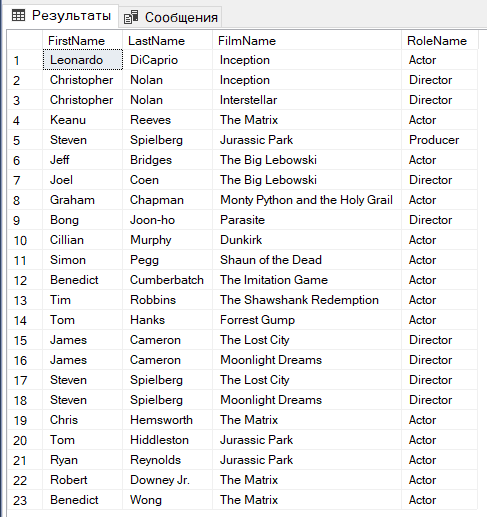
r.RoleName

from Contributions c

join Persons p on c.Person = p.PersonId

join Films f on c.Film = f.FilmId

join Roles r on c.Role = r.RoleId



4) Представление с фильмами, со средней оценкой > 4.5

create view TopRatedFilms as

select

f.FilmName,

AVG(r.Rating) as AverageRating

from Reviews r

join Films f on r.Film = f.FilmId

group by f.FilmName

having AVG(r.Rating) > 4.5



4) Представление с фильмами и странами

create view FilmCountries as

select f.FilmName, c.CountryName

from Films f

join CinemaCompanies cc on f.Company = cc.CompanyId

join Countries c on cc.Country = c.CountryId



**Полный SQL код:**

--create database IndividualWorkDB

--use IndividualWorkDB

--drop database IndividualWorkDB

--use Users

-- DB SETUP

create table Countries(

CountryId int,

CountryName varchar(255) not null,

primary key (CountryId))

create table Roles(

RoleId int,

RoleName varchar(255) not null,

primary key (RoleId))

create table Users(

UserId int,

UserName varchar(255) not null,

UserPassword varchar(255) not null,

primary key (UserId))

create table CinemaCompanies(

CompanyId int not null,

CompanyName varchar(255) not null,

Country int not null,

primary key (CompanyId),

foreign key (Country) references Countries(CountryId))

create table Genres(

GenreId int not null,

GenreName varchar(255) not null,

primary key (GenreId))

create table Films(

FilmId int not null,

FilmName varchar(255) not null,

Company int,

Genre int,

primary key (FilmId),

foreign key (Company) references CinemaCompanies(CompanyId),

foreign key (Genre) references Genres(GenreId))

create table Reviews(

ReviewId int not null,

Film int not null,

Author int not null,

Text varchar(255),

Rating float not null,

primary key (ReviewId),

foreign key (Film) references Films(FilmId),

foreign key (Author) references Users(UserId))

create table Persons(

PersonId int not null,

FirstName varchar(255) not null,

LastName varchar(255) not null,

Age int not null,

Country int not null,

primary key (PersonId),

foreign key (Country) references Countries(CountryId))

create table Contributions(

Person int not null,

Film int not null,

Role int not null,

primary key (Person, Film),

foreign key (Person) references Persons(PersonId),

foreign key (Film) references Films(FilmId),

foreign key (Role) references Roles(RoleId))

-- DATA SETUP

INSERT INTO Countries (CountryId, CountryName) VALUES

(1, 'USA'),

(2, 'UK'),

(3, 'Canada');

INSERT INTO Roles (RoleId, RoleName) VALUES

(1, 'Director'),

(2, 'Actor'),

(3, 'Producer');

INSERT INTO CinemaCompanies (CompanyId, CompanyName, Country) VALUES

(1, 'Warner Bros', 1),

(2, 'Universal Pictures', 1),

(3, 'BBC Films', 2);

INSERT INTO Genres (GenreId, GenreName) VALUES

(1, 'Action'),

(2, 'Comedy'),

(3, 'Drama');

INSERT INTO Films (FilmId, FilmName, Company, Genre) VALUES

(1, 'Inception', 1, 1),

(2, 'The Big Lebowski', 2, 2),

(3, 'Pride and Prejudice', 3, 3),

(4, 'The Matrix', 1, 1),

(5, 'Jurassic Park', 2, 1),

(6, 'Monty Python and the Holy Grail', 3, 2),

(7, 'Interstellar', 1, 1),

(8, 'Hot Fuzz', 3, 2),

(9, 'The Godfather', 2, 3),

(10, 'Parasite', 3, 3),

(11, 'Dunkirk', 3, 1),

(12, 'Shaun of the Dead', 3, 2),

(13, 'The Imitation Game', 2, 3),

(14, 'The Shawshank Redemption', 2, 3),

(15, 'Forrest Gump', 1, 3);

INSERT INTO Users (UserId, UserName, UserPassword) VALUES

(1, 'johndoe', 'password123'),

(2, 'janedoe', 'password456'),

(3, 'alexsmith', 'password789'),

(4, 'mariagarcia', 'password234'),

(5, 'michaelbrown', 'password567'),

(6, 'emilyjohnson', 'password890'),

(7, 'williamjones', 'password345'),

(8, 'sophiewilson', 'password678'),

(9, 'oliverlee', 'password901'),

(10, 'isabellaharris', 'password1234');

INSERT INTO Reviews (ReviewId, Film, Author, Text, Rating) VALUES

(1, 1, 1, 'A mind-bending masterpiece!', 5.0),

(2, 2, 2, 'Hilarious from start to finish.', 4.0),

(3, 3, 3, 'A beautiful adaptation of a classic novel.', 4.5),

(4, 4, 1, 'A revolutionary sci-fi thriller.', 4.7),

(5, 5, 2, 'Dinosaurs on the big screen - incredible!', 4.8),

(6, 6, 3, 'Comedy gold, endlessly quotable.', 5.0),

(7, 7, 4, 'An epic journey through space and time.', 4.6),

(8, 8, 5, 'A perfect blend of action and humor.', 4.3),

(9, 9, 6, 'An unforgettable crime drama.', 5.0),

(10, 10, 7, 'A poignant and intense social satire.', 4.9),

(11, 11, 8, 'Visually stunning war film.', 4.5),

(12, 12, 9, 'Zombie comedy at its finest!', 4.7),

(13, 13, 10, 'Inspiring and captivating.', 4.4),

(14, 14, 1, 'A story of hope and resilience.', 4.9),

(15, 15, 2, 'Heartwarming and deeply moving.', 4.8),

(16, 1, 3, 'Thrilling and beautifully executed.', 4.9),

(17, 2, 4, 'Never gets old, pure comedy genius.', 4.8),

(18, 3, 5, 'Outstanding and visually impressive.', 4.6),

(19, 4, 6, 'Ahead of its time in many ways.', 4.7),

(20, 5, 7, 'The best sci-fi adventure ever.', 5.0);

INSERT INTO Persons (PersonId, FirstName, LastName, Age, Country) VALUES

(1, 'Leonardo', 'DiCaprio', 48, 1),

(2, 'Christopher', 'Nolan', 53, 1),

(3, 'Keanu', 'Reeves', 59, 1),

(4, 'Steven', 'Spielberg', 77, 1),

(5, 'Jeff', 'Bridges', 74, 1),

(6, 'Joel', 'Coen', 69, 1),

(7, 'Graham', 'Chapman', 48, 2),

(8, 'Quentin', 'Tarantino', 61, 1),

(9, 'Martin', 'Scorsese', 81, 1),

(10, 'Bong', 'Joon-ho', 55, 3),

(11, 'Cillian', 'Murphy', 47, 2),

(12, 'Simon', 'Pegg', 54, 2),

(13, 'Benedict', 'Cumberbatch', 48, 2),

(14, 'Tim', 'Robbins', 65, 1),

(15, 'Tom', 'Hanks', 68, 1);

INSERT INTO Contributions(Person, Film, Role) VALUES

(1, 1, 2),

(2, 1, 1),

(2, 7, 1),

(3, 4, 2),

(4, 5, 3),

(5, 2, 2),

(6, 2, 1),

(7, 6, 2),

(10, 10, 1),

(11, 11, 2),

(12, 12, 2),

(13, 13, 2),

(14, 14, 2),

(15, 15, 2);

INSERT INTO Films (FilmId, FilmName, Company, Genre) VALUES

(16, 'Terrible Movie 1', 1, 2),

(17, 'Worst Movie Ever', 2, 3),

(18, 'Disaster Film', 1, 1),

(19, 'Flop of the Century', 3, 2);

INSERT INTO Reviews (ReviewId, Film, Author, Text, Rating) VALUES

(21, 16, 1, 'Utterly disappointing. A waste of time!', 1.0),

(22, 17, 2, 'Poorly written, badly executed, just awful.', 1.5),

(23, 18, 3, 'Not even worth a one-star rating. Terrible!', 1.8),

(24, 19, 4, 'Boring and unoriginal, I wouldn’t recommend it to anyone.', 2.0);

INSERT INTO Films (FilmId, FilmName, Company, Genre) VALUES

(20, 'The Lost City', 1, 1),

(21, 'Moonlight Dreams', 2, 2);

INSERT INTO Persons (PersonId, FirstName, LastName, Age, Country) VALUES

(16, 'James', 'Cameron', 70, 1),

(17, 'Steven', 'Spielberg', 77, 1);

INSERT INTO Contributions (Person, Film, Role) VALUES

(16, 20, 1),

(17, 20, 1),

(16, 21, 1),

(17, 21, 1);

INSERT INTO Persons (PersonId, FirstName, LastName, Age, Country) VALUES

(21, 'Robert', 'Downey Jr.', 58, 1),

(22, 'Benedict', 'Wong', 52, 2),

(18, 'Chris', 'Hemsworth', 40, 3),

(19, 'Tom', 'Hiddleston', 42, 2),

(20, 'Ryan', 'Reynolds', 47, 3);

INSERT INTO Contributions (Person, Film, Role) VALUES

(21, 4, 2),

(22, 4, 2),

(18, 4, 2),

(19, 5, 2),

(20, 5, 2)

-- SELECTS

--1 Select all the films where Christopher Nolan is a contributor

select distinct \*

from Films

where FilmId in

(select Film

from Contributions

where Person =

(select PersonId

from Persons

where CONCAT(FirstName, ' ', LastName) = 'Christopher Nolan'))

--2 Select all reviews of a film

select \*

from Reviews

where Film =

(select FilmId

from Films

where FilmName = 'Inception')

--3 Select all films with ammount of their reviews and rating as HIGH, LOW, MEDIUM

select

FilmName,

COUNT(ReviewId) AS ReviewCount,

case

when AVG(Rating) < 2.5 then 'LOW'

when AVG(Rating) < 4 then 'MEDIUM'

else 'HIGH'

end as RatingCategory

from Reviews

join Films on Film = FilmId

group by FilmName

--4 Count films by countries

select

CountryName,

COUNT(Films.FilmId) as FilmCount

from Countries

join CinemaCompanies on Countries.CountryId = CinemaCompanies.Country

join Films on CinemaCompanies.CompanyId = Films.Company

group by Countries.CountryName

--5 Select top 5 most high rated films in USA

select top 5

Films.FilmName,

avg(Reviews.Rating)

from Films

join CinemaCompanies on CinemaCompanies.CompanyId = Films.Company

join Countries on Countries.CountryId = CinemaCompanies.Country

join Reviews on Reviews.Film = Films.FilmId

where Countries.CountryName = 'USA'

group by

Films.FilmName,

Countries.CountryName

order by AVG(Rating) desc

--6. Selecty films where was more than 1 director

select

FilmName,

COUNT(Contributions.Person) as Directors

from Films

join Contributions on Contributions.Film = Films.FilmId

where Contributions.Role = 1

group by Films.FilmName

having count (Contributions.Person) > 1

--7 Select top 5 users with most of the written reviews

select top 5

Users.UserName,

COUNT(Reviews.ReviewId) as ReviewCount

from Users

join Reviews on Reviews.Author = Users.UserId

group by Users.UserName

order by ReviewCount desc

--8 Select films along with the count of unique countries represented by individuals who contributed to the movies as Actors

select

Films.FilmName,

count(distinct Persons.Country) as CountryCount

from Films

join Contributions on Contributions.Film = Films.FilmId

join Persons on Persons.PersonId = Contributions.Person

where Contributions.Role = 2

group by Films.FilmName

having count(distinct Persons.Country) > 1

--9 Count films by genres

select

Genres.GenreName,

COUNT(Films.FilmId) as FilmCount

from Films

join Genres on Genres.GenreId = Films.Genre

group by Genres.GenreName

--10 Select average rating of films by genres

select

Genres.GenreName,

AVG(Reviews.Rating) as AverageRating

from Films

join Reviews on Films.FilmId = Reviews.Film

join Genres on Films.Genre = Genres.GenreId

group by Genres.GenreName

-- VIEWS

--1 Select film details

--select \* from FilmDetails

--drop view FilmDetails

create view FilmDetails as

select

f.FilmId,

f.FilmName,

c.CompanyName,

g.GenreName

from Films f

join CinemaCompanies c on f.Company = c.CompanyId

join Genres g on f.Genre = g.GenreId

--2 Select rivew details

--select \* from ReviewDetails

--drop view ReviewDetails

create view ReviewDetails as

select

r.ReviewId,

f.FilmName,

u.UserName as ReviewerName,

r.Rating

from Reviews r

join Films f on r.Film = f.FilmId

join Users u on r.Author = u.UserId

--3 Select contributions along with people and their role

--select \* from PersonContributions

--drop view PersonContributions

create view PersonContributions as

select

p.FirstName,

p.LastName,

f.FilmName,

r.RoleName

from Contributions c

join Persons p on c.Person = p.PersonId

join Films f on c.Film = f.FilmId

join Roles r on c.Role = r.RoleId

--4 Select top rated films with rating > 4.5

--select \* from TopRatedFilms

--drop view TopRatedFilms

create view TopRatedFilms as

select

f.FilmName,

AVG(r.Rating) as AverageRating

from Reviews r

join Films f on r.Film = f.FilmId

group by f.FilmName

having AVG(r.Rating) > 4.5

--5 Select films with companies and countries

--select \* from FilmCountries

--drop view FilmCountries

create view FilmCountries as

select f.FilmName, c.CountryName

from Films f

join CinemaCompanies cc on f.Company = cc.CompanyId

join Countries c on cc.Country = c.CountryId