Vysoké učení technické v Brně
Fakulta informačních technologií

IDS - Databázové systémy

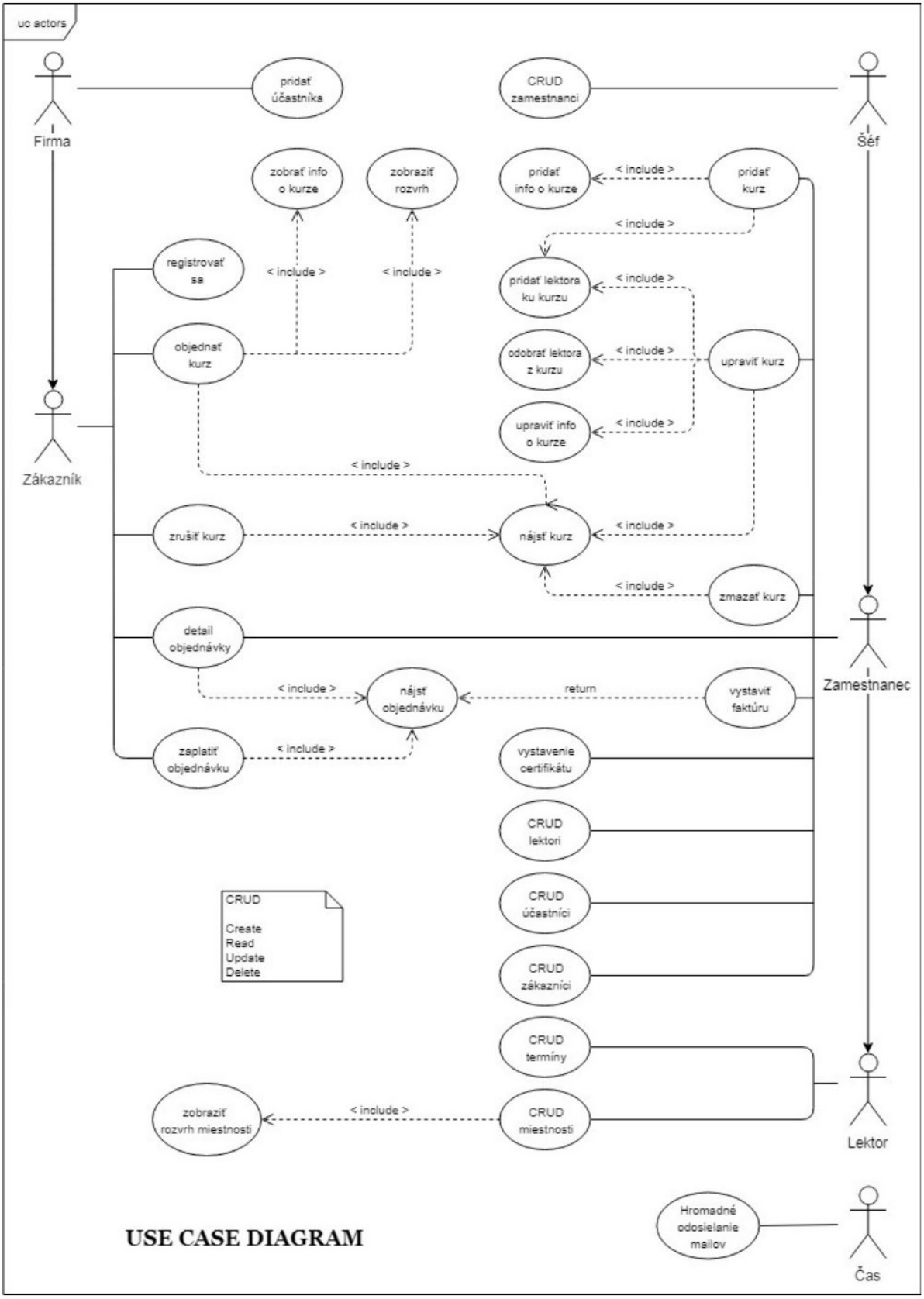
Datový model školícího střediska

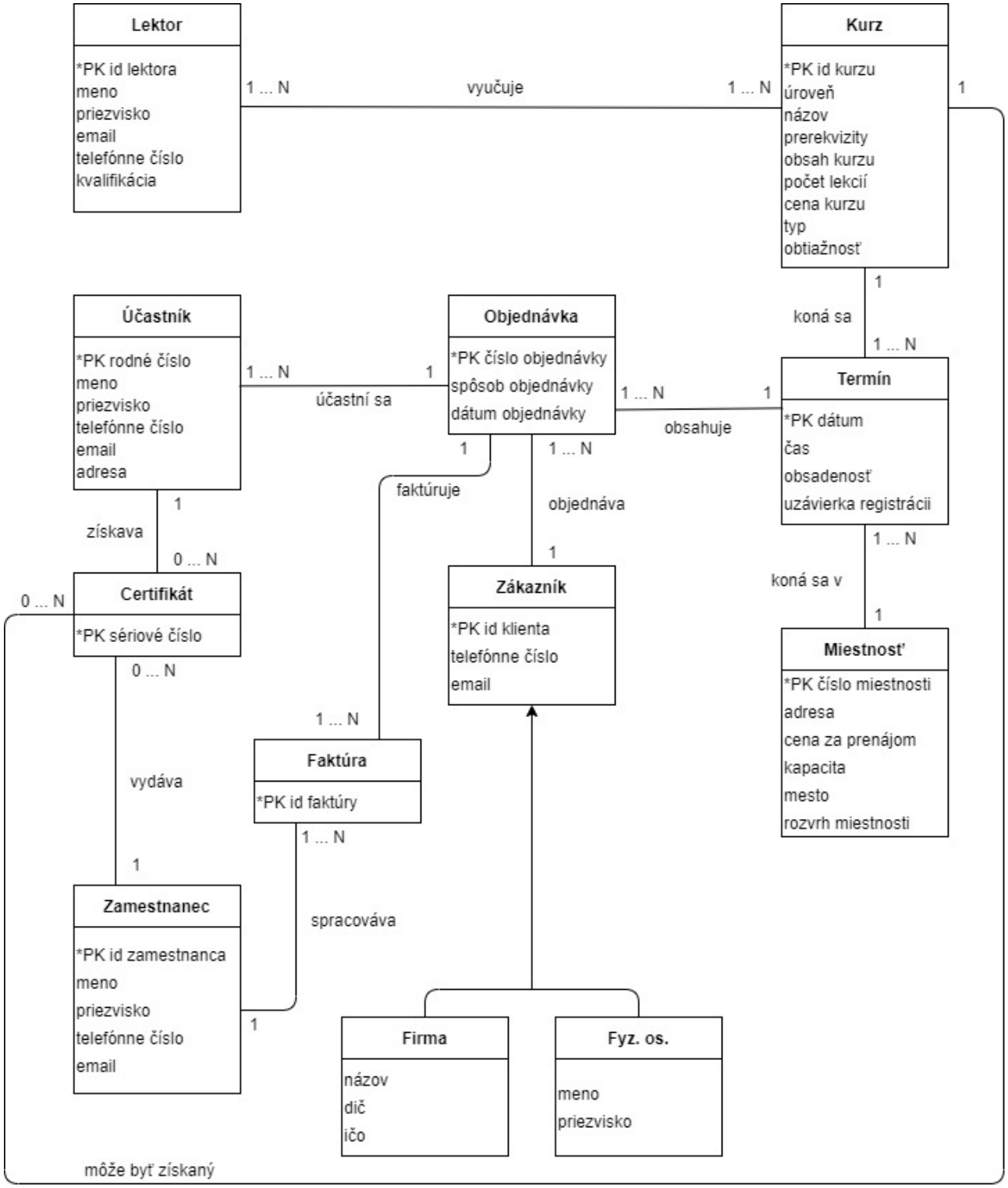
Ďurovič Róbert, xdurov01

Brno, 30. dubna 2020

1 Školící středisko

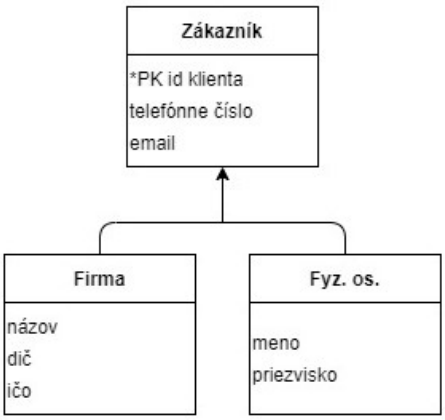
Navrhněte jednoduchý IS malého školicího střediska, které organizuje kurzy orientované na programové vybavení osobních počítačů. Kurzy probíhají v různých městech v pronajatých místnostech a jsou vypisovány přímo střediskem nebo na základě objednávky zákazníka (firma objednávající kurz pro své zaměstnance). Firma si může objednat určitý typ kurzu, u kterého jsou k dispozici informace o jeho úrovni, obtížnosti, prerekvizitách, ceně, počtu a obsahu lekcí. Konkrétní kurzy určitého typu, které si zákazník/firma objednává jsou vedeny jedním lektorem, mají maximální kapacitu účastníků a odehrávají se v určité místnosti v konkrétní čas a den v týdnu. U vypsaných kurzů lze dohledat jaké lekce obsahují a datum jejich konání. Školicí středisko si může pronajímat místnosti v různých budovách (i v různých městech), přičemž každá místnost má stanovenu cenu za hodinu a maximální kapacitu. Zároveň předpokládejte, že jeden lektor může být vyškolen pro více typů kurzů, toto modelujte. Účastníci po absolvování kurzu obdrží certifikát se svým jménem na svou fyzickou adresu. Systém musí být na požádání schopen vypsat rozvrh pro jednotlivé místnosti. Zaměstnanec školicího střediska má možnost vložit do systému nové typy kurzu a konkrétní kurzy. Lektor může měnit čas a místnost konání konkrétního kurzu. Zákazník si může vypsat u kurzů, které platí, své přihlášené zaměstnance (zaměstnanec si však může kurz platit sám a tudíž by neměl být takto dohledatelný svým zaměstnavatelem, a také může změnit zaměstnavatele). Potenciální účastník kurzu si může přes webové rozhraní vypsat rozvrh vypisovaných kurzů, zjistit počet volných míst v kurzu a zaregistrovat se.


uc actors
Firma
pridať účastníka
CRUD zamestnanci
Šéf
zobrať info o kurze
zobraziť rozvrh
pridať info o kurze
< include >
pridať kurz
registrovať sa
pridať lektora ku kurzu
objednať kurz
odobrať lektora z kurzu
upraviť kurz
upraviť info o kurze
Zákazník
zrušiť kurz
nájsť kurz
zmazať kurz
detail objednávky
Zamestnanec
nájsť objednávku
return
vystaviť faktúru
zaplatiť objednávku
vystavenie certifikátu
CRUD lektori
CRUD účastníci
CRUD zákazníci
CRUD
Create
Read
Update
Delete
CRUD termíny
zobraziť rozvrh miestnosti
CRUD miestnosti
Lektor
Hromadné odosielanie mailov
Čas
USE CASE DIAGRAM


Lektor
*PK id lektora
meno
priezvisko
email
telefónne číslo
kvalifikácia
1 ... N
vyučuje
1 ... N
Kurz
*PK id kurzu
úroveň
názov
prerekvizity
obsah kurzu
počet lekcií
cena kurzu
typ
obtiažnosť
1
1
koná sa
1 ... N
Účastník
*PK rodné číslo
meno
priezvisko
telefónne číslo
email
adresa
1 ... N
účastní sa
1
Objednávka
*PK číslo objednávky
spôsob objednávky
dátum objednávky
1 ... N
obsahuje
1
Termín
*PK dátum
čas
obsadenosť
uzávierka registrácii
1
faktúruje
1 ... N
objednáva
1 ... N
koná sa v
1
1
získava
0 ... N
0 ... N
Certifikát
*PK sériové číslo
Zákazník
*PK id klienta
telefónne číslo
email
Miestnosť
*PK číslo miestnosti
adresa
cena za prenájom
kapacita
mesto
rozvrh miestnosti
0 ... N
1 ... N
Faktúra
*PK id faktúry
vydáva
1
1 ... N
Zamestnanec
*PK id zamestnanca
meno
priezvisko
telefónne číslo
email
spracováva
1
Firma
názov
dič
ičo
Fyz. os.
meno
priezvisko
môže byť získaný

Generalizácia/špecializácia

V našom zadaní sme problém generalizácie/špecializácie riešili na prípade zákazníka. Zákazník môže byť ako fyzická osoba, tak aj firma. Typicky sa to rieši pomocou troch tabuliek: `Zákazník`, `Fyzická osoba` a `Firma`, kde v posledných dvoch tabuľkách by cudzím kľúčom bol primárny kľúč tabuľky `Zákazník`.



Implementácia SQL skriptu

Skript na začiatku zahodí pomocou príkazu `DROP` databázové objekty, aby sme predišli možným konfliktom. Následne sa vytvoria tabuľky pomocou príkazu `CREATE TABLE`, nastavia kľúče, sekvencie, triggery a procedúry. Skript nakoniec naplní tabuľky testovacími dátami, nad ktorými potom prevádzame na ukážku niekoľko príkazov `SELECT` a procedúr.

Triggery

Projekt obsahuje celkovo 3 triggery, pričom dva z nich sa spúšťajú po vložení alebo aktualizácii dát do tabuľky (`AFTER INSERT OR UPDATE`) a jeden sa spúšťa pred vložením (`BEFORE INSERT OR UPDATE`). Prvý z týchto triggerov slúži na validáciu dát u termínu kurzu. V prípade,že vkladáme do tabuľky dátum z minulosti alebo dátum konkrétneho dňa, napr. 24/12/2018, na vystúpe sa nám objaví chybové hlásenie.

Druhý trigger validuje rodné číslo účastnika kurzu. Kontrolujeme či je toto číslo deliteľné jedenástimi. Z čísla postupne extrahujeme deň, mesiac i rok. A na základe týchto údajov potom ešte zisťujeme či ide o ženu alebo nie v prípade,že mesiac > 50. Ak nastane niekde chyba a číslo nie je validné, na výstupe sa objaví chybové hlásenie.

Posledný trigger slúži na autoinkrementáciu identifikátora v tabuľke `Zamestnanec` v prípade, že pri v kladaní dát nie je `id` určené.

Procedúry

Projekt obsahuje celkovo 2 netriviálne procedúry s využitím `CURSOR`u a dátovým typom odkazujúcim na riadok v tvare nazov `tabulky%ROWTYPE`. Prvá procedúra počíta priemernú cenu jednej lekcie z vybraného kurzu, ktorý nám bude slúžiť ako vstupný parameter. Nadfinujeme si pomocné premenné

`num_lectures` (počet lekcií) a `overall_price` (celková cena kurzu), ktoré spočítavame v cykle. Po skončení cyklu spravíme aritmetický priemer delením týchto premenných. V prípade, že by sa delilo nulou,procedúra vypíše chybu. Nakoniec výsledok ešte zaokrúhlime, aby sme dostali číslo v priateľnom nedesatinnom formáte a hlásenie vypíšeme na výstup.

Druhá procedúra počíta celkovú percentuálnu vyťaženosť miestností v danom meste. Postup je podobný ako v prvej procedúre. Vstupný parametrom je mesto, v ktorom sa miestnosti nachádzajú. Máme nadefinované premenné `cap_classroom` (aktualna obsadenost kurzu) a capacity (maximálna kapacita miestnosti). Cez cyklus spočítame tieto hodnoty pre dané mesto. Aby sme získali výsledok do premennej `overall_percentage` v percentách, musíme tie premenné podeliť a následne násobiť 100. Pre výsledok v priateľnom nedesatinnom formáte ešte zaokrúhlime a vypíšeme na výstup. V prípade delenia nulou procedúra vypíše chybovú hlášku.

Explain plan

Vďaka `Explain plan` nám databáza ukáže ako spracováva určitý dotaz. Na základe toho môžeme sledovať spracovanie dotazu `SELECT` nad tabuľkami `Lektor` a `Kurz`, kde sa dotazujeme na celkový počet kurzov vyučovaných lektorom a celkovú cenu týchto kurzov.

```
-------------------------------------------------------------------------------
| Id  | Operation           | Name   | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT    |        |     9 |   963 |     7  (15)| 00:00:01 |
|   1 |  HASH GROUP BY      |        |     9 |   963 |     7  (15)| 00:00:01 |
|*  2 |   HASH JOIN         |        |     9 |   963 |     6   (0)| 00:00:01 |
|   3 |    TABLE ACCESS FULL| LEKTOR |     3 |   243 |     3   (0)| 00:00:01 |
|   4 |    TABLE ACCESS FULL| KURZ   |     9 |   234 |     3   (0)| 00:00:01 |
-------------------------------------------------------------------------------
```

V znázornenej tabuľke vidíme niekoľko operácii, ktoré si vysvetlíme detailnejšie. SELECT STATEMENT znamená vykonaný `SELECT` dotaz. `HASH GROUP BY` nám grupuje položky podľa hashovacieho kľúča. `TABLE ACCESS FULL` značí prechod tabuľkou bez použitia indexov.

`EXPLAIN PLAN` ale neprevedie dotaz, iba vyvolá optimalizátor a vysvetlenie dotazu nie je zobrazené, ale uložené do systémovej tabuľky. Vysvetlené dotazy však možno získať tabuľkami s rôznou podrobnosťou. Taktiež je využitý databázový `INDEX`, ktorý slúži k zrýchlenému prístupu k dátam.

Materializovaný pohľad

V projekte sme implementovali materializovaný pohľad, ktorý patrí druhému členovi tímu. Ten používa tabuľky nadefinované prvým členom. Najprv sme teda vytvorili materializované logy, ktoré uchovájú zmeny hlavnej tabuľky a použili `REFRESH FAST ON COMMIT`, aby sa nemusel dotaz spúšťať celý. `CACHE` nám zabezpečuje optimalizáciu čítania z pohľadu a vďaka `BUILD IMMIDIATE` sa nám materializovaný pohľad okamžite naplní. `ENABLE QUERY REWRITE` znamená použitie materializovaného pohľadu optimalizátorom. Pohľad sme potom otestovali na priloženom ukážkovom dotaze a cez `COMMIT` potvrdili zmeny.

Záver

Projekt som spracoval a otestoval v prostredí Oracle na serveroch Oracle12c pomocou softwaru SQL-Developer a VS Code. K projektu stačili vedomosti z projektu IDS nadobudnuté na prednáškach a demonštračných cvičeniach, ale hodili sa i znalosti nadobudnuté vo firemnej praxi.