Kino lístky

Správa(1)

projekt na Databázy (2)

Matej Komlóssy

2.ročník

5.5.2020

O čom je moja aplikácia

Moja aplikácia s názvom Kino lístky slúži na prevádzkovanie kina pomocou databázového systému.

V databáze su uložené údaje o zákazníkoch, ich objednávkach, a tiež o predstaveniach a filmoch.

Zákazníci si môžu objednávať lístky na jednotlivé predstavenia. Predstavenia sa odohrávajú v kinosálach. Po vytvorení objednávky na ňu môžu uplatniť zľavové kupóny pre zníženie ceny.

Taktiež si môžu pozrieť štatistiky týkajúce sa najúspešnejších filmov.

Prevádzkovateľ kina máa možnosť predstavenie zrušiť. Vtedy sú zákazníkovi odporučené iné predstavenia alebo ponúknutá kompenzácia v podobe kupónu.

Štruktúra aplikácie

Vonkajšia štruktúra pozostáva z hlavného menu, ktoré je rozdelené do viacerých menších menu podľa kategórii ako napríklad objednávky či filmy.

Každé submenu je v aplikácii reprezentované vlastnou triedou, v ktorej sa nachádzajú metódy vykonávajúce akcie dostupné v danom menu.

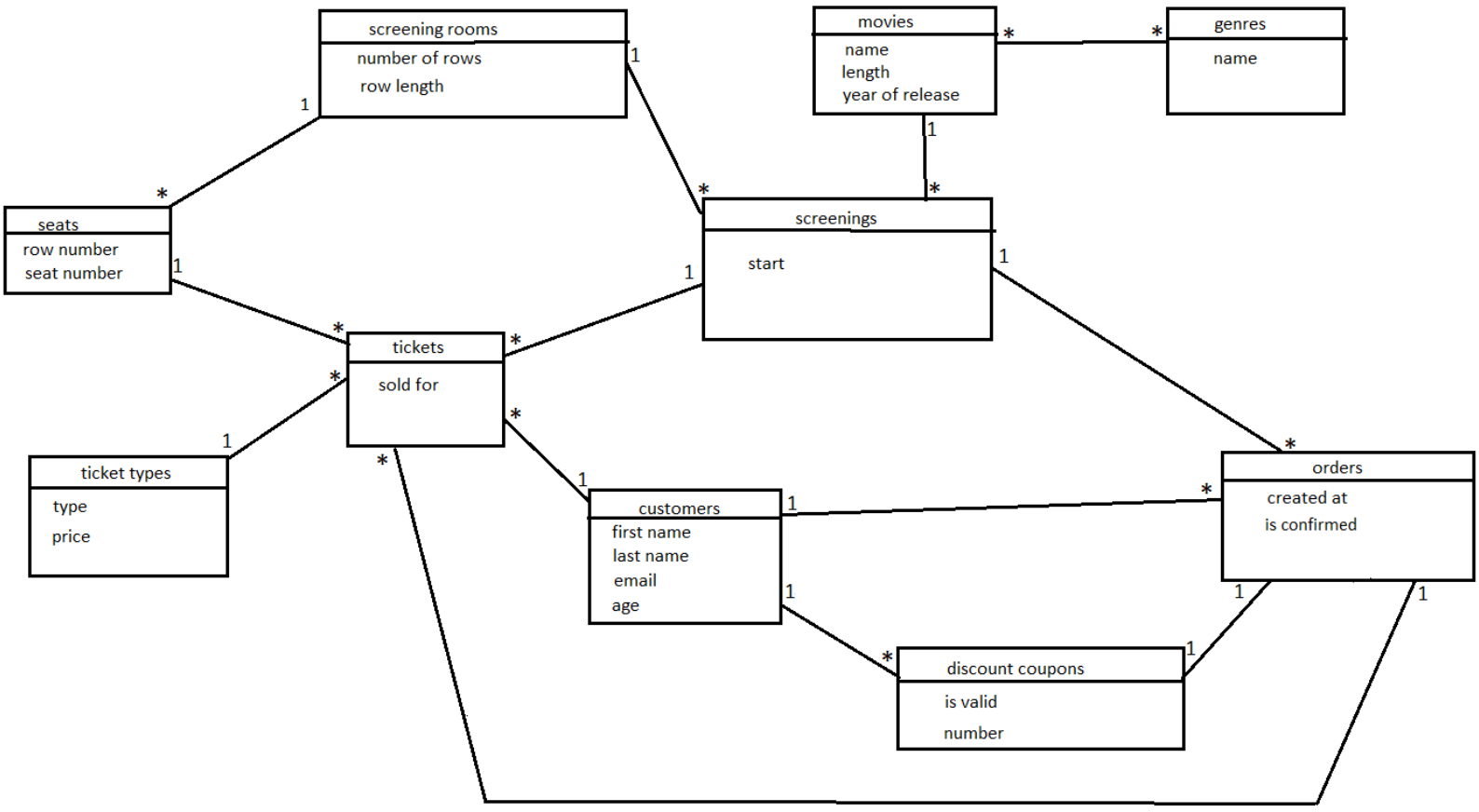
Štruktúra aplikácie sa riadi vzormi Row data gateway a Transaction script.

Podľa vzoru Row data gateway je každá tabuľka reprezentovaná vlastnou triedou, ktorá riadi prístup do nej. Inštancia takejto triedy predstavuje jeden riadok tabuľky.

Podľa vzoru Transaction script sú v aplikácii oddelené metódy, ktoré majú na starosti používateľské rozhranie od metód vykonávajúcich doménové operácie, ktoré sa nachádzajú v oddelených triedach.

Metódy vykonávajúce doménové operácie som rozdelil do tried podľa toho, čoho sa týkajú. (napríklad doménové operácie týkajúce sa objednávok sú v spoločnej triede)

Entitno-relačný model



V množine orders sú uložené objednávky vytvorené zákazníkmi. Pri vytváraní objednávky si zákazník zvolí počet a typy lístkov, ktoré si tak rezervuje.

Pri potvrdzovaní objednávky si zákazník zvolí miesta. Potvrdenej objednávke tak budú priradené lístky podľa zvolených typov lístkov a miest.

Predstavenia sa odohrávajú v kinosálach, počas jedného predstavenia je premietaný jeden film. Dĺžka predstavenia závisí od dĺžky filmu.

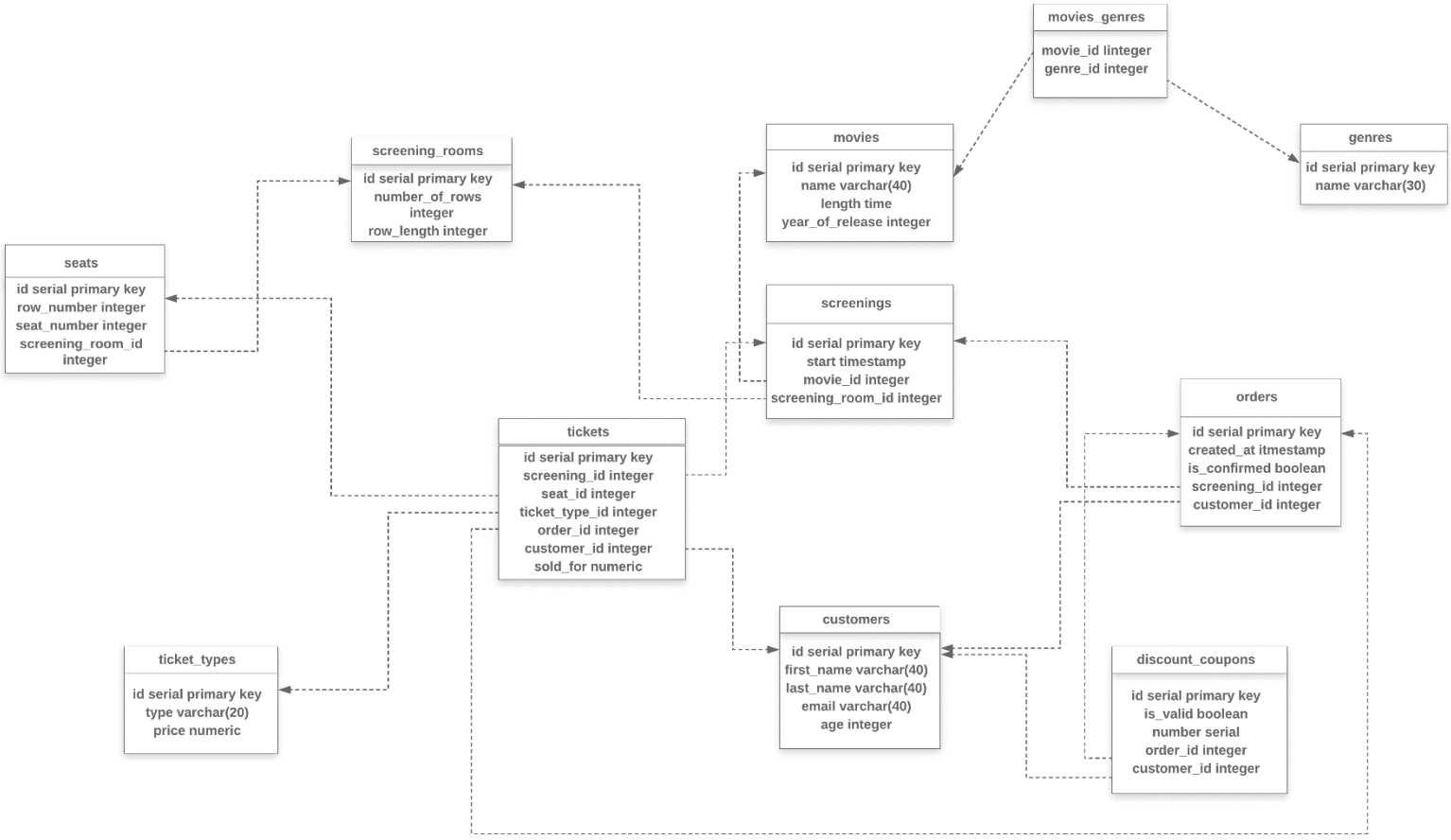
Každý film je jedného alebo viacerých žánrov (množina genres).

Objednávka je vytvorená pre konkrétne predstavenie z množiny screenings. Cena lístkov, a teda aj objednávky závisí od ich typu.

Na objednávku môže zákazník použiť jemu pridelený zľavový kupón (množina discount coupons)

a cenu lístkov objednaných touto objednávkou znížiť.

Transformácia entitno-relačného modelu na relačný model



V relačnom modeli mi vznikla jedna väzobná tabuľka movies_genres, a to zo vzťahu množín movies a genres s kardinalitou M:N. Táto tabuľka dostala dva cudzie kľúče, jeden odkazujúci sa na tabuľku movies a druhý na tabuľku genres. Sú v nej uložené informácie o žánroch filmov.

V entitno-relačnom modeli sa nachádzal vzťah s kardinalitou 1:1 iba medzi dvomi entitami, a to orders a discount_coupons. Na základe tohto vzťahu som sa rozhodol pridať cudzí kľúč odkazujúci sa na tabuľku orders s integritným obmedzením UNIQUE do tabuľky discount_coupons.

Pri vzťahoch s kardinalitou 1:N som vždy pridal cudzí kľúč do tabuľky ktorá vznikla z množiny na strane N, odkazujúci sa na tabuľku ktorá vznikla z množiny na strane 1.

Porovnanie optimalizácie

Nasledovná tabuľka znázorňuje optimalizáciu príkazu, ktorým som hľadal najbližšie predstavenia, ktoré sú podobné zadanému predstaveniu. Podobnosť sa porovnáva podľa prieniku žánrov premietaných filmov.

verzia bez intersect:

```
with zanre_zruseneho as (select genre_id from movies_genres mg where movie_id = 39489)

(select * from screenings s join movies m on s.movie_id = m.id
where s.start > timestamp ' 2018-11-14 23:23:15' --start zruseneho filmu
and s.movie_id = 39489 LIMIT 5)

union

(select * from screenings s join movies m on s.movie_id = m.id
where s.start > timestamp ' 2018-11-14 23:23:15' --start zruseneho filmu
order by s.start asc,
        (select count(*) from movies_genres mg where
        mg.genre_id in (select * from zanre_zruseneho)) desc LIMIT 10)
```

verzia s intersect:

```
with zanre_zruseneho as (select genre_id from movies_genres mg where movie_id = 39489)

(select * from screenings s join movies m on s.movie_id = m.id
where s.start > timestamp ' 2018-11-14 23:23:15' --start zruseneho filmu
and s.movie_id = 39489 LIMIT 5)

union

(select * from screenings s join movies m on s.movie_id = m.id
where s.start > timestamp ' 2018-11-14 23:23:15' --start zruseneho filmu
order by s.start asc,
          (select count(*) from ((select genre_id from movies_genres mg where mg.movie_id = s.movie_id)
           intersect (select * from zanre_zruseneho))as x) desc LIMIT 10)
```

index	verzia s intersect	verzia bez intersect
bez indexov	viac ako 2 minuty	1207 ms
movies_genres(movie_id)	9885 ms	1204 ms
screenings(movie_id)	viac ako 2 minuty	1211 ms
screenings(start)	viac ako 2 minuty	314 ms
movies_genres(movie_id) a screenings(movie_id)	9879 ms	1203 ms
movies_genres(movie_id) a screenings(start)	9885 ms	272 ms
screenings(movie_id) a screenings(start)	viac ako 2 minuty	279 ms
vsetky tri	9871 ms	270 ms

Z tabuľky je vidno, že verzia bez použitia intersect bola rýchlejšia. Zatiaľ čo v pomalšej verzií príkaz bez indexov ani nezbehol, v tejto verzii zbehol v celkom prijateľnom čase. Ako najlepšia verzia sa

ukázala verzia so všetkými indexami (na stĺpcoch movies_genres.movie_id, screenings.start, screenings.movie_id). V rýchejšej verzii sa ako kľúčový ukázal index na stĺpci screenings.start, ktorý priniesol najväčšie zrýchlenie.

Riešený problém

Najzložitejšie bolo pre mňa vymyslieť, ako zabezpečiť opakovanie výberu miest, ak si počas neho miesta zarezervoval iný zákazník. Problém som vyriešil aplikačnou transakciou.

Celý proces od vytvorenia objednávky až po jej potvrdenie mohol byť uzatvorený v jednej databázovej transakcii. Toto riešenie však nebolo vhodné, lebo zákazník si miesta môže vyberať dlho a bolo by preto veľmi neefektívne.

Radšej som teda vytváranie objednávky aj jej potvrdzovanie uzatvoril do zvlášť databázových transakcii, samotné vyberanie miest v nej nie je. Aplikačná transakciai je realizovaná tak, že pri potvrdzovaní skontrolujem, či sú vybrané miesta stále voľne a ak nie, upozorním používateľa a nechám ho vybrať si znova.

Naučil som sa teda, ako a v akých situáciách používať databázové transakcie, kedy je potrebné ich použiť.