# Movies WatchList

## -Proiect Baze de Date-

Student: Buzatu Rares Tudor

Grupa: 332AA

**Cuprins:**

# 1.Descrierea cerintelor

Proiectul implica crearea unei baze de date care trebuie asociata cu o interfata grafica pentru a putea fi manipulata de catre un utilizator si de catre un administrator. Conditiile bazei de date impun ca aceasta sa contina:

1. Minim 5 tabele (fara cele de legatura)
2. Minim 2 relatii 1:N
3. Minim 1 relatie N:N
4. Insert, Update, Delete la minim 2 tabele
5. Minim 6 interogari simple
6. Minim 4 interogari complexe

Aplicatia ce contine baza de date trebuie insotita de un document explicativ al acesteia.

Tema proiectului meu a fost stabilita ca fiind “Movies WatchList”, adica o lista de filme preferate de un utilizator, filme alese din baza mea de date.

Am ales aceasta tema, deoarece imi doream sa fac aceasta aplicatie inca de anul trecut, iar aceasta oportunitatea mi s-a parut perfecta. Aplicatia a fost proiectata nu in scopul folosirii ei pe o scara mare de personae, ci a fost proiectata din nevoia personala de a avea un WatchList personalizat. Acest lucru explica inca de pe acum faptul ca aplicatia va avea un singur administrator.

# 2.Etapa de proiectare

Pentru inceput m-am gandit la ce informatii vreau sa contina aceasta baza de date, apoi am incercat sa constuiesc tabelele pentru a respecta conceptele de baza ale bazelor de date si cerintele impuse. In urma proiectarii am ajuns la urmatoarele 7 tabele:

1. Film (FilmID, Titlu, Durata, Limba, Tara, Theme_songID, Gen, Detalii, RegizorID, An)
2. Actor (ActorID, NumeA, PrenumeA, Data_nasteriiA, SexA, TaraA, Nr_premii, Nr_nominalizari, Nr_filmeJucate)
3. Casting (ActorID, FilmID, Rol)
4. Regizor (RegizorID, Nume, Prenume, Data_nasterii, Sex, TaraR, Nr_filmeProduse)
5. Theme_song (ThemeSongID, Nume, Autor, Durata, Gen, An_aparitie)
6. Utilizator (UtilizatorID, Username, Parola, Email)
7. Utilizator_film (UtilizatorID, FilmID, Rating)

# 3. Descrierea tabelelor si a relatiilor dintre ele

- Tabela Film contine informatii despre filmele stocate in baza de date
- Tabela Actor contine informatii despre actorii stocati in baza de date
- Tabela Regizor contine informatii despre regizorii stocati in baza de date
- Tabela Theme_song contine informatii despre theme_songurile stocate in baza de date
- Tabela Utilizator contine informatii despre utilizatorii stocati in baza de date
- Tabela Casting face legatura intre tabela Film si cea Actor, ea continand cheile primare ale celor 2 tabele, prin urmare ea arata ce actor joaca in ce film si ce rol are respectviul actor. Fiind o relatie N:N aveam nevoie de aceasta tabela de legatura pentru a putea face posibila asocierea.
- Tabela Utilizatori_film face legatura intre tabela Film si cea Utilizator, ea continand cheile primare ale celor 2 tabele, prin urmare ea arata filmele preferate ale fiecarui utilizator. Fiind o relatie N:N aveam nevoie de aceasta tabela de legatura pentru a putea face posibila asocierea.

Fiecare film din baza de date contine un regizor si un theme_song, rezultand doua relatii 1:N.

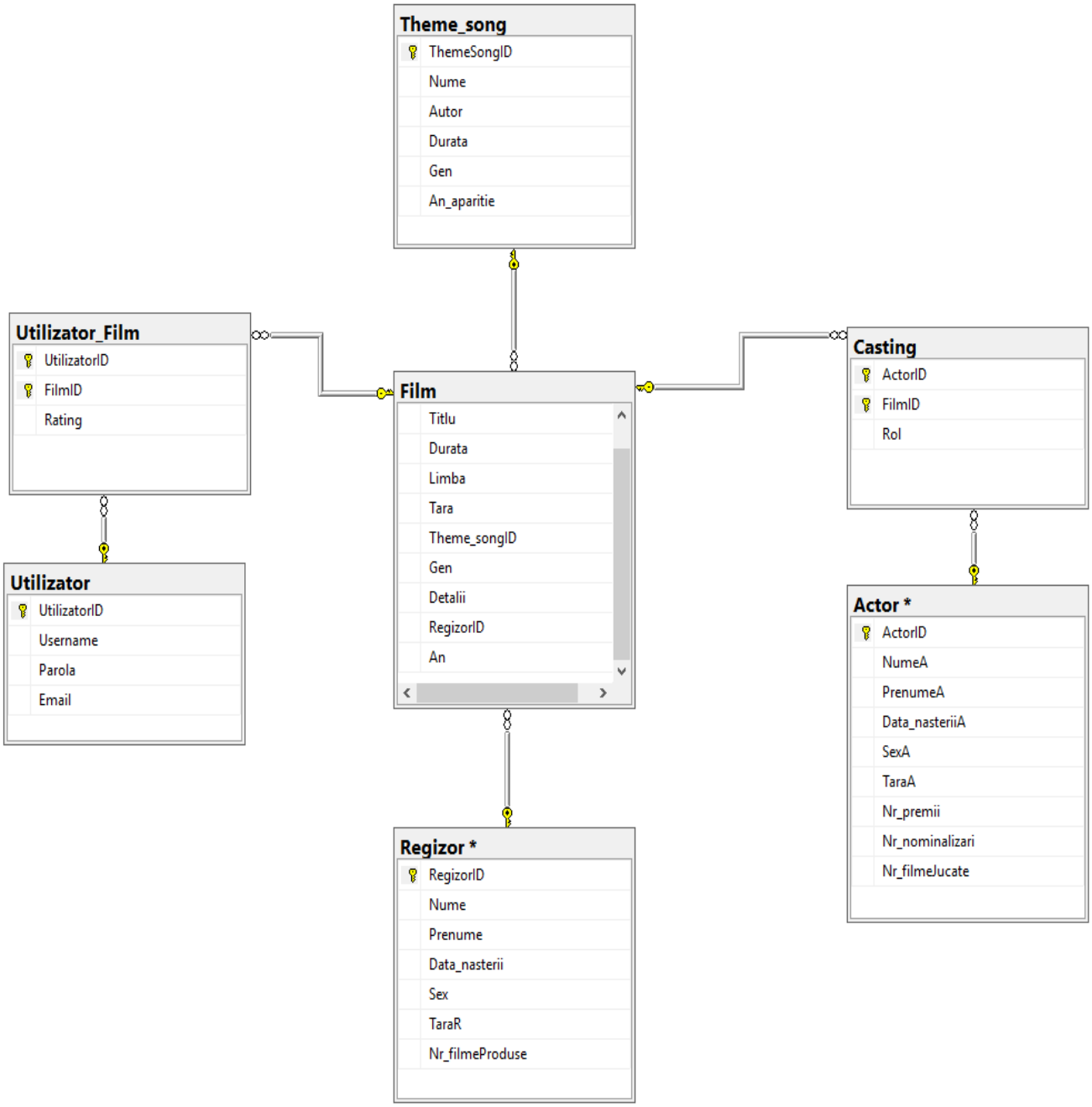


*Figure 1: Diagrama tabele*

Asadar, dupa cum se vede si in diagrama, exista 4 relatii: doua N:N si doua 1:N, avand 5 tabele excluzandu-le pe cele de legatura.

# 4. Constrangeri impuse

Pe langa relatiile de primary key si foreign key necesare pentru legarea tabelelor intre ele, am adaugat urmatoarele constrangeri:

- Sexul actorilor si al regizorilor poate fi doar F sau M;
- Numele filmelor si al theme_song-urilor, numele+prenumele actorilor si regizorilor si username-ul utilizatorilor sunt unice;
- Multe dintre atributele de baza ale entitatilor nu pot fi "*NULL*"

# 5. Functionalitatea aplicatiei

Aplicatia a fost proiectata pentru a lasa utilizatorul sa isi aleaga o multime de filme din baza de date si a le adauga intr-o lista de filme preferate de el sau vizionate de el, astfel incat sa poata tina minte ce filme a vizionat sau ce filme vrea sa vizioneze in continuare.

Administratorul poate adauga, modifica si sterge orice entitate din baza de date care tine de Watchlist (nu poate modifica datele utilizatorilor).

Utilizatorii isi creaza un cont cu username, email si parola, iar datele lor sunt tinute minte in baza de date

***Parola este salvata in baza de date cu ajutorul unei functii de hash(ex: m5()), astfel parola are un grad de securitate sporit

Utilizatorul are acces doar la tabela cu filme.

Atat utilizatorul cat si administratorul pot filtra si ordona datele din tabel cu ajutorul unor filter de baza sau cu ajutorul unor filter special.

In urma aplicarii filtrelor, query cu care este interogata baza de date este dezvoltat procedural, astfel ca exista o infinitate de query-uri posibile cu care se poate interoga baza de date.

Am ales aceasta alcatuire procedurala pentru a lasa la voia utilizatorului / administratorului criteriile de filtrare oferindu-I libertate deplina in vizualizarea datelor. Din cauza acestei metode de alcatuire, query-ul format necesita mai multe linii de cod, asa ca nu v-a putea sa fie vizualizata linia de cod ce il contine, dar v-a putea fi vizualizat chiar in aplicatie.

Folosindu-ma de salvarea datelor introduse de catre utilizator in variabile, am reusit sa evit hardcodingul.

Modurile de stergere, modificare, si adaugare sunt intuitive.

Atat utilizatorul, cat si administratorul, sunt atentionati in cazul in care introduce in baza de date o entitate care are acelasi nume ca una deja existent sau daca uita sa introduca un atribut esential pentru o entitate.

# 6. Exemple

## 6.1. Selecturi

### 6.1.1. Selecturi usoare:

Dupa cum am precizat si mai sus, cu ajutorul butoanelor de filtru pot avea foarte multe selecturi usoare. Mai jos sunt exemplificate cateva cat mai diferite unele fata de altele

1. Selecteaza toate filmele, numele+prenumele regizorului si numele theme song-ului care dureaza mai mult de 120 de minute, joaca in ele actorul cu numele “DiCaprio” si regizorul lor este barbat. Ordonati rezultatele crescator dupa nume.

```
SELECT DISTINCT film.*,CONCAT(regizor.prenume,' ',regizor.nume) as
Nume_Regizor, theme_song.Nume from regizor,actor,casting, film LEFT JOIN
    theme_song ON film.Theme_songID = theme_song.ThemeSongID WHERE
film.RegizorID=regizor.RegizorID AND film.Durata > '120' AND film.FilmID
 = casting.FilmID AND casting.ActorID = actor.ActorID AND actor.NumeA =
 'DiCaprio' AND film.RegizorID = regizor.RegizorID AND regizor.Sex = 'M'
                          ORDER BY Titlu ASC
```

2. Selecteaza toti actorii care joaca in cel putin un film Drama care dureaza mai putin de 200 de minute, sunt din SUA si au mai mult de 80 de premii. Ordonati descrescator dupa varsta rezultatele

```
    SELECT DISTINCT actor.* from actor ,film,casting WHERE
actor.ActorID = casting.ActorID AND film.FilmID = casting.FilmID
AND film.Gen = 'Drama' AND film.Durata < '200' AND actor.TaraA =
      'USA' AND actor.Nr_premii > '80' ORDER BY '$data'-
                  year(Data_nasteriiA) DESC
```

3. Selecteaza toti regizorii al caror prenume este alfabetic inaintea literei “r”, au varsta mai mare de 50 de ani si au regizat filme in limba engleza cu titlul alfabetic  mai mare de litera “b”.

Ordonati crescator rezultatul dupa varsta regizorilor.

```
SELECT DISTINCT regizor.* from regizor ,film WHERE
film.RegizorID = regizor.RegizorID AND film.Titlu > 'b'
AND film.Limba = 'Engleza' AND regizor.Prenume < 'r' AND
2019-year(regizor.Data_nasterii) > '50' ORDER BY
'$data'-year(Data_nasterii) ASC
```

4. Actorii care nu joaca in niciun film

```
SELECT a2.* from actor as a2 WHERE a2.ActorID NOT IN (SELECT DISTINCT
a.ActorID from actor as a, casting WHERE a.ActorID = casting.ActorID )
```

5. Selecteaza toate filmele din WatchList-ul utilizatorului, numele+prenumele regizorului si numele theme song-ului cu username-ul Maria care au genul Drama si durata mai mica de 190 de minute, joaca in ele cel putin un actor care a castigat mai mult de 40 de premii si este barbat si sunt produse de un regizor mai batran de 30 de ani care a produs mai mult de 15 filme. Ordonati rezultatul descrescator dupa anul aparitiei filmelor.

```
SELECT DISTINCT utilizator_film.Rating, film.*,CONCAT(regizor.prenume,' ',regizor.nume)
        as Nume_Regizor, theme_song.Nume from utilizator, utilizator_film,
      regizor,actor,casting, film LEFT JOIN theme_song ON film.Theme_songID =
theme_song.ThemeSongID WHERE film.RegizorID=regizor.RegizorID AND utilizator.Username =
  'Maria' AND utilizator.UtilizatorID = utilizator_film.UtilizatorID AND film.FilmID =
utilizator_film.FilmID AND film.Gen = 'Drama' AND film.Durata < '190' AND film.FilmID =
     casting.FilmID AND casting.ActorID = actor.ActorID AND actor.SexA = 'M' AND
       actor.Nr_premii > '20' AND film.RegizorID = regizor.RegizorID AND 2019-
 year(regizor.Data_nasterii) > '30' AND regizor.Nr_filmeProduse > '15' ORDER BY An DESC
```

6. Selecteaza toate theme song-urile care au durata egala cu 180 de secunde si au genul “Romance” si sunt interpretate in macar un film din SUA. Ordonati crescator rezultatele dupa autorul theme song-ului.

```
SELECT DISTINCT theme_song.* from theme_song ,film WHERE
film.Theme_songID = theme_song.ThemeSongID AND film.Tara = 'USA'
AND theme_song.Durata = '180' AND theme_song.Gen = 'Romance' ORDER
BY Autor ASC
```

## 6.1.2.Selecturi complicate:

1.Selectati numele+prenumele regizorului filmelor si numele theme song-ului filmelor si filmele in care joaca actorul cu numele “DiCaprio” si au durata mai mare decat media tuturor filmelor care au genul Drama.

```
SELECT DISTINCT f.*,CONCAT(regizor.prenume,' ',regizor.nume) as
Nume_Regizor, theme_song.Nume from regizor,actor,casting,film as f LEFT
JOIN theme_song ON f.Theme_songID = theme_song.ThemeSongID WHERE f.FilmID
= casting.FilmID AND casting.ActorID = actor.ActorID AND actor.NumeA =
'DiCaprio' AND f.RegizorID=regizor.RegizorID HAVING f.Durata > (SELECT
avg(f2.Durata) from film as f2 WHERE f2.Gen = 'Drama')
```

2. Selectati numele+prenumele regizorului filmelor si numele theme song-ului filmelor si filmele care au aparut in anii in care au aparut mai multe filme decat media filmelor aparute in toti anii in care au aparut filme

```
SELECT CONCAT(regizor.prenume,' ',regizor.nume) as Nume_Regizor, film.*,theme_song.Nume from
regizor, film LEFT JOIN theme_song ON film.Theme_songID = theme_song.ThemeSongID, (select
count(*) nrfilme, an from film group by an) nrf, (select count(*)/b.ani as medie from film,
(SELECT MAX(film.An)-MIN(film.An) ani from film) as b ) m where nrfilme >= m.medie and nrf.an =
film.an and regizor.RegizorID=film.RegizorID
```

3.Filmele care au ratingul peste ratingul mediu

```
SELECT CONCAT(regizor.prenume,' ',regizor.nume) as Nume_Regizor,
f.*,theme_song.Nume,b.medie from regizor ,(SELECT FilmID, avg(rating) as medie from
utilizator_film GROUP BY utilizator_film.FilmID) as b,(SELECT avg(rating) as medietotala
from utilizator_film) as a,film as f LEFT JOIN theme_song ON f.Theme_songID =
theme_song.ThemeSongID where f.filmID = b.filmID and b.medie>a.medietotala and
f.RegizorID=regizor.RegizorID
```

4.Afisati numarul actorilor care joaca in minim 2 categorii de filme

```
SELECT count(*) as Nr from actor, (SELECT distinct a.ActorID,count(f.Gen)
nrgenuri from actor a, casting c, film f where a.ActorID = c.ActorID and
c.FilmID = f.FilmID group by a.ActorID) s where actor.ActorID = s.ActorID
and nrgenuri >= '2'
```

5.Afisati actorii care nu joaca in niciun film Drama

```
SELECT a2.* from actor as a2 WHERE a2.ActorID NOT IN (SELECT DISTINCT
a.ActorID from actor as a, casting,film WHERE a.ActorID = casting.ActorID
        AND casting.FilmID = film.FilmID AND film.Gen = 'Drama')
```

## 6.2. Insert

Adauga film “Pirates of the Caribbean” cu durata 100 de min, limba engleza, tara USA, theme_song-ul “He’s a pirate”, genul “Actiune”, Detaliile “Aventura, Drama”, si regizorul Sam Mendes din anul 2000.

Adauga actori

```
INSERT INTO film
(Titlu,Durata,Limba,Tara,Theme_songID,Gen,Detalii,RegizorID,AN)
VALUES ('Pirates of the
Caribbean','100','Engleza','USA' ,
(SELECT ThemeSongID from
theme_song WHERE
theme_song.Nume = 'He`s a Pirate'),
'Actiune','Aventura, Drama',
(SELECT RegizorID from regizor
WHERE
CONCAT(regizor.prenume,'
',regizor.nume) = 'Sam
Mendes'),'2000')
```

## 6.3 Update

Modifica titlul filmului selectat in “Alt Titanic”

```
UPDATE film SET Titlu= 'Alt Titanic' WHERE film.FilmID = '1'
```

## 6.4 Delete

Sterge filmul selectat din baza de date

DELETE from film where film.FilmID = '$value'

# 7. Concluzii

Dupa terminarea proiectului, am constatat indeplinite toate functionalitatile propuse pentru aplicatia de WatchList. Baza de Date respecta structura si conditiile impuse.

# 8. Bibliografie

Link curs PHP: http://www.marplo.net/php-mysql/
Exercitii selecturi: www.w3shools.com
Exercitii Join: sqlzoo.net