Online hra

KIV/DB1 - Semestrální práce

student:	Jan Hinterholzinger
studijní číslo:	A19B0052P
email:	hintik@students.zcu.cz
datum:	18. 01. 2021

FAKULTA APLIKOVANÝCH VĚD ZÁPADOČESKÉ UNIVERZITY V PLZNI

Databázové systémy I

zimní semestr 2020/2021

Jméno a příjmení: Jan Hinterholzinger

Osobní číslo: A19B0052P

Kontaktní e-mail: hintik@students.zcu.cz

Orion login: hintik

Obor/kombinace: IVTB-INF18

Rok studia: 2.

Název práce: Databáze online hry

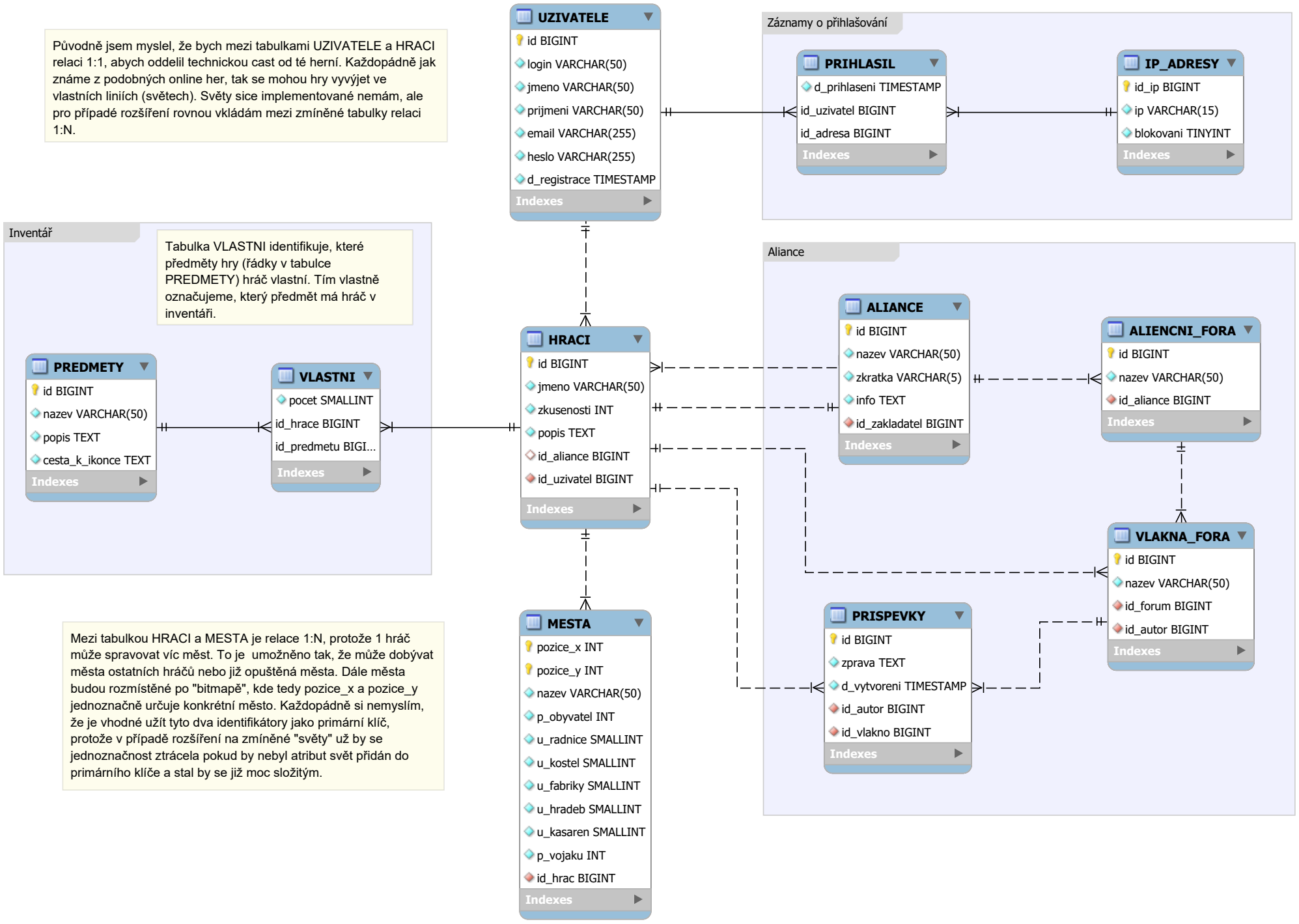
Chtěl jsem si zkusit vytvořit model nějaké online hry, kde by si hráči spravovali své město a bojovali mezi sebou. Nakonec mi z toho vyšlo něco připomínající hru Divoké Kmeny. Hra spočívá v budování vlastního města a vytváření vojenských jednotek, se kterými může hráč zaútočit na ostatní hráče s cílem vyrabovat zásoby nebo získat kontrolu nad městem. Hráči se dále mohou seskupovat do aliancí (celky hráčů bojujících pospolu). Hráč má také inventář předmětů, které sbírá v průběhu hry.

Při řešení mého zadání pravděpodobně použiji (ponechte tu variantu, kterou chcete použít):

- SŘBD MariaDB na svém hardware (PC, notebook či tablet)

Při řešení mého zadání pravděpodobně budu řešit tyto čtyři dotazy:

- Selekce potřebných informací pro stručný přehled informací o hráči.
  - Seznam měst hráče s vypsanými informacemi o městě (např. počet obyvatel a aktuální počet zásob).
  - Jméno, popis (bio), počet zkušenostních bodů a případně aliance
- Přihlášení uživatele, kde se přihlášení zapíše do tabulky PRIHLASIL a pokud tabulka IP_ADRESY neobsahuje aktuální IP adresu, tak ji vloží. (pokud zvládnu nacpat do jednoho dotazu)
- Vložení předmětu do inventáře hráče. Pokud hráč již předmět má, tak se inkrementuje počet kolikrát daný předmět vlastní. (pokud zvládnu nacpat do jednoho dotazu)
- Výpis přehledu přihlášení (datum přihlášení a IP adresa případně s informaci o použitém prohlížeči)


Původně jsem myslel, že bych mezi tabulkami UZIVATELE a HRACI relaci 1:1, abych oddelil technickou cast od té herní. Každopádně jak známe z podobných online her, tak se mohou hry vyvýjet ve vlastních liniích (světech). Světy sice implementované nemám, ale pro případé rozšíření rovnou vkládám mezi zmíněné tabulky relaci 1:N.
UZIVATELE
id BIGINT
login VARCHAR(50)
jmeno VARCHAR(50)
prijmeni VARCHAR(50)
email VARCHAR(255)
heslo VARCHAR(255)
d_registrace TIMESTAMP
Indexes
Záznamy o přihlašování
PRIHLASIL
d_prihlaseni TIMESTAMP
id_uzivatel BIGINT
id_adresa BIGINT
Indexes
IP_ADRESY
id_ip BIGINT
ip VARCHAR(15)
blokovani TINYINT
Indexes
Inventář
Tabulka VLASTNI identifikuje, které předměty hry (řádky v tabulce PREDMETY) hráč vlastní. Tím vlastně označujeme, který předmět má hráč v inventáři.
PREDMETY
id BIGINT
nazev VARCHAR(50)
popis TEXT
cesta_k_ikonce TEXT
Indexes
VLASTNI
pocet SMALLINT
id_hrace BIGINT
id_predmetu BIGI...
Indexes
HRACI
id BIGINT
jmeno VARCHAR(50)
zkusenosti INT
popis TEXT
id_aliance BIGINT
id_uzivatel BIGINT
Indexes
Aliance
ALIANCE
id BIGINT
nazev VARCHAR(50)
zkratka VARCHAR(5)
info TEXT
id_zakladatel BIGINT
Indexes
ALIENCNI_FORA
id BIGINT
nazev VARCHAR(50)
id_aliance BIGINT
Indexes
VLAKNA_FORA
id BIGINT
nazev VARCHAR(50)
id_forum BIGINT
id_autor BIGINT
Indexes
PRISPEVKY
id BIGINT
zprava TEXT
d_vytvoreni TIMESTAMP
id_autor BIGINT
id_vlakno BIGINT
Indexes
Mezi tabulkou HRACI a MESTA je relace 1:N, protože 1 hráč může spravovat víc měst. To je umožněno tak, že může dobývat města ostatních hráčů nebo již opuštěná města. Dále města budou rozmístěné po "bitmapě", kde tedy pozice_x a pozice_y jednoznačně určuje konkrétní město. Každopádně si nemyslím, že je vhodné užít tyto dva identifikátory jako primární klíč, protože v případě rozšíření na zmíněné "světy" už by se jednoznačnost ztrácela pokud by nebyl atribut svět přidán do primárního klíče a stal by se již moc složitým.
MESTA
pozice_x INT
pozice_y INT
nazev VARCHAR(50)
p_obyvatel INT
u_radnice SMALLINT
u_kostel SMALLINT
u_fabriky SMALLINT
u_hradeb SMALLINT
u_kasaren SMALLINT
p_vojaku INT
id_hrac BIGINT
Indexes

Kapitola 1

Popis a charakteristika zadání SP

Databázový model webové online hry. Jedná se o typickou online hru inspirovanou úspěšnou hrou Divoké kmeny. Hráč vlastní města, která postupně buduje a dobývá. Na mapě plné hráčů může obchodovat, útočit nebo spojit se do aliancí. Zkrátka obyčejná hra takového typu. Realizovaný model je pouze fragmentem celé dospělé hry.

Zadání

Uživatel se může přihlásit a aplikace bude jeho záznamy o přihlášení ukládat společně s IP adresou, se kterou se připojil, pro monitorování bezpečnostních rizik. Uživatel také má možnost mít několik herních postav, kvůli případnému budoucímu rozšíření o herní světy.

Každá herní postava zprvu vlastní jedno město, o které se stará (staví budovy, rekrutuje vojenské jednotky, vybírá daně). Jednotlivá města jsou rozmístěna na bitmapě, kde každé město má svá jednoznačné souřadnice. Hráč může získat více měst dobýváním okolních měst dalších hráčů. Pokud hráč ztratí všechna města a nemá žádné, tak pro něj hra končí.

Hráč má k dispozici také inventář, ve kterém shromažďuje předměty, které v průběhu hry získá a se kterými by mohl obchodovat. Inventář je neomezený a může vlastnit neomezené množsví všech herních předmětů.

Pro sjednocení sil, mohou hráči vstupovat do aliancí, kde se mohou na aliančním fóru v jednotlivých vláknech domlouvat o herní strategii.

Kapitola 2

Reprezentativní dotazy

2.1 Získání informací o hráči

2.1.1 Seznam měst hráče s vypsanými informacemi

Je vhodné hráči ukázat přehled jeho vlastněných měst, aby hráč měl přehled bez zbytečného překlikávání. Proto je připraven dotaz, který vypíše seznam jeho měst společně s úrovněmi jednotlivých budov, počtem obyvatel a počtem vojenských jednotek.

Pro předvedení použiji testovacího uživatele s identifikačním číslem 1.

SQL dotaz:

```
SELECT m.nazev as 'Název města', m.p_obyvatel as 'Počet obyvatel',
m.p_vojaku as 'Počet vojenských jednotek', m.u_radnice as 'Úroveň
radnice', m.u_kostel as 'Úroveň kostela', m.u_kasaren as 'Úroveň
kasáren', m.u_fabriky as 'Úroveň fabriky', m.u_hradeb as 'Úroveň hradeb'
from MESTA m where m.id_hrac = 1;
```

SQL dotaz (zkráceně):

```
SELECT m.nazev, m.p_obyvatel, m.p_vojaku, m.u_radnice, m.u_kostel,
m.u_kasaren, m.u_fabriky, m.u_hradeb from MESTA m where m.id_hrac = 1;
```

Výsledek dotazu:

Název města	Počet obyva-tel	Počet vojen. jedno.	Úroveň radnice	Úroveň kostela	Úroveň kasáren	Úroveň fabriky	Úroveň hradeb
Velkoměsto	1002	105	5	6	3	2	4
Maloměsto	302	15	1	2	1	1	0

2.1.2 Informace o profilu hráče

Uživatel si také může rozkliknout svůj profil hráče, kde se ukáží informace ohledně jeho stavu hry.

SQL dotaz:

```
SELECT h.jmeno as 'Přezdívka', h.zkusenosti as 'Zkušenostní body',
COUNT(id_hrac) as 'Počet měst', SUM(m.p_obyvatel) as 'Celkem poddaných',
```

```
a.nazev as 'Aliance', COUNT(p.id) as 'Počet příspěvků' from HRACI h left
join MESTA m on h.id = m.id_hrac inner join ALIANCE a on h.id_aliance =
a.id left join PRISPEVKY p on h.id = p.id_autor WHERE h.id = 1;
```
SQL dotaz (zkráceně):
```
SELECT h.jmeno, h.zkusenosti, COUNT(id_hrac), SUM(m.p_obyvatel), a.nazev,
COUNT(p.id) from HRACI h left join MESTA m on h.id = m.id_hrac inner join
ALIANCE a on h.id_aliance = a.id left join PRISPEVKY p on h.id =
p.id_autor WHERE h.id = 1;
```
Výsledek dotazu:

Přezdívka	Zkušenostní body	Počet měst	Celkem poddaných	Aliance	Počet příspěvků
hintik	30056	4	2608	Velkoobchodní Aliance	4

2.2 Výpis seznamu přihlášení

Hráč by měl mít možnost zobrazit si jeho seznam přihlášení, aby měl jistotu, že za jeho herní účet nehraje i někdo jiný. To je dosaženo výpisem seznamu přihlášení společně se zdrojovou IP adresou.

SQL dotaz:
```
SELECT p.d_prihlaseni as 'Datum a čas přihlášení', ip.ip as 'Adresa
připojení' FROM PRIHLASIL p inner join IP_ADRESY ip on p.id_adresa =
ip.id_ip where p.id_uzivatel = 1;
```
SQL dotaz (zkráceně):
```
SELECT p.d_prihlaseni, ip.ip FROM PRIHLASIL p inner join IP_ADRESY ip on
p.id_adresa = ip.id_ip where p.id_uzivatel = 1;
```
Výsledek dotazu:

Datum a čas přihlášení	Adresa připojení
2021-01-15 12:58:05	192.168.0.1
2021-01-15 22:23:53	192.168.5.3
2021-01-16 10:42:13	192.168.0.1
2021-01-16 12:32:39	10.10.10.10
2021-01-17 16:15:42	10.10.10.10

Kapitola 3

Předpřipravené scénáře

3.1 Přihlášení uživatele

Do webové aplikace se přihlašují uživatelé. Tito uživatelé se přihlašují z nějaké sítě nějakou IP adresou. Vývojář webové aplikace by měl počítat s útoky na aplikaci a útoky na uživatele. Například některý uživatel z některé IP adresy vytváří velké množsví falešných účtů nebo provádí DDoS útok. Jedna z možných obranných strategií je zablokovat IP adrese, ze které se připojuje. Útok na uživatele může začít odcizením hesla, tím má útočník plný přístup k ovládání uživatelského účtu. Takové chvování může být odhaleno například logem připojení, kdy uživatel vidí, že se na účet připojil z adresy, ze které se pravý uživatel nikdy nepřipojuje a může tak odhalit útočka.

Pro tyto ochranné mechanismy je tedy vhodné každé přihlášení zapisovat a přikládat čas a IP adresu připojení. V tomto scénáři bych rád simuloval zaznamenání přihlášení. Úkolem je tedy přiřadit uživatele k IP adrese a čase. Máme tedy k dispozici identifikátor (například `1`) uživatele a IP adresu z které se připojuje (např `192.168.255.1`).

V první řadě nemáme jistotu, že adresa, se kterou se uživatel přihlašuje není v databázi a je potřeba ji přidat. Pokud bychom adresu nepřidali, tak nám integritní omezení nedovolí vložit záznam do tabulky `PRIHLASIL`.

SQL příkaz:

```
INSERT INTO IP_ADRESY (ip) VALUES ('192.168.255.1') ON DUPLICATE KEY
UPDATE ip='192.168.255.1';
```

Tento SQL příkaz přidá do tabulky záznam s IP adresou pokud nějaký záznam s takovou IP adresou již neexistuje. Pokud ano, tak provede příkaz UPDATE, kdy se hodnota adresy změní na stejnou hodnotu.

Nyní již nám nic nebrání zaznamenat přihlášení, protože IP adresa již existuje a uživatel (jelikož se právě přihlásil) také.

Následující příkaz tedy přidá do „pomocné“ tabulky pro uskutečnění M:N vazby (tabulka `PRIHLASIL`) identifikátor uživatele a identifikátor IP adresy. Časový údaj v tabulce se doplní sám díky výchozí hodnotě sloupce. Jelikož neznáme přímo identifikátor IP adresy, tak spustíme příkaz SELECT na jeho zjištění.

SQL příkaz:

```
INSERT INTO PRIHLASIL (id_uzivatel, id_adresa) VALUES (1, (SELECT id_ip
from IP_ADRESY where ip = '192.168.255.1'));
```

3.2 Přidání předmětu do inventáře

Jednou z herních mechanik je i hráčský inventář. Jedná se o podmnožinu všech předmětů ve hře s tím, že hráč může mít předměty i v nějakém větším počtu než je jeden.

Opět se tedy jedná o relaci M:N mezi tabulkami `HRACI` a `PREDMETY`. Pomocnou rozkladovou tabulkou je tabulka `VLASTNI`, která drží informaci, který hráč vlastní který předmět a v jakém počtu.

Přidání předmětů do hráčského inventáře tedy budeme reprezentovat buď přidáním záznamu do tabulky `VLASTNI`, nebo inkrementovaním položky `pocet` v případě, že záznam již v tabulce existuje.

SQL příkaz:

```
INSERT INTO VLASTNI (id_hrace, id_predmetu) VALUES (1,1) ON DUPLICATE KEY
UPDATE pocet = pocet + 1;
```

Tento příkaz se tedy snaží přidat záznam, kde spojuje hráče (s identifikátorem 1) a předmět (s identifikátorem 1). Pokud ale tato kombinace klíče již existuje, tak se inkrementuje položka `pocet` označující počet vlastněných předmětů.

3.3 Odebrání předmětu z inventáře

Obdobný problé jako v předchozím případě a přitom úplně jiný. Potřebujeme totiž rozhodovat, zda záznam dekrementovat (zmenšovat o jedna) v případě, kdy počet vlastněných předmětů je větší než 1. Nebo zda odstranit celý záznam pokud je počet roven jedné (po odstranění by byl roven nule).

Jako první se nabízí zjistit, jaký je aktuální počet vlastněných předmětů daného typu. To by vyřešil příkaz `SELECT` a na základě této hodnoty bychom rozhodli, zda spustit příkaz `UPDATE` nebo `DELETE`.

Každopádně to lze provést i jinak, a to bez příkazu `SELECT` a s vhodými výrazy za klauzulí `WHERE`. Můžeme totiž vyslat dva příkazy `UPDATE`, který sníží hodnotu `pocet` za předpokladu, že je větší než 1 a příkaz `DELETE`, který smaže záznam, pokud je hodnota `pocet` roven 1.

SQL příkazy:

1. `DELETE FROM VLASTNI WHERE id_hrace = 1 AND id_predmetu = 1 AND pocet = 1;`
2. `UPDATE VLASTNI SET pocet = pocet - 1 WHERE id_hrace = 1 AND id_predmetu = 1 AND pocet > 1;`

Těmito dvěmi příkazy provedené v tomto pořadí by odebraly jeden kud předmětu z inventáře hráče. Opět také předpokládáme, že známe identifikátor hráče, kterému odstraňujeme a identifikátor předmětu, který odstraňujeme.

Kapitola 4

Závěr

Výsledná práce obsahuje databázovou strukturu o 11 tabulkách, do kterých je zadání příměřeně rozděleno. Tabulky jsou propojeny cizími klíči a dalšími integritními omezeními.

Vytvořené dotazy splňují požadavky na složitost a propojení více tabulek. Jsou zde použity také funkce `COUNT(...)` a `SUM(...)` použité pro výspis počtu a součtu.

Práce celkově dodržuje minimálně 3. normální formu a zároveň se drží zadání.

Testovací data jsou poněkud chudá, ale postačují na ověření funkčnosti práce.