Arman YAYCI B201202050
Ahmet Eray KARADAĞ B201202021
Rıdvan MUSAOĞLU B201202062

Problem

The movie/TV series viewing activity, which is one of the most up-to-date activities of today, is one of the most frequently performed activities all over the world. Choosing the series and movies to watch is a common problem for all of us. The choice of series or movies according to our time, a lot of options, such as content with actors we like, can be a factor for the movie we will watch. In this case, developing an application for convenience in the selection part and designing a helpful interface for selecting can help everyone.
In addition to these, the critical thoughts of the series to be selected by other people to create interaction between people also provide assistance in the selection by providing an objective perspective.

Business Rules

Scenario

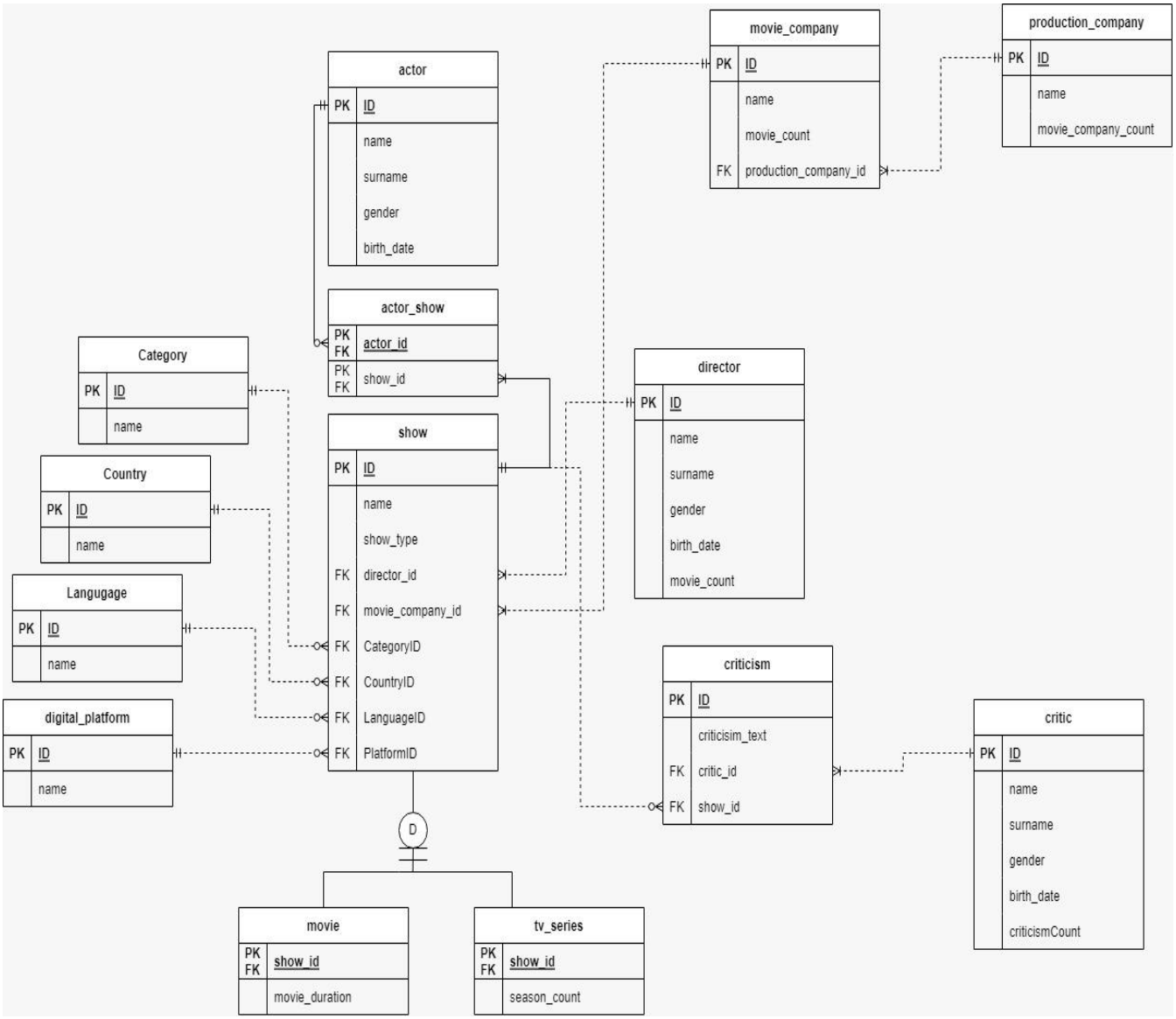
- It is expected that the database will contain information about the movies and tv shows, the publishers of these movies and shows, directors, companies, and critics about the show.

Business Rules

- Each category has a unique id, name and related movie count.
- Each actor has a unique id, name, surname, gender and birth date.
- Each director has a unique id, name, surname, gender and birth date.
- Each movie company has a unique id, name and related movie company count.
- Each production company has an id, name and related movie count.
- Each criticism has an id, criticism.
- Each critic has an id, name, surname and birth date.
- Each language has an id and name.
- Each country has an id and name.
- Each platform has an id and name.
- Show has id, name and show type.
- Movie has id and show id.
- TV Series has id and show id.
- A show has at least one category or many categories, a category may not have a show or many shows.
- An actor can play at one show or many shows, a show has an actor or many actors.
- A director can direct one show or many shows, a show can be directed by an actor.
- Each movie belongs to one or no company, a movie company has at least 1 movie or none.
- Each production company belongs to at least 1 movie company, a production company may consist of many or one movie company.
- A criticism can be written by 1 critic, a critic can write many or no criticism.
- a show can be watched in 1 language, a language can be in 0 or many show
- A show can be watched in 1 country or many countries, a country has 1 show or many shows.
- a show can be watched in 0 or many digital platforms, a digital platform has 1 movie or many movies.

- Instance of Super-type Show can only be member of exactly one of movie or TV show.
- A show can only and only belong to 1 movie company, a movie company may have consisted of many or 1 show
- A show has zero or many criticisms. a criticism only and only belongs to 1 show.

Entity Relationship Model



Relational Model (Textual Representation)

Category(category_id: int, name: string, movie_count: int)

Language(language_id: int, name: string)

digital_platform(digital_platform_id: int, name: string)

Country(Country_id: int, name: string)

category_show(digital_platform_id: int, show_id: int)

language_show(language_id: int, show_id: int)

platform_show(digital_platform_id: int, show_id: int)

country_show(country_id: int, show_id: int)

actor(actor_id: int, name: string, surname: string, gender: char, birth_date: date)

actor_show(actor_id: int, show_id: int)

show(show_id: int, name: string, show_type: string, director_id: int, movie_company_id: int,critisim_id: int)

movie(show_id: int, movie_duration: int)

tv_series(show_id: int, season_count: int)

movie_company(movie_company_id: int, name: string, movie_count: int, production_company_id: string)

director(director_id: int, name: string, surname: string, gender:char, birth_date: date)

critic(critic_id: int,name: string, surname: string, gender: char, birth_date: date)

production_company(production_company_id: int, name: string, movie_company_count: int)

critisim(critism_id: int, critism_text: string, critic_id: int)

SQL Statements

```
CREATE TABLE "Production_Company" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"movie_company_count" INT DEFAULT 0,
	CONSTRAINT "Production_Company_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Movie_Company" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"movie_count" INT DEFAULT 0,
	"production_company_id" INT
);
ALTER TABLE  "Movie_Company"
ADD CONSTRAINT "Movie_Company_PK" PRIMARY KEY ("ID"),
ADD CONSTRAINT "Movie_Production_FK" FOREIGN KEY ("production_company_id") REFERENCES
"Production_Company" ("ID") ON DELETE CASCADE ON UPDATE CASCADE;

CREATE TABLE "Director" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"Surname" VARCHAR(50),
	"Gender" VARCHAR(1),
	"Birth_Date" DATE NOT NULL,
```

```
	"Movie_Count" INTEGER DEFAULT 0,
	CONSTRAINT "Director_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Criticism" (
	"ID" SERIAL,
	"Criticism_Text" VARCHAR(250),
	"Critic_ID" INT,
	"Show_ID" INT
);
ALTER TABLE "Criticism"
ADD CONSTRAINT "CriticismShowFK" FOREIGN KEY ("Show_ID") REFERENCES  "Show"("ID"),
ADD CONSTRAINT "Criticism_PK" PRIMARY KEY ("ID"),
ADD CONSTRAINT "CriticCriticismFK" FOREIGN KEY ("Critic_ID") REFERENCES  "Critic"("ID") ON DELETE
CASCADE ON UPDATE CASCADE;
CREATE TABLE "Critic" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"Surname" VARCHAR(50),
	"Gender" VARCHAR(1),
	"Birth_Date" DATE,
	"Criticism_Count" INTEGER DEFAULT 0,
	CONSTRAINT "Critic_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Actor" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"Surname" VARCHAR(50),
	"Gender" VARCHAR(1),
	"Birth_Date" DATE,
	CONSTRAINT "Actor_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Actor_Show" (
	"Actor_ID" INT,
	"Show_ID" INT
);
ALTER TABLE "Actor_Show"
ADD CONSTRAINT "Actor_Show_PK" PRIMARY KEY ("Actor_ID", "Show_ID"),
ADD CONSTRAINT "Actor_ID_FK" FOREIGN KEY ("Actor_ID") REFERENCES  "Actor" ("ID"),
ADD CONSTRAINT "Show_ID_FK" FOREIGN KEY ("Show_ID") REFERENCES  "Show" ("ID");

CREATE TABLE "Category" (
	"ID" SERIAL,
	"Name" VARCHAR(50) UNIQUE,
	CONSTRAINT "Category_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Language" (
	"ID" SERIAL,
	"Name" VARCHAR(50) UNIQUE,
	CONSTRAINT "Language_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Digital_Platform" (
	"ID" SERIAL,
	"Name" VARCHAR(50) UNIQUE,
	CONSTRAINT "Digital_Platform_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Country" (
	"ID" SERIAL,
```

```
	"Name" VARCHAR(50) UNIQUE,
	CONSTRAINT "Country_PK" PRIMARY KEY ("ID")
);
CREATE TABLE "Show" (
	"ID" SERIAL,
	"Name" VARCHAR(50),
	"Show_Type" VARCHAR(1),
	"Director_ID" INT,
	"Movie_Company_ID" INT,
	"CategoryID" INT,
	"CountryID" INT,
	"LanguageID" INT,
	"PlatformID" INT
);
ALTER TABLE "Show"
ADD CONSTRAINT "Show_PK" PRIMARY KEY ("ID"),
ADD CONSTRAINT "Category_ID_FK" FOREIGN KEY ("CategoryID") REFERENCES  "Category" ("ID"),
ADD CONSTRAINT "Country_ID_FK" FOREIGN KEY ("CountryID") REFERENCES  "Country" ("ID"),
ADD CONSTRAINT "Language_ID_FK" FOREIGN KEY ("LanguageID") REFERENCES  "Language" ("ID"),
ADD CONSTRAINT "Platform_ID_FK" FOREIGN KEY ("PlatformID") REFERENCES  "Digital_Platform" ("ID"),
ADD CONSTRAINT "Director_ID_FK" FOREIGN KEY ("Director_ID") REFERENCES  "Director" ("ID"),
ADD CONSTRAINT "Movie_Company_ID_FK" FOREIGN KEY ("Movie_Company_ID") REFERENCES
"Movie_Company" ("ID") ON DELETE CASCADE ON UPDATE CASCADE;

CREATE TABLE  "Movie" (
	"show_ID" INT,
	"Movie_Duration" INT
);
ALTER TABLE  "Movie"
ADD CONSTRAINT "Movie_PK" PRIMARY KEY ("show_ID"),
ADD CONSTRAINT "Movie_FK" FOREIGN KEY ("show_ID") REFERENCES  "Show" ("ID");

CREATE TABLE  "tv_series" (
	"show_ID" INT,
	"Season_Count" INT
);
ALTER TABLE  "tv_series"
ADD CONSTRAINT "Series_PK" PRIMARY KEY ("show_ID"),
ADD CONSTRAINT "Series_FK" FOREIGN KEY ("show_ID") REFERENCES  "Show" ("ID");
```

Functions

1-

```
CREATE OR REPLACE FUNCTION public.critic_criticism_showview()
 RETURNS TABLE(criticismid integer, criticismtext character varying, criticname character varying, criticsurname
character varying, criticgender character varying, criticbirthdate date)
 LANGUAGE plpgsql
AS $function$
BEGIN
  RETURN QUERY
  SELECT "criticism"."id",  "criticism"."criticism_text",  "critic"."name",  "critic"."surname",  "critic"."gender",
"critic"."birth_date" FROM  "criticism" JOIN  "critic" ON  "criticism"."critic_id" =  "critic"."id";
END;
$function$
```

2-

```
CREATE OR REPLACE FUNCTION public.criticismcountchange()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
   IF TG_OP = 'INSERT' THEN
      UPDATE "critic"
      SET "criticism_count" = "criticism_count" + 1
      WHERE "id" = NEW."critic_id";
   ELSIF TG_OP = 'DELETE' THEN
      UPDATE "critic"
      SET "criticism_count" = "criticism_count" - 1
      WHERE "id" = OLD."critic_id";
   END IF;
   RETURN NEW;
END;
$function$
```

3-

```
CREATE OR REPLACE FUNCTION public.moviecompany_productioncompany_showview()
 RETURNS TABLE(moviecompanyid integer, productioncompanyid integer, moviecompanyname character
varying, moviecount integer, productioncompanyname character varying)
 LANGUAGE plpgsql
AS $function$
BEGIN
  RETURN QUERY
  SELECT  "movie_company"."id", "movie_company"."production_company_id" AS productioncompanyid,
"movie_company"."name",  "movie_company"."movie_count" AS movieCount,
"production_company"."name" AS productionName FROM  "movie_company" JOIN  "production_company"
on  "movie_company"."production_company_id" =  "production_company"."id";
END;
$function$
```

4-

```
CREATE OR REPLACE FUNCTION public.moviecompanycountchange()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
DECLARE
   num integer;
BEGIN
   IF TG_OP = 'INSERT' THEN
      UPDATE "production_company"
      SET "movie_company_count" = "movie_company_count" + 1
      WHERE "id" = NEW."production_company_id";
   ELSIF TG_OP = 'DELETE' THEN
      UPDATE "production_company"
      SET "movie_company_count" = "movie_company_count" - 1
      WHERE "id" = OLD."production_company_id";
   END IF;
   RETURN NEW;
END;
$function$
```

5-

```
CREATE OR REPLACE FUNCTION public.moviecountchange()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
   IF TG_OP = 'INSERT' THEN
      UPDATE "movie_company"
      SET "movie_count" = "movie_count" + 1
      FROM "show"
      WHERE "movie_company"."id" = "show"."movie_company_id";
   ELSIF TG_OP = 'DELETE' THEN
      UPDATE "movie_company"
      SET "movie_count" = "movie_count" - 1
      FROM "show"
      WHERE "movie_company"."id" = "show"."movie_company_id";
   END IF;
   RETURN NEW;
END;
$function$
```

6-

```
CREATE OR REPLACE FUNCTION public.moviecountchangeondirector()
 RETURNS trigger
 LANGUAGE plpgsql
AS $function$
BEGIN
   IF TG_OP = 'INSERT' THEN
      UPDATE "director"
      SET "movie_count" = "movie_count" + 1
      WHERE "id" = NEW."director_id";
   ELSIF TG_OP = 'DELETE' THEN
      UPDATE "director"
      SET "movie_count" = "movie_count" - 1
      WHERE "id" = OLD."director_id";
   END IF;
   RETURN NEW;
END;
$function$
```

7-

```
CREATE OR REPLACE FUNCTION public.showsbytype(stype character varying)
 RETURNS TABLE(id integer, moviename character varying, show_type character varying, director_id integer,
criticism_id integer, movie_company_id integer)
 LANGUAGE plpgsql
AS $function$
BEGIN
  RETURN QUERY
  SELECT * FROM "show" WHERE "show"."show_Type" = sType;
END;
$function$
```

8-

```
CREATE OR REPLACE FUNCTION public.totalmoviecountonproductioncompany(pid integer)
 RETURNS integer
 LANGUAGE plpgsql
AS $function$
DECLARE
   num INTEGER;
BEGIN
  SELECT SUM(movie_count) INTO num FROM "movie_company" WHERE "production_company_id" = pID;
  return num;
END;
$function$
```

Triggers

1-

```
CREATE TRIGGER "movieCompanyCountChangeTrig" AFTER INSERT OR DELETE ON public.movie_company FOR EACH ROW EXECUTE FUNCTION moviecompanycountchange();
```

2-

```
CREATE TRIGGER "CriticismCountChangeTrig" AFTER INSERT OR DELETE ON public.criticism FOR EACH ROW EXECUTE FUNCTION criticismcountchange();
```

3-

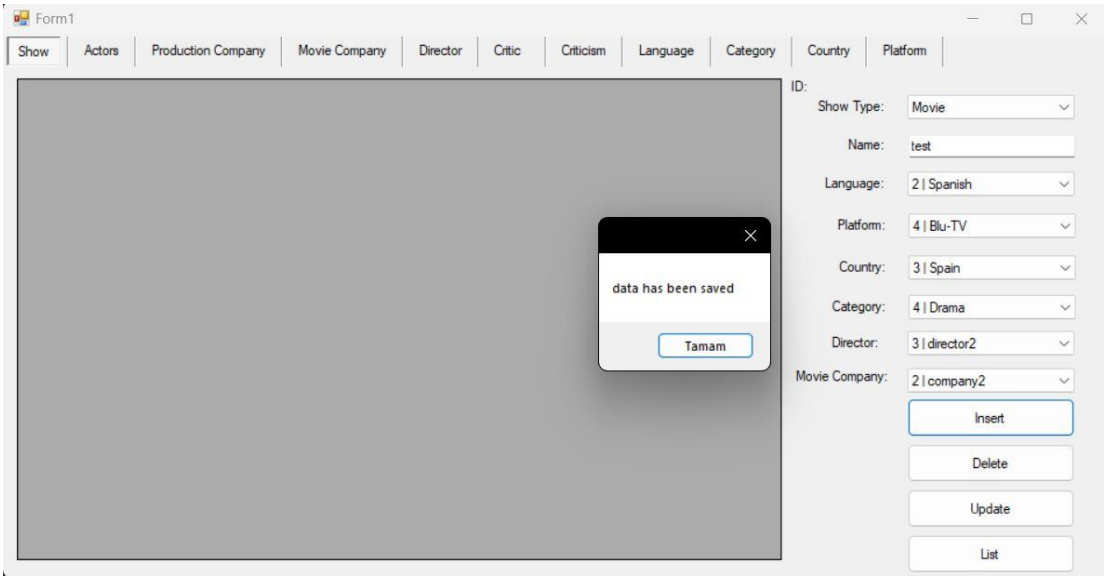
```
CREATE TRIGGER "MovieCountChangeOnDirectorTrig" AFTER INSERT OR DELETE ON public.show FOR EACH ROW EXECUTE FUNCTION moviecountchangeondirector();
```

4-

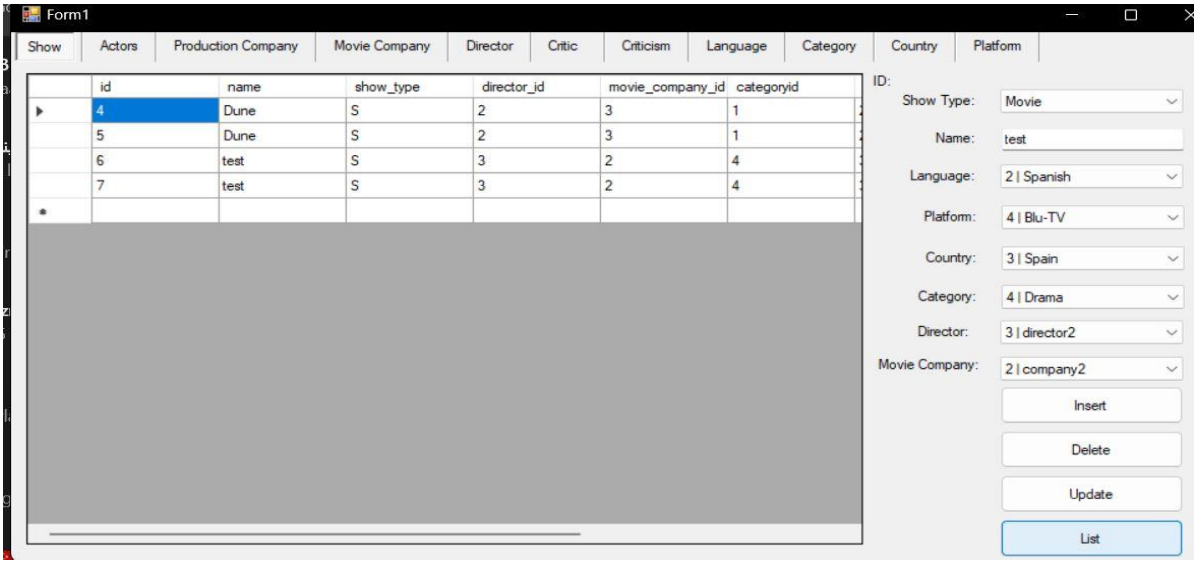
```
CREATE TRIGGER "MovieCountChangeTrig" AFTER INSERT OR DELETE ON public.show FOR EACH ROW EXECUTE FUNCTION moviecountchange();
```

Operations

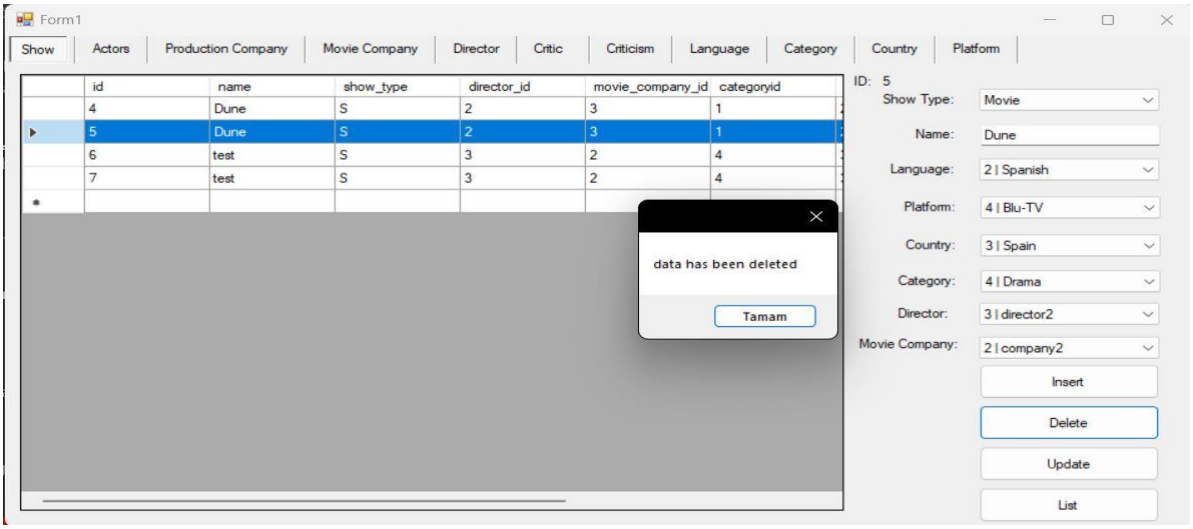
Insert



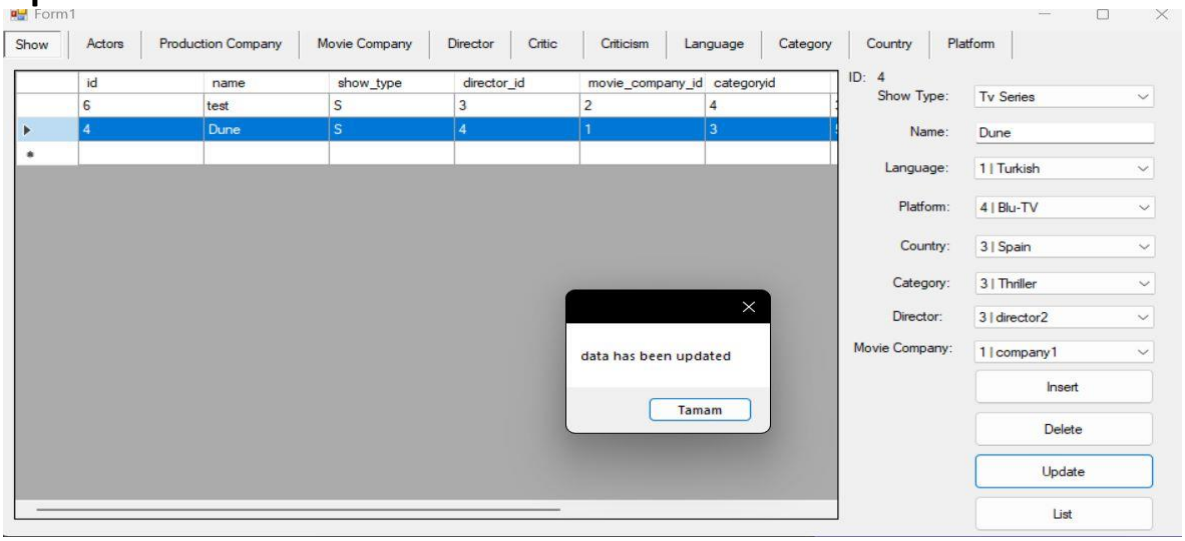
List



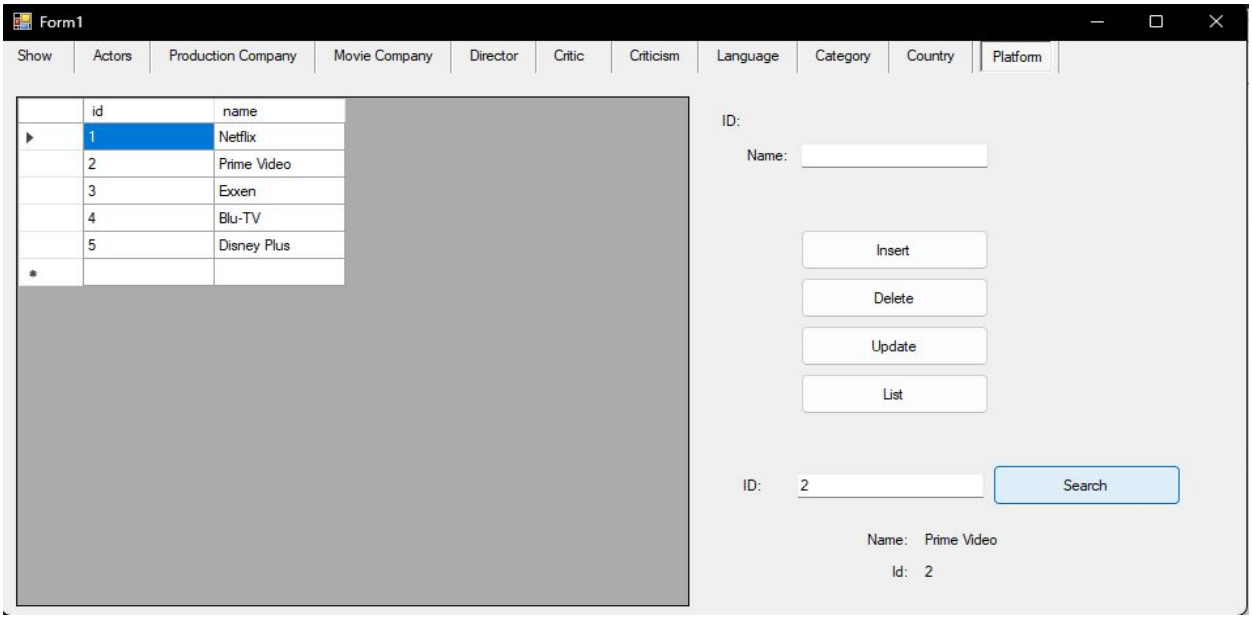
Delete



Update



Search


Form1
Show
Actors
Production Company
Movie Company
Director
Critic
Criticism
Language
Category
Country
Platform
id
name
1
Netflix
2
Prime Video
3
Exxen
4
Blu-TV
5
Disney Plus
ID:
Name:
Insert
Delete
Update
List
ID:
2
Search
Name: Prime Video
Id: 2