**1. Konceptuālais ER modelis**

**Mērķis:** Izveidot augsta līmeņa pārskatu par sistēmas būtiskākajām entītijām un attiecībām.

**Entītijas un attiecības:**

* **Film** (Filma) — entītija, kas pārstāv filmu.
* **Theater** (Kinozāle) — entītija, kas pārstāv kinozāli.
* **Show** (Seanss) — entītija, kas pārstāv konkrētu filmas seansu.
* **Customer** (Klients) — entītija, kas pārstāv klientu.
* **Reservation** (Rezervācija) — entītija, kas pārstāv klienta rezervāciju uz konkrētu seansu.

**Attiecības:**

* **One-to-Many** starp **Film** un **Show**: Vienai filmai var būt vairāki seansi.
* **One-to-Many** starp **Theater** un **Show**: Vienā kinozālē var notikt vairāki seansi.
* **Many-to-Many** starp **Customer** un **Show**: Katrs klients var rezervēt vietas uz vairākiem seansiem, un katram seansam var būt vairāki klienti.

**2. Loģiskais ER modelis**

**Mērķis:** Definēt entītijas ar primārajām atslēgām, ārējām atslēgām un attiecībām, saglabājot augstu abstrakcijas līmeni.

**Entītijas un atribūti:**

* **Film**: FilmID (PK), Title, Genre, Duration.
* **Theater**: TheaterID (PK), Name, Location.
* **Show**: ShowID (PK), TheaterID (FK), FilmID (FK), ShowTime.
* **Customer**: CustomerID (PK), Name, Email.
* **Reservation**: ReservationID (PK), ShowID (FK), CustomerID (FK), Seats, ReservationTime.

**Attiecības:**

* **One-to-Many** starp **Film** un **Show**: FilmID ir ārējā atslēga Show tabulā.
* **One-to-Many** starp **Theater** un **Show**: TheaterID ir ārējā atslēga Show tabulā.
* **Many-to-Many** starp **Customer** un **Show**: tiek realizēta ar Reservation tabulu, kas satur CustomerID un ShowID ārējās atslēgas.

**3. Fiziskais ER modelis**

**Mērķis:** Norādīt detalizētas datu kolonnas, datu tipus un attiecības, kas jāizmanto, lai SQL kodā izveidotu tabulas.

**Tabulu struktūra:**

1. **Film**
   * **FilmID**: INTEGER (primārā atslēga)
   * **Title**: TEXT
   * **Genre**: TEXT
   * **Duration**: INTEGER
2. **Theater**
   * **TheaterID**: INTEGER (primārā atslēga)
   * **Name**: TEXT
   * **Location**: TEXT
3. **Show**
   * **ShowID**: INTEGER (primārā atslēga)
   * **TheaterID**: INTEGER (ārējā atslēga uz Theater)
   * **FilmID**: INTEGER (ārējā atslēga uz Film)
   * **ShowTime**: DATETIME
4. **Customer**
   * **CustomerID**: INTEGER (primārā atslēga)
   * **Name**: TEXT
   * **Email**: TEXT (unikāls ierobežojums)
5. **Reservation**
   * **ReservationID**: INTEGER (primārā atslēga)
   * **ShowID**: INTEGER (ārējā atslēga uz Show)
   * **CustomerID**: INTEGER (ārējā atslēga uz Customer)
   * **Seats**: INTEGER
   * **ReservationTime**: DATETIME

**Attiecību vizualizācija:**

* **Film** ↔ **Show**: One-to-Many, savienots caur FilmID.
* **Theater** ↔ **Show**: One-to-Many, savienots caur TheaterID.
* **Customer** ↔ **Reservation** ↔ **Show**: Many-to-Many, savienots caur Reservation tabulu ar CustomerID un ShowID ārējām atslēgām.

-- Tabula Film (Filma)

CREATE TABLE Film (

FilmID INTEGER PRIMARY KEY,

Title TEXT NOT NULL,

Genre TEXT,

Duration INTEGER -- ilgums minūtēs

);

-- Tabula Theater (Kinozāle)

CREATE TABLE Theater (

TheaterID INTEGER PRIMARY KEY,

Name TEXT NOT NULL,

Location TEXT

);

-- Tabula Show (Seanss)

CREATE TABLE Show (

ShowID INTEGER PRIMARY KEY,

TheaterID INTEGER,

FilmID INTEGER,

ShowTime DATETIME,

FOREIGN KEY (TheaterID) REFERENCES Theater(TheaterID),

FOREIGN KEY (FilmID) REFERENCES Film(FilmID)

);

-- Tabula Customer (Klients)

CREATE TABLE Customer (

CustomerID INTEGER PRIMARY KEY,

Name TEXT NOT NULL,

Email TEXT UNIQUE

);

-- Tabula Reservation (Rezervācija)

CREATE TABLE Reservation (

ReservationID INTEGER PRIMARY KEY,

ShowID INTEGER,

CustomerID INTEGER,

Seats INTEGER, -- vietu skaits

ReservationTime DATETIME,

FOREIGN KEY (ShowID) REFERENCES Show(ShowID),

FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)

);