

Portfólió — Videotéka kölcsönzési rendszer

Tartalom:

Rövid összefoglaló:.....	2
Követelmények (átfogóan):	2
Funkcionális igények:	2
Adatmodell — rövid áttekintés:.....	2
Javasolt SQL létrehozó parancsok (MySQL kompatibilis):	2
Példák — tipikus lekérdezések és működés:	3
Működése:.....	4
Önreflexió:	4

Szerző: Fekete Ádám

Projekt címe: Videotéka kölcsönzési adatbázis — tervezés és megvalósítás

Dátum: 2025. május 23.

Rövid összefoglaló:

A projekt célja egy egyszerű, megbízható relációs adatbázis megtervezése és megvalósítása egy kisebb videotéka kölcsönzési folyamataihoz. A rendszer tárolja a filmeket, ügyfeleket, dolgozókat és a kölcsönzési tranzakciókat; cél a könnyű lekérdezhetőség, késések követése és a skálázhatóság biztosítása.

Követelmények (átfogóan):

- Ügyfelek: név, kapcsolat (e-mail, telefon), lakcím, regisztráció dátuma
- Filmek: cím, műfaj/kategória, megjelenési év, kölcsönzési díj, állapot (elérhető / foglalt / sérült)
- Dolgozók: név, beosztás, munkába lépés dátuma
- Kölcsönzések: ügyfél, film, dolgozó (kiadta), kölcsönzés dátuma, határidő, visszahozás dátuma, késedelmi díj (számított)

Funkcionális igények:

- Új kölcsönzés és visszavétel rögzítése
- Elérhető filmek és kölcsönzési statisztikák lekérdezése
- Késedelmes tételek és késedelmi díjak kiszámítása
- Adatkonzisztencia idegen kulcsokkal és alapvető integritási szabályokkal

Adatmodell — rövid áttekintés:

A rendszer négy fő táblát használ: ügyfelek, filmek, dolgozók, kölcsönzések. A táblák egymáshoz kapcsolódva biztosítják a normalizált adatstruktúrát. Az idegen kulcsok segítenek a tranzakciók és entitások közti kapcsolatok fenntartásában.

Javaolt SQL létrehozó parancsok (MySQL kompatibilis):

```
CREATE DATABASE IF NOT EXISTS videoteka;  
USE videoteka;
```

```
CREATE TABLE ugyfelek (  
    ugyfel_id INT AUTO_INCREMENT PRIMARY KEY,  
    nev VARCHAR(120) NOT NULL,  
    email VARCHAR(150),  
    telefonszam VARCHAR(30),  
    cim VARCHAR(250),  
    regisztracio_datum DATE DEFAULT CURDATE()  
);
```

```
CREATE TABLE filmek (  
    film_id INT AUTO_INCREMENT PRIMARY KEY,  
    cim VARCHAR(150) NOT NULL,  
    mufaj VARCHAR(60),  
    megjelenesi_ev YEAR,  
    dij DECIMAL(6,2) DEFAULT 0.00,  
    statusz ENUM('elerheto','kicolcsonozve','serult') DEFAULT 'elerheto'  
);
```

```
CREATE TABLE dolgozok (  
    dolgozo_id INT AUTO_INCREMENT PRIMARY KEY,  
    nev VARCHAR(120) NOT NULL,  
    beosztas VARCHAR(80),  
    munkaba_lepes DATE  
);
```

```
CREATE TABLE kolcsonzesek (  
    kolcsonzes_id INT AUTO_INCREMENT PRIMARY KEY,  
    ugyfel_id INT NOT NULL,  
    film_id INT NOT NULL,  
    dolgozo_id INT NOT NULL,  
    kolcsonzes_datum DATE NOT NULL,  
    visszahozas_hatarido DATE NOT NULL,  
    visszahozva DATE DEFAULT NULL,  
    kesedelmi_dij DECIMAL(7,2) DEFAULT 0.00,  
    FOREIGN KEY (ugyfel_id) REFERENCES ugyfelek(ugyfel_id),  
    FOREIGN KEY (film_id) REFERENCES filmek(film_id),  
    FOREIGN KEY (dolgozo_id) REFERENCES dolgozok(dolgozo_id)  
);
```

(Megjegyzés: a kesedelmi_dij kiszámítható alkalmazásszinten triggerrel vagy lekérdezés során.)

Példák — tipikus lekérdezések és működés:

Elérhető filmek listázása:

```
SELECT * FROM filmek WHERE statusz = 'elerheto';
```

Lejárt, vissza nem hozott kölcsönzések:

```
SELECT k.*, u.nev, f.cim FROM kolcsonzesek k JOIN ugyfelek u ON  
k.ugyfel_id=u.ugyfel_id JOIN filmek f ON k.film_id=f.film_id WHERE k.visszahozva
```

IS NULL AND k.visszahozas_hatarido < CURDATE());

Késedelmi díj számítása (példa logika):

Ha napi díj: 200 Ft/nap, akkor a késedelmi díj: GREATEST(DATEDIFF(CURDATE(), visszahozas_hatarido),0) * napi_dij.

Működése:

- Érdemes triggert írni a kölcsönzések táblához a visszahozva mező frissítésére és a film státuszának automatikus módosítására.
- Későbbi bővítések: díjfizetés rögzítése, több példány kezelése (példányszám), előfizetési modellek, riportok dolgozó teljesítményéről.

Önreflexió:

Az átdolgozott változat célja áttekinthetőbb struktúrát és néhány gyakorlati továbbfejlesztési javaslatot adni (pl. státuszok, késedelmi díj mező). A projekt tanulsága továbbra is az, hogy a relációs modellezés és a jó normalizáció fontos a későbbi lekérdezések és karbantartás miatt.