Audience(username: string, password: string, name: string, surname: string)

CREATE TABLE Audience(

username VARCHAR(255),

password VARCHAR(255),

name VARCHAR(255),

surname VARCHAR(255),

PRIMARY KEY (username)

)

Director(username: string, password: string, name: string, surname: string, nation: string)

CREATE TABLE Director(

username VARCHAR(255),

password VARCHAR(255),

name VARCHAR(255),

surname VARCHAR(255),

nation VARCHAR(255) NOT NULL,

PRIMARY KEY (username)

)

Rating\_Platform(id: integer, name: string)

CREATE TABLE Rating\_Platform(

id INT,

name VARCHAR(255) UNIQUE,

PRIMARY KEY (id)

)

Movie(id: integer, name: string, duration: integer, overall\_rating: real, director\_name: string)

CREATE TABLE Movie(

id INT,

name VARCHAR(255),

duration INT,

overall\_rating REAL,

director\_name VARCHAR(255) NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (director\_name) REFERENCES Director

)

What should happen on delete?

Movie\_Session(id: integer, movie\_id: integer, theater\_id: integer, time\_slot: integer, date: string)

CREATE TABLE Movie\_Session(

id INT,

movie\_id INT,,

theater\_id INT,

time\_slot INT,

date DATE,

PRIMARY KEY (id),

FOREIGN KEY (movie\_id) REFERENCES Movie,

FOREIGN KEY (theater\_id) REFERENCES Theater

)

Theater(id: integer, name: string, capacity: integer, district: string)

CREATE TABLE Theater

(

id INT,

name VARCHAR(255),

capacity INT,

district VARCHAR(255),

PRIMARY KEY (id)

)

Genre(id: integer, name: string)

CREATE TABLE Genre(

id INT,

name VARCHAR(255)

PRIMARY KEY (id)

)

Movie\_Director\_On\_Platform(movie\_id: integer, director\_name: string, platform\_id: integer)

CREATE TABLE Movie\_Director\_On\_Platform(

movie\_id INT,

director\_name INT,,

platform\_id INT,

PRIMARY KEY (movie\_id, director\_name),

FOREIGN KEY (movie\_id) REFERENCES Movie

ON DELETE CASCADE

ON UPDATE CASCADE

FOREIGN KEY (director\_name) REFERENCES Director

ON DELETE CASCADE

ON UPDATE CASCADE

)

Ticket(username: string, session\_id: integer)

CREATE TABLE Ticket(

username VARCHAR(255),

session\_id INT,,

PRIMARY KEY (username, session\_id),

FOREIGN KEY (username) REFERENCES Audience,

FOREIGN KEY (session\_id) REFERENCES Movie\_Session

)

Platform\_Subscription(username: string, platform\_id: integer)

CREATE TABLE Platform\_Subscription(

username VARCHAR(255),

platform\_id INT,,

PRIMARY KEY (username, platform\_id),

FOREIGN KEY (username) REFERENCES Audience,

FOREIGN KEY (platform\_id) REFERENCES Rating\_Platform

)

Movie\_Ratings(username: string, rating: real, movie\_id: integer)

CREATE TABLE Movie\_Ratings(

username VARCHAR(255),

rating REAL,

movie\_id INT,,

PRIMARY KEY (username, movie\_id),

FOREIGN KEY (username) REFERENCES Audience,

FOREIGN KEY (movie\_id) REFERENCES Movie

)

Movie\_Genre(movie\_id: integer, genre\_id: integer)

CREATE TABLE Movie\_Genre(

movie\_id INT,

genre\_id INT,

PRIMARY KEY (movie\_id, genre\_id),

FOREIGN KEY (movie\_id) REFERENCES Movie,

FOREIGN KEY (genre\_id) REFERENCES Genre

)

Movie\_Predecessor(predecessor\_id: integer, successor\_id: integer)

CREATE TABLE Movie\_Predecessor(

predecessor\_id INT,

successor\_id INT,

PRIMARY KEY (predecessor\_id, successor\_id),

FOREIGN KEY (predecessor\_id) REFERENCES Movie,

FOREIGN KEY (successor\_id) REFERENCES Movie

)

Database\_Managers(username: string, password: string)

CREATE TABLE Audience(

username VARCHAR(255),

password VARCHAR(255),

PRIMARY KEY (username)

)

Comments About The ER Diagram:

Bahadirin aggregation zort

Add overall rating,

Add the attributes of the movie session

General Questions:

Should we add NOT NULL everywhere,

We may specify each table id accordingly instead of simply ‘id’

Constraints that we cannot cover in ER:

* Each director can have at most one platform id.
* No two movie sessions can overlap in terms of theater and the time it’s

screened

* There are four time slots for each day.
* If a movie has any predecessor movies, all predecessor movies need to

be watched in order to watch that movie.

* A user can rate a movie

– if they are already subscribed to the platform that the movie can be rated.

AND

– if they have bought a ticket to the movie.

* There can be at most 4 database managers registered to the system.

Constraints we cannot cover in Schemas:

* The ones in ER
* Every movie needs to have at least one genre.