



SAKARYA
ÜNİVERSİTESİ

**DATABASE MANAGEMENT SYSTEMS
PROJECT**

B201202032
Habibe AYDIN

habibe.aydin@ogr.sakarya.edu.tr

Introduction

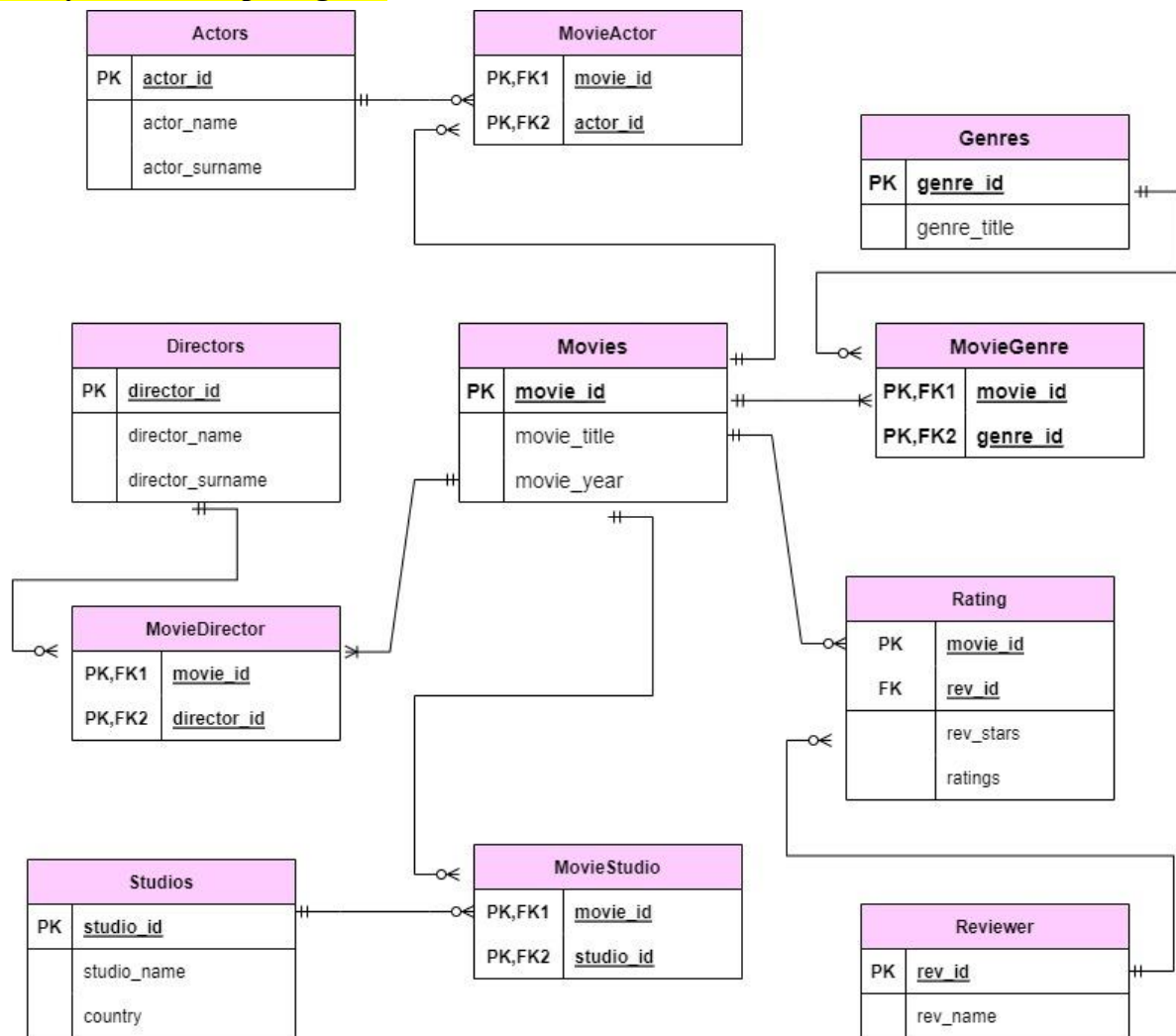
The Movie Database is an database containing information and statistics about movies, as well as actors, directors. This information can include lists of cast, movie release years, actor and director info.

Users can also submit reviews of movies and TV shows on a one to ten scale, which are then used to create a weighted mean of all user reviews to be displayed on the movie or TV show's page.

Business Rules

- * Each movie has an id, movie's title and movie's release year. Movies are distinguished from each other by their id.
- * Each actor has an id, name and surname info. Each actor has a different id.
- * Each Actor can act in no movie or many movies.
- * Each movie must be acted by at least one actor or many actors.
- * Each director has an id, name and surname info. Each one has a different id.
- * Each director can direct no movie or many movies.
- * Each movie must be directed by at least one director or many directors.
- * Each genre has an id that is distinguished from each other.
- * Each movie can contain one or many genres.
- * Each genre can be found in no movie or many movies.
- * Each studio has an id, name and country info. Each one has a different id.
- * Each movie can be in no studio or many studios.
- * Each studio can have no movie or many movies.
- * Each reviewer has an id and name.
- * Each rating has a movieID, reviewerID, stars and ratings number.
- * Each reviewer can rate no movie or many movies.
- * Each movie can be rated by no reviewer or many reviewers.

Entity Relationship Diagram



Relational Model

- * Movies(movie_id:int, movie_title:varchar, release_year:int)
- * Actors(actor_id:int, actor_name:varchar, actor_surname:varchar)
- * MovieActor(movie_id:int, actor_id:int)
- * Directors(director_id:int, director_name:varchar)
- * MovieDirector(movie_id:int, director_id:int)
- * Genres(genre_id:int, genre_title:varchar)
- * MovieGenre(movie_id:int, genre_id:int)
- * Studios(studio_id:int, studio_name:varchar, country:varchar)
- * MovieStudio(movie_id:int, studio_id:int)
- * Reviewer(rev_id:int, rev_name:varchar)
- * Rating(movie_id:int, rev_id:int, rev_stars:int, ratings:int)

SQL statements

Creating the tables

```
CREATE TABLE "Movies" (  
    "movie_id" INT NOT NULL,  
    "movie_title" VARCHAR(100),  
    "release_year" SMALLINT,  
    CONSTRAINT "moviePK" PRIMARY KEY ("movie_id")  
);
```

```
CREATE TABLE "Actors" (  
    "actor_id" INT NOT NULL,  
    "actor_name" VARCHAR(100),  
    "actor_surname" VARCHAR(100),  
    CONSTRAINT "actorPK" PRIMARY KEY ("actor_id")  
);
```

```
CREATE TABLE "Directors" (  
    "director_id" INT NOT NULL,  
    "director_name" VARCHAR(100),  
    "director_surname" VARCHAR(100),  
    CONSTRAINT "directorPK" PRIMARY KEY ("director_id")  
);
```

```
CREATE TABLE "Studios" (  
    "studio_id" INT NOT NULL,  
    "studio_name" VARCHAR(100),  
    "country" VARCHAR(100),  
    CONSTRAINT "studioPK" PRIMARY KEY ("studio_id")  
);
```

```
CREATE TABLE "Genres" (  
    "genre_id" INT NOT NULL,  
    "genre_title" VARCHAR(100),  
    CONSTRAINT "genrePK" PRIMARY KEY ("genre_id")  
);
```

```
CREATE TABLE "Reviewer" (  
    "rev_id" INT NOT NULL,  
    "rev_name" VARCHAR(100),  
    CONSTRAINT "revPK" PRIMARY KEY ("rev_id")  
);
```

```
CREATE TABLE "MovieActor" (  
    "movie_id" INT,  
    "actor_id" INT,  
    CONSTRAINT "movieFK" FOREIGN KEY ("movie_id") REFERENCES "Movies"  
    ("movie_id") ON DELETE CASCADE ON UPDATE CASCADE,
```

```
        CONSTRAINT "actorFK" FOREIGN KEY ("actor_id") REFERENCES "Actors"
("actor_id") ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE "MovieGenre" (
    "movie_id" INT,
    "genre_id" INT,
    CONSTRAINT "movieFK" FOREIGN KEY ("movie_id") REFERENCES "Movies"
("movie_id") ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT "genreFK" FOREIGN KEY ("genre_id") REFERENCES "Genres"
("genre_id") ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE "MovieDirector" (
    "movie_id" INT,
    "director_id" INT,
    CONSTRAINT "movieFK" FOREIGN KEY ("movie_id") REFERENCES "Movies"
("movie_id") ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT "directorFK" FOREIGN KEY ("director_id") REFERENCES "Directors"
("director_id") ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE "MovieStudio" (
    "movie_id" INT,
    "studio_id" INT,
    CONSTRAINT "movieFK" FOREIGN KEY ("movie_id") REFERENCES "Movies"
("movie_id") ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT "studioFK" FOREIGN KEY ("studio_id") REFERENCES "Studios"
("studio_id") ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE "Ratings" (
    "movie_id" INT,
    "rev_id" INT,
    "rev_stars" INT,
    "rating" INT,
    CONSTRAINT "movieFK" FOREIGN KEY ("movie_id") REFERENCES "Movies"
("movie_id") ON DELETE CASCADE ON UPDATE CASCADE,
    CONSTRAINT "rev_id" FOREIGN KEY ("rev_id") REFERENCES "Reviewer"
("rev_id") ON DELETE CASCADE ON UPDATE CASCADE
);
```

Inserting the values into the tables

```
INSERT INTO "Movies" ("movie_id","movie_title","release_year") VALUES
('101','Agora','2009'),
('202','Divergent','2014'),
('303','Thor','2011'),
('404','Maleficent','2014'),
('505','Focus','2015'),
('606','Insomnia','2002'),
('707','Pirates of the Caribbean','2003'),
('808','Lucy','2014'),
('909','Bus Stop','1956'),
('1010','Titanic','1997'),
('1111','The Post','2017'),
('1212','War and Peace','1958')
```

```
INSERT INTO "Actors" ("actor_id", "actor_name", "actor_surname") VALUES
('1001','Audrey', 'Hepburn'),
('1010','Tom', 'Hanks'),
('1100','Leonardo', 'DiCaprio'),
('2002','Kate', 'Winslet'),
('2020','Marilyn', 'Monroe'),
('2200','Morgan', 'Freeman'),
('3003','Johnny', 'Depp'),
('3030','Al', 'Pacino'),
('3300','Will', 'Smith'),
('4004','Angelina', 'Jolie'),
('4040','Natalie', 'Portman'),
('4400','Theo', 'James'),
('5005','Rachel', 'Weisz'),
('5050','Oscar', 'Isaac'),
('5500','Shailene', 'Woodley')
```

```
INSERT INTO "Directors" ("director_id", "director_name", "director_surname") VALUES
('111','King','Vidor'),
('222','Steven','Spielberg'),
('333','Alejandro','Amenabar'),
('444','James','Kameron'),
('555','Neil','Burger'),
('666','Joshua', 'Logan'),
('777','Robert','Stromberg'),
('888','Luc','Besson'),
('999','Kenneth','Branagh'),
('1112','Glenn','Fiacarra'),
('1113','Rob','Marshall'),
('1114','Christopher','Nolan')
```

```
INSERT INTO "Studios" ("studio_id","studio_name","country") VALUES
('2201','Arborfield Studios','England'),
('2202','Bovingdon Airfield Studios','Germany'),
('2203','Bray Film Studios','Italy'),
('2204','Vision Film Studios','USA'),
('2205','Cardington Studios','England'),
('2206','Church Fenton Studios','France'),
('2207','Elstree Studios','USA'),
('2208','Farnborough Film Studios','Spain'),
('2209','Hoddesdon Studios','Japan'),
('2210','Star Film Studios','USA')
```

```
INSERT INTO "Genres" ("genre_id","genre_title") VALUES
('001','Action'),
('002','Adventure'),
('003','Animation'),
('004','Biography'),
('005','Comedy'),
('006','Crime'),
('007','Drama'),
('008','Horror'),
('009','Music'),
('010','Mystery'),
('011','Romance'),
('012','Thriller'),
('013','War')
```

```
INSERT INTO "Reviewer" ("rev_id","rev_name") VALUES
('9001','Righty Sock'),
('9002','Jack Malvern'),
('9003','Flagrant Baronessa'),
('9004','Alec Shaw'),
('9005','Victor Wöltje'),
('9006','Simon Wright'),
('9007',''),
('9008','Paul Monks'),
('9009','Mike Salvati'),
('9010','Wesley S. Walker'),
('9011','Sasha Goldshtein'),
('9012','Josh Cates'),
('9013',''),
('9014','Scott LeBrun'),
('9015','Hannah Steele'),
('9016','Vincent Cadena'),
('9017','Brandt Sponseller'),
('9018','Richard Adams'),
('9019','Neal Wruck'),
('9020','Krug Stillo')
```

```
INSERT INTO "MovieActor" ("movie_id","actor_id") VALUES
('101','5005'),
('101','5050'),
('202','4400'),
('202','5500'),
('303','4040'),
('404','4004'),
('505','3300'),
('606','3030'),
('707','3003'),
('808','2200'),
('909','2020'),
('1010','1100'),
('1010','2002'),
('1111','1010'),
('1212','1001')
```

```
INSERT INTO "MovieGenre" ("movie_id","genre_id") VALUES
('101','007'),
('202','001'),
('303','010'),
('404','002'),
('505','006'),
('606','012'),
('707','001'),
('808','004'),
('909','005'),
('1010','011'),
('1111','012'),
('1212','013')
```

```
INSERT INTO "MovieDirector" ("movie_id","director_id") VALUES
('101','333'),
('202','555'),
('303','999'),
('404','777'),
('505','1112'),
('606','1114'),
('707','1113'),
('808','888'),
('909','666'),
('1010','444'),
('1111','222'),
('1212','111')
```



```
INSERT INTO "MovieStudio" ("movie_id","studio_id") VALUES
('101','2201'),
('202','2202'),
('303','2203'),
('404','2204'),
('505','2205'),
('606','2206'),
('707','2207'),
('808','2208'),
('909','2209'),
('1010','2210'),
('1111','2208'),
('1212','2204')
```

```
INSERT INTO "Ratings" ("movie_id","rev_id","rev_stars","rating") VALUES
('101','9001','8','263575'),
('202','9002','7','20207'),
('303','9003','8','202778'),
('404','9004','7','779489'),
('505','9005','7','227235'),
('606','9006','6','195961'),
('707','9007','8','203875'),
('808','9008','8','862618'),
('909','9009','4','642132'),
('1010','9010','6','609451'),
('1111','9011','3','511613'),
('1212','9012','8','667758')
```

Functions/Stored Procedure

(1)

```
CREATE OR REPLACE FUNCTION searchMovie(movieID INT)
RETURNS TABLE(id INT, title VARCHAR(10), year INT)
LANGUAGE plpgsql
AS
$$
BEGIN
    RETURN QUERY SELECT "movie_id", "movie_title", "release_year" FROM
    "Movies"
        WHERE "movie_id" = movieID;
END;
$$
```

Query Query History

```
1 CREATE OR REPLACE FUNCTION searchMovie(movieID INT)
2 RETURNS TABLE(id INT, title VARCHAR(10), year INT)
3 LANGUAGE plpgsql
4 AS
5 $$
6 BEGIN
7     RETURN QUERY SELECT "movie_id", "movie_title", "release_year" FROM "Movies"
8         WHERE "movie_id" = movieID;
9 END;
10 $$
11
12 SELECT * FROM searchMovie(101);
```

Data Output Messages Notifications

	id integer	title character varying	year integer
1	101	Agora	2009

(2)

```
SELECT * FROM "Ratings" WHERE "rating">600000
```

Query Query History

```
1 SELECT * FROM "Ratings" WHERE "rating">600000;
```









Data Output Messages Notifications

	movie_id integer	rev_id integer	rev_stars integer	rating integer
1	404	9004	7	779489
2	808	9008	8	862618
3	909	9009	4	642132
4	1010	9010	6	609451
5	1212	9012	8	667758

(3)

```
SELECT * FROM "Actors"  
WHERE "actor_name" LIKE '%a%'
```









Query		Query History	
1	SELECT	*	FROM "Actors"
2	WHERE	"actor_name"	LIKE '%a%'

Data Output		Messages	Notifications
			
			
	actor_id [PK] integer	actor_name character varying (100)	actor_surname character varying (100)
1	1100	Leonardo	DiCaprio
2	2002	Kate	Winslet
3	2020	Marilyn	Monroe
4	2200	Morgan	Freeman
5	4004	Angelina	Jolie
6	4040	Natalie	Portman
7	5005	Rachel	Weisz
8	5050	Oscar	Isaac
9	5500	Shailene	Woodley

(4)

```
SELECT * FROM "Directors"  
ORDER BY "director_surname" ASC
```

Query		Query History	
1	SELECT	*	FROM "Directors"
2	ORDER BY	"director_surname"	ASC;

Data Output		Messages	Notifications
			
			
	director_id [PK] integer	director_name character varying (100)	director_surname character varying (100)
1	333	Alejandro	Amenabar
2	888	Luc	Besson
3	999	Kenneth	Branagh
4	555	Neil	Burger
5	1112	Glenn	Fiacarra
6	444	James	Kameron
7	666	Joshua	Logan
8	1113	Rob	Marshall
9	1114	Christopher	Nolan
10	222	Steven	Spielberg
11	777	Robert	Stromberg
12	111	King	Vidor

Screenshots of Search, Insert, Delete, Update operations

The screenshot shows a window titled "Form1" with a gray background. On the left, the text "MovieDB" is displayed above a large, empty rectangular box. On the right, there are four input fields with labels: "Table Name" (a dropdown menu), "Movie ID", "Movie Title", and "Release Year". Below these fields are four buttons stacked vertically: "Search", "Insert", "Delete", and "Update".

Search Operation

This screenshot shows the same "Form1" window as the previous one, but with the "Table Name" dropdown menu open. The dropdown list displays several options: "Movies", "Actors", "Genres", "Directors", "Studios", "Reviewer", "Ratings", "MovieActor", "MovieDirector", "MovieGenre", and "MovieStudio". The "Movies" option is currently selected and highlighted in blue. The other input fields and buttons remain visible and unchanged.

Form1

MovieDB

movie_id	movie_title	rel
101	Agora	200
202	Divergent	201
303	Thor	201
404	Maleficent	201
505	Focus	201
606	Insomnia	200
707	Pirates of the Cari...	200
808	Lucy	201
909	Bus Stop	195
1010	Titanic	195
1111	The Post	201

Table Name Movies

Movie ID

Movie Title

Release Year

Search *

Insert

Delete

Update

Insert Operation

Form1

MovieDB

movie_id	movie_title	rel
303	Thor	201
404	Maleficent	201
505	Focus	201
606	Insomnia	200
707	Pirates of the Cari...	200
808		
909		
1010		
1111		
1212		

Table Name "Movies"

Movie ID 1313

Movie Title Hachiko

Release Year 2009

Search

Insert *

Delete

Update

Insert operation has been done successfully.

Tamam

Form1

MovieDB

movie_id	movie_title	rel
404	Maleficent	201
505	Focus	201
606	Insomnia	200
707	Pirates of the Cari...	200
808	Lucy	201
909	Bus Stop	195
1010	Titanic	195
1111	The Post	201
1212	War	200
1313	Hachiko	200

* 1313 Hachiko 200

Table Name: "Movies" ▾

Movie ID: 1313

Movie Title: Hachiko

Release Year: 2009

Search

Insert

Delete

Update

Delete Operation

Form1

MovieDB

movie_id	movie_title	rel
404	Maleficent	201
505	Focus	201
606	Insomnia	200
707	Pirates of the Cari...	200
808	Lucy	201
909		
1010		
1111		
1212		
1313		

* 1313

Table Name: "Movies" ▾

Movie ID: 1010

Movie Title:

Release Year:

Search

Insert

Delete *

Update

Delete operation has been done successfully.

Tamam

Form1

MovieDB

	movie_id	movie_title	rel
	303	Thor	201
	404	Maleficent	201
	505	Focus	201
	606	Insomnia	200
	707	Pirates of the Cari...	200
	808	Lucy	201
*	909	Bus Stop	195
	1111	The Post	201
	1212	War	200
	1313	Hachiko	200
*			

Table Name: "Movies"

Movie ID: 1010

Movie Title:

Release Year:

Search

Insert

Delete

Update

Update Operation

Form1

MovieDB

	movie_id	movie_title	rel
	303	Thor	201
	404	Maleficent	201
	505	Focus	201
	606	Insomnia	200
	707	Pirates of the Cari...	200
	808	Lucy	201
	909	Bus Stop	195
	1111	The Post	201
	1212	War	200
	1313	Hachiko	200
*			

Table Name: "Movies"

Movie ID: 1212

Movie Title: War and Peace

Release Year: 2005

Search

Insert

Delete

Update

Form1

MovieDB

movie_id	movie_title	release_year
303	Thor	2011
404	Maleficent	2014
505	Focus	2015
606	Insomnia	2002
707	Pirates of the Caribbean	2003
808		
909		
1111		
1212		
1313		

Table Name "Movies"

Movie ID 1212

Movie Title War and Peace

Release Year 2005

Search

Insert

Delete

Update

Update operation has been done successfully.

Form1

MovieDB

movie_id	movie_title	release_year
303	Thor	2011
404	Maleficent	2014
505	Focus	2015
606	Insomnia	2002
707	Pirates of the Caribbean	2003
808	Lucy	2014
909	Bus Stop	1956
1111	The Post	2017
1313	Hachiko	2009
1212	War and Peace	2005

Table Name "Movies"

Movie ID 1212

Movie Title War and Peace

Release Year 2005

Search

Insert

Delete

Update

The source codes of the application

```
Form1.cs [Design]
moviedatabase moviedatabase.Form1 Form10

1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10 using Npgsql;
11
12 namespace moviedatabase
13 {
14     3 references
15     public partial class Form1 : Form
16     {
17         1 reference
18         public Form1()
19         {
20             InitializeComponent();
21
22             NpgsqlConnection connect = new NpgsqlConnection("server=localhost;port=5432;UserId=postgres;password=habibe;database=mdb");
23
24         0 references
25         private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
26         {
27         }
28
29         1 reference
30         private void btn_list_Click(object sender, EventArgs e)
31         {
32             string table = comboBox1.Text;
33             string query = "select * from" + table;
34             NpgsqlDataAdapter adapt = new NpgsqlDataAdapter(query, connect);
35             DataSet ds = new DataSet();
36             adapt.Fill(ds);
37             dataGridView1.DataSource = ds.Tables[0];
38         }
39     }
40 }
```

```
Form1.cs* [Design]*
moviedatabase moviedatabase.Form1 textBox1_TextChanged(object sender, EventArgs e)

37
38     1 reference
39     private void btn_insert_Click(object sender, EventArgs e)
40     {
41         connect.Open();
42         NpgsqlCommand command1 = new NpgsqlCommand("insert into \"Movies\" values(@movie_id, @movie_title, @release_year)", connect);
43         command1.Parameters.AddWithValue("@movie_id", int.Parse(textBox_id.Text));
44         command1.Parameters.AddWithValue("@movie_title", textBox_title.Text);
45         command1.Parameters.AddWithValue("@release_year", int.Parse(textBox_year.Text));
46         command1.ExecuteNonQuery();
47         connect.Close();
48         MessageBox.Show("Insert operation has been done successfully.");
49     }
50
51     1 reference
52     private void btn_delete_Click_1(object sender, EventArgs e)
53     {
54         connect.Open();
55         NpgsqlCommand command2 = new NpgsqlCommand("delete from \"Movies\" where \"movie_id\"=@movie_id", connect);
56         command2.Parameters.AddWithValue("@movie_id", textBox_id.Text);
57         command2.ExecuteNonQuery();
58         connect.Close();
59         MessageBox.Show("Delete operation has been done successfully.");
60     }
61
62     1 reference
63     private void btn_update_Click_1(object sender, EventArgs e)
64     {
65         connect.Open();
66         NpgsqlCommand command3 = new NpgsqlCommand("update \"Movies\" set \"movie_title\"=@movie_title , " +
67             "\"release_year\"=@release_year where \"movie_id\"=@movie_id", connect);
68         command3.Parameters.AddWithValue("@movie_title", textBox_title.Text);
69         command3.Parameters.AddWithValue("@release_year", int.Parse(textBox_year.Text));
70         command3.Parameters.AddWithValue("@movie_id", textBox_id.Text);
71         command3.ExecuteNonQuery();
72         connect.Close();
73         MessageBox.Show("Update operation has been done successfully.");
74         connect.Close();
75     }
76 }
```