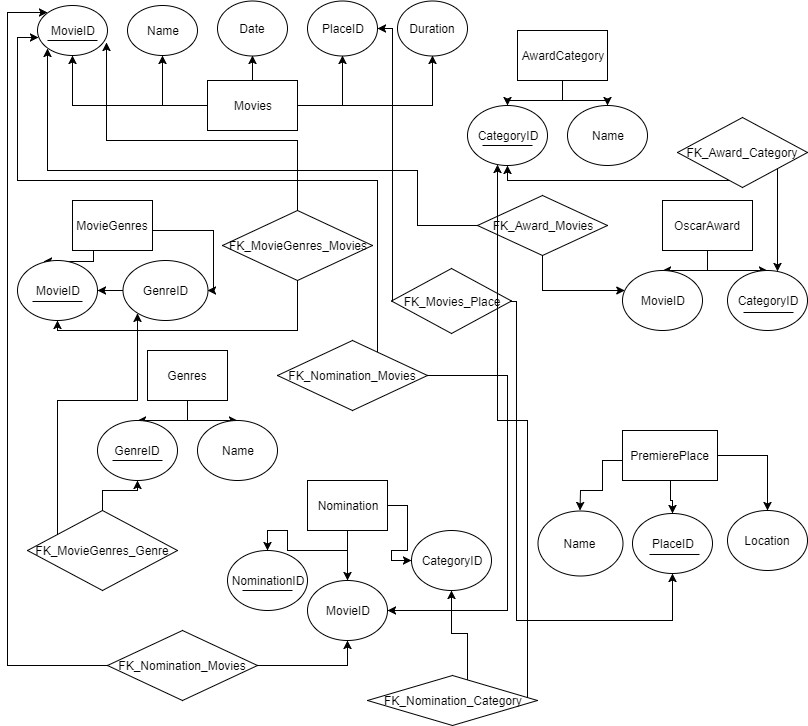
**Курсова по Бази данни – Драгомир Галинов Стойчев, Софтуерно инженерство, 2гр., 20055 вх. №**

**1.** **Описание на предметната област.**

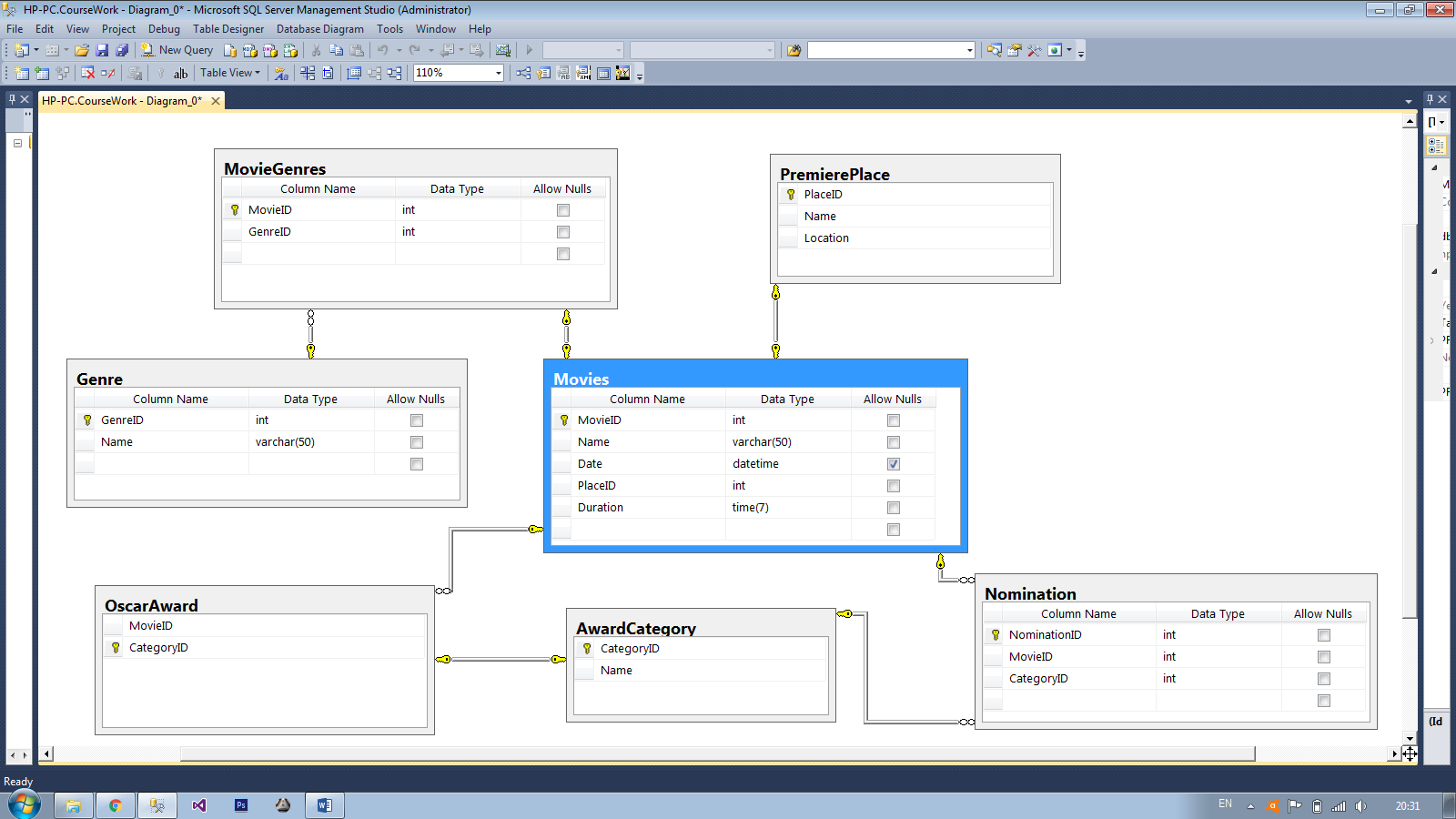
**-** Базата данни се отнася за филми, номинации, награди и място на провеждане на премиерата на съответния филм.

- Базата данни служи за набиране на специфична информация относно част от схемата във филмовата индустрия.

**2.** **ER-модел на базата от данни.**

****

**3. Релационен модел на базата от данни.**



**-таблица "Movies" определя номера на съответния филм(MovieID), неговото име(Name), дата на премиера(Date), място на премиерата(PlaceID, което съответства с PlaceID от таблица PremierePlace, които са взаимносвързани чрез FK\_Movies\_Place), времетраене на самият филм(Duration);**

**-таблица "Genre" определя номер(GenreID) и име на жанр(Name), които по-късно се използват(в "MovieGenres"), за да свържат определен филм(MovieID-FK\_MovieGenres\_Movies) със съответния за него жанр(GenreID-чрез FK\_MovieGenres\_Genre);**

**-таблица "AwardCategory" оказва определен номер(CategoryID) на филмова награда(Name), който по-късно служи за свързване на съответния филм със съответната награда, за която е номиниран(в "OscarАward"-чрез FK\_Award\_Category);**

**-таблица "OscarAward" свързва номера на съответния филм(MovieID) с номера на съответната категория(CategoryID), към която принадлежи;**

**-таблица "Nomination" номерира в филма(МovieID, чрез FK\_Nomination\_Movies) с неговата съответна категория(CategoryID-чрез FK\_Nomination\_Category);**

**-таблица "PremierePlace" задава съответния номер(PlaceID), името на мястото(Name) и държавата, в което е самото мероприятие(Location), където ще се проведе съответния филм; тази информация играе роля в колоната "PlaceID"(чрез FK\_Movies\_Place) в таблица "Movies";**

**4. Реализация на релационния модел на базата от данни.**

**Movies таблица:**

**-Създаване на таблиците + ограничения към тях.**

**dbo.Movies:**

CREATE TABLE [dbo].[Movies](

[MovieID] [int] NOT NULL,

[Name] [varchar](50) NOT NULL UNIQUE,

[Date] [datetime] NULL,

[PlaceID] [int] NOT NULL UNIQUE,

[Duration] [time](7) NOT NULL,

CONSTRAINT [PK\_Movies] PRIMARY KEY CLUSTERED ([MovieID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

CONSTRAINT [MovieNameUnique] UNIQUE NONCLUSTERED ([Name] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**dbo.Genre:**

CREATE TABLE [dbo].[Genre](

[GenreID] [int] NOT NULL,

[Name] [varchar](50) NOT NULL,

CONSTRAINT [PK\_Genre] PRIMARY KEY CLUSTERED ([GenreID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

CONSTRAINT [GenreNameUnique] UNIQUE NONCLUSTERED ([Name] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**dbo.MovieGenres:**

CREATE TABLE [dbo].[MovieGenres](

[MovieID] [int] NOT NULL,

[GenreID] [int] NOT NULL,

CONSTRAINT [PK\_MovieGenres] PRIMARY KEY CLUSTERED ([MovieID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

CONSTRAINT [IX\_MovieGenres] UNIQUE NONCLUSTERED ([GenreID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**dbo.AwardCategory:**

CREATE TABLE [dbo].[AwardCategory](

[CategoryID] [int] NOT NULL,

[Name] [varchar](50) NOT NULL,

CONSTRAINT [PK\_Category] PRIMARY KEY CLUSTERED ([CategoryID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

CONSTRAINT [NameCategoryUnique] UNIQUE NONCLUSTERED ([Name] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**dbo.OscarAward:**

CREATE TABLE [dbo].[OscarAward](

[MovieID] [int] NOT NULL,

[CategoryID] [int] NOT NULL,

CONSTRAINT [PK\_Award] PRIMARY KEY CLUSTERED ([MovieID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**dbo.Nomination:**

CREATE TABLE [dbo].[Nomination](

[NominationID] [int] NOT NULL,

[MovieID] [int] NOT NULL,

[CategoryID] [int] NOT NULL,

CONSTRAINT [PK\_Nomination] PRIMARY KEY CLUSTERED ([NominationID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

ALTER TABLE [dbo].[Nomination] WITH CHECK ADD CONSTRAINT [FK\_Nomination\_Category] FOREIGN KEY([CategoryID])

REFERENCES [dbo].[Category] ([CategoryID])

ALTER TABLE [dbo].[Nomination] CHECK CONSTRAINT [FK\_Nomination\_Category]

ALTER TABLE [dbo].[Nomination] WITH CHECK ADD CONSTRAINT [FK\_Nomination\_Movies] FOREIGN KEY([MovieID])

REFERENCES [dbo].[Movies] ([MovieID])

ALTER TABLE [dbo].[Nomination] CHECK CONSTRAINT [FK\_Nomination\_Movies]

**dbo.PremierePlace**

CREATE TABLE [dbo].[PremierePlace](

[PlaceID] [int] NOT NULL,

[Name] [varchar](50) NOT NULL,

[Location] [varchar](50) NOT NULL,

CONSTRAINT [PK\_Place] PRIMARY KEY CLUSTERED ([PlaceID] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY],

CONSTRAINT [PlaceNameUnique] UNIQUE NONCLUSTERED ([Name] ASC)

WITH (IGNORE\_DUP\_KEY = OFF,

ALLOW\_ROW\_LOCKS = ON,

ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]);

**-Вмъкване на информация за таблиците и извършване на (INSERT, DELETE, UPDATE) функции:**

-INSERT функция:

INSERT INTO dbo.Movies

(MovieID, Name, Date, PlaceID, Duration)

VALUES

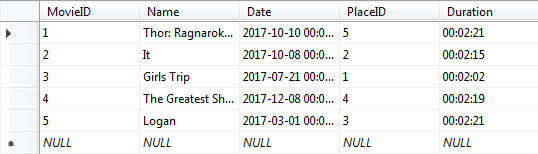
(1, 'Wonder Woman', '2017-06-15', '5', '02:21'),

(2, 'It', '2017-10-08', '2', '02:15'),

(3, 'Girls Trip', '2017-12-20', '1', '02:02'),

(4, 'The Greatest Showman', ‘2017-12-08’, '4', '02:19'),

(5, 'Logan', '2017-03-01', '3', '02:21');



-DELETE функция:

DELETE FROM dbo.Movies

WHERE Name ='Wonder Woman';

-UPDATE функция:

UPDATE dbo.Movies

SET Name = 'Thor: Ragnarok (2017)', Date='2017-10-10'

WHERE MovieID = '5';

**Genre таблица:**

-INSERT функция:

INSERT INTO dbo.Genre

(GenreID, Name)

VALUES

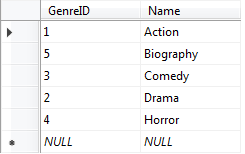
(1, 'Action'),

(2, 'Drama'),

(3, 'Comedy'),

(4, 'Horror'),

(5, 'Biography');



-DELETE функция:

DELETE FROM dbo.Genre

WHERE Name = 'Biography';

-UPDATE функция:

UPDATE dbo.Genre

SET Name = 'Cartoon'

WHERE MovieID = '2';

**MovieGenres** таблица:

-INSERT функция:

INSERT INTO dbo.MovieGenres

(MovieID, GenreID)

VALUES

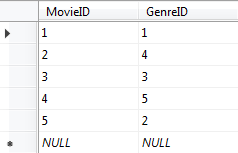
(1, 1),

(2, 4),

(3, 3),

(4, 5),

(5, 2);



-DELETE функция:

DELETE FROM dbo.MovieGenres

WHERE MovieID='3';

-UPDATE функция:

UPDATE dbo.MovieGenres

SET MovieID= '4', GenreID= '1'

WHERE MovieID = 3;

**AwardCategory** таблица:

-INSERT функция:

INSERT INTO dbo.AwardCategory

(CategoryID, Name)

VALUES

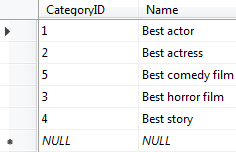
(1, 'Best actor'),

(2, 'Best actress'),

(5, 'Best comedy film'),

(3, 'Best horror film'),

(4, 'Best story');



-DELETE функция:

DELETE FROM dbo.AwardCategory

WHERE CategoryID='5';

-UPDATE функция:

UPDATE dbo.AwardCategory

SET Name = 'Best adventure film'

WHERE CategoryID = 1;

**Nomination** таблица:

-INSERT функция:

INSERT INTO dbo.Nomination

(NominationID, MovieID, CategoryID)

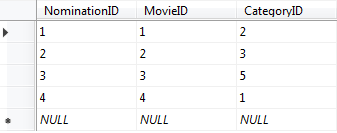
VALUES

(1, 1, 2),

(2, 2, 3),

(3, 3, 5),

(4, 4, 1);



-DELETE функция:

DELETE FROM dbo.Nomination

WHERE NominationID='3';

-UPDATE функция:

UPDATE dbo.Nomination

SET CategoryID = '4'

WHERE CategoryID = 4;

**OscarAward** таблица:

-INSERT функция:

INSERT INTO dbo.OscarAward

(MovieID, CategoryID)

VALUES

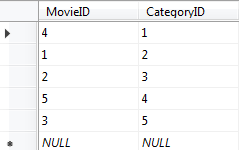
(1, 2),

(2, 3),

(3, 5),

(4, 1),

(5, 4);



-DELETE функция:

DELETE FROM dbo.OscarAward

WHERE MovieID='1';

-UPDATE функция:

UPDATE dbo.OscarAward

SET CategoryID = '1'

WHERE CategoryID = 5;

**PremierePlace** таблица:

-INSERT функция:

INSERT INTO dbo.PremierePlace

(PlaceID, Name, Location)

VALUES

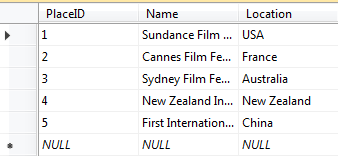
(1, 'Sundance Film Festival', 'USA'),

(2, 'Cannes Film Festival', 'France'),

(3, 'Sydney Film Festival', 'Australia'),

(4, ' New Zealand International Film Festival', ‘New Zealand’),

(5, 'First International Film Festival Xining', 'China');



-DELETE функция:

DELETE FROM dbo.PremierePlace

WHERE PlaceID = '4';

-UPDATE функция:

UPDATE dbo.PremierePlace

SET Name = 'Austin Film Festival', Location = 'USA'

WHERE CategoryID = 5;

**5. Създаване на заявки:**

*-заявки към повече от една релации:*

SELECT Name

FROM dbo.Movies

WHERE MovieID <= '3';

*Резултат:(извежда имената всички филми(“Name”) с “MovieID” по-малко от 3)*

C:\Users\HP\Desktop\Screenshot_4.png

*-заявки, включващи обобщаващи функции (като SUM, COUNT, AVG, MIN, MAX):*

SELECT COUNT (PlaceID)

FROM dbo.Movies;

*Резултат:(изброява резултатите с ID по-голямо от 1)*



SELECT Max(Duration)

AS LongestFilm

FROM dbo.Movies;

*Резултат:(извежда продължението на най-дългия филм)*

*C:\Users\HP\Desktop\Screenshot_3.png*

SELECT Min(Date)

AS EarliestDate

FROM dbo.Movies;

*Резултат:(извежда най-ранната дата на премиера от Movies)*

*C:\Users\HP\Desktop\Screenshot_2.png*

*-заявки, включващи GROUP BY, HAVING:*

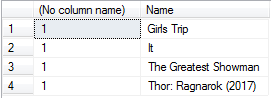
SELECT COUNT(MovieID), Name

FROM dbo.Movies

GROUP BY Name

HAVING COUNT(MovieID) <= 4;

*Резултат:(извежда имената на филми с MovieID по-малко от 4 и ги групира в азбучен ред)*



*-заявки, включващи външни съединения и/или подзаявки:*

SELECT MovieID, GenreID, Duration

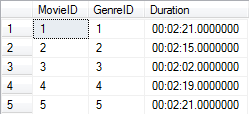
FROM dbo.Movies

INNER JOIN dbo.Genre

ON dbo.Movies.MovieID = dbo.Genre.GenreID;

GO

*Резултат:(извежда номерът на филм и продължението на съответния филм от Movies и вмъква информацията от Genre, като информацията от MovieID се приравнява към тази от GenreID)*



*-заявки, които изискват използването на изрази - CASE изрази:*

SELECT MovieID, Name, Date, Duration

FROM dbo.Movies

ORDER BY

(CASE

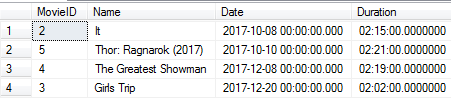
WHEN Date IS NULL THEN Duration

ELSE Date

END);

GO

*Резултат:(извежда колони MovieID, Name, Date, Duration от Movies и задава условие - Ако колоната Date е зададена като NULL, тогава информацията се групира по продължение(Duration) на филмите, ако не, тогава се групират спрямо датата(Date))*



**6. Създаване на:**

*-изгледи:*

CREATE VIEW [Movie] AS

SELECT MovieID, Name

FROM dbo.Movies

WHERE MovieID >= 1;

*Резултат:(създава се гледка, която се състои от колоните MovieID и Name от таблица Movies, където номерът на филма трябва да е 1 или повече от 1)*

C:\Users\HP\Desktop\Screenshot_5.png

*-съхранени процедури и/или потребителски дефинирани функции:*

CREATE PROCEDURE MovieInfo

AS

SELECT Name AS MovieName,

MovieID AS FollowingMovie,

Duration AS MovieDuration,

Date AS DateOfFirstProjection

FROM dbo.Movies

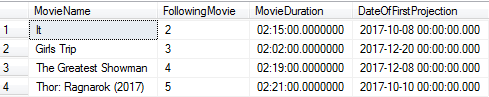
ORDER BY FollowingMovie

GO

EXECUTE MovieInfo

*Резултат:(създава процедура на име MovieInfo, където Name приема името FollowingMovie, Duration приема име MovieDuration, Date приема име DateOfFirstProjection, които се извеждат от таблица Movies и информацията се разделя по номер на филм чрез MovieID или FollowingMovie)*

C:\Users\HP\Desktop\Screenshot_5.png



*-тригери:*

CREATE TRIGGER MovieCheck

ON dbo.Movies

AFTER INSERT, DELETE, UPDATE

AS RAISERROR ('Notify Customer Relations', 16, 10);

GO

*Резултат:(Създава се тригер MovieCheck, базиран на информацията от таблица Movies. Ако се извърши промяна на информация чрез функции(INSERT, DELETE, UPDATE), се извежда съобщение, което осведомява потребителя за съответните направени промени)*

C:\Users\HP\Desktop\Screenshot_5.png

**7. Бъдеща работа**

*-описване на възможните посоки на бъдещо развитие на разработеното приложение: Съхраняване на по-подробна информация спрямо филмовата индустрия.*