VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií

DATABÁZOVÉ SYSTÉMY

2021

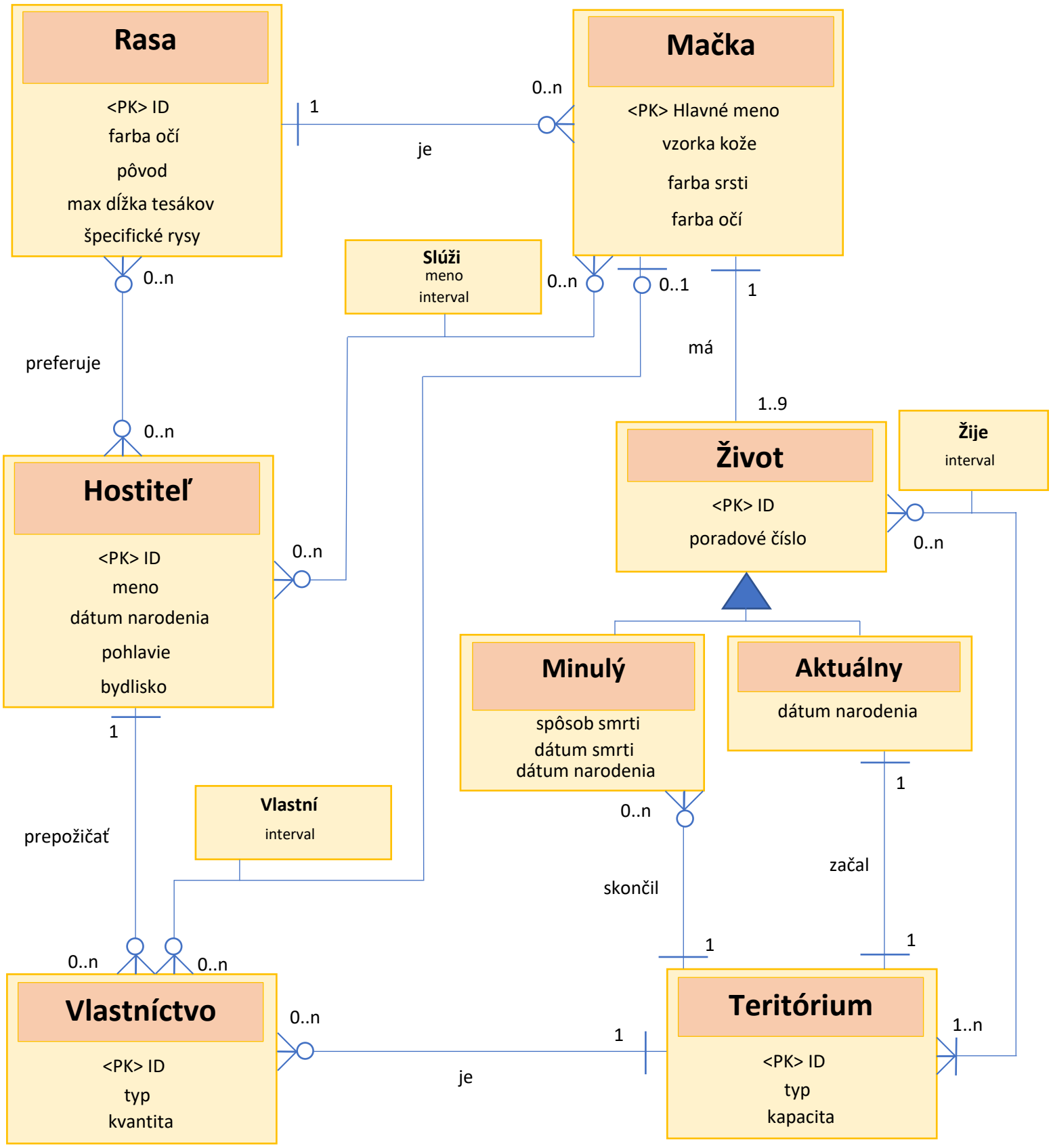
Projekt 1. část – Datový model (ERD), model případů užití a obhajoba modelů

Rebeka Černianska (xcerni13)
Peter Rúček (xrucek00)

48. Kočičí Informační Systém

Kočky chtějí zefektivnit jejich dominanci lidského světa a proto Vám zadaly vytvořit KIS (Kitty Information System). Tento systém uchovává informace o jednotlivých rasách koček, jejich specifické rysy, jako možné barvy očí, původ, maximální délku tesáků, apod. a u konkrétních koček pak jejich hlavní jméno, vzorek kůže, barvu srsti a pod. Každá kočka má právě devět životů, nicméně v systému vedeme pouze ty, které již proběhly a aktuálně probíhají, a vedeme u nich informaci o délce života, místo narození a případně (u minulých životů) o místě (v rámci, kterého teritoria) a způsobu smrti. Kočky jsou samozřejmě majetnické a chtějí si vést všechny teritoria (máme teritoria různých typů, jako např. jídelna, klub, .), ve kterých se kdy pohybovaly a které věci si přivlastnily a v kterém intervalu je vlastnily (kočky se lehce znudí a své věci prostě zahodí). Systém rovněž vede informace o jejich hostitelích, kteří jim slouží. Veďte u nich jejich základní informace (jméno, věk, pohlaví, místo bydlení, .), které rasy koček preferují a rovněž jméno, kterým kočku nazývali (např. Pan Tlapoň, Bublina, Gaston, .). Některé vlastnictví koček však mohou být propůjčována svým hostitelům. Současně veďte informaci (pokud je přítomna) o teritoriu v rámci kterého se vlastnictví nachází, typ vlastnictví (hračka, cokoliv,.) a jeho kvantitu. Jednotlivá teritoria však mají omezenou kapacitu na kočky a v případě překočení (doslova) se kočky přesídlí. Systém umožňuje kočkám zasílat pravidelné novinky o životech ostatních koček a nových dostupných hostitelích, ke kterým by se mohly přesídlit a věcech, které by mohly zabrat.

ERD


Rasa
<PK> ID
farba očí
pôvod
max dĺžka tesákov
špecifické rysy
Mačka
<PK> Hlavné meno
vzorka kože
farba srsti
farba očí
1
je
0..n
0..n
preferuje
0..n
Slúži
meno
interval
0..n
0..1
1
má
1..9
Hostiteľ
<PK> ID
meno
dátum narodenia
pohlavie
bydlisko
0..n
Život
<PK> ID
poradové číslo
Žije
interval
0..n
Minulý
spôsob smrti
dátum smrti
dátum narodenia
Aktuálny
dátum narodenia
1
prepožičať
Vlastní
interval
0..n
skončil
začal
1
1
0..n
0..n
Vlastníctvo
<PK> ID
typ
kvantita
0..n
1
je
Teritórium
<PK> ID
typ
kapacita
1..n

Komentár:

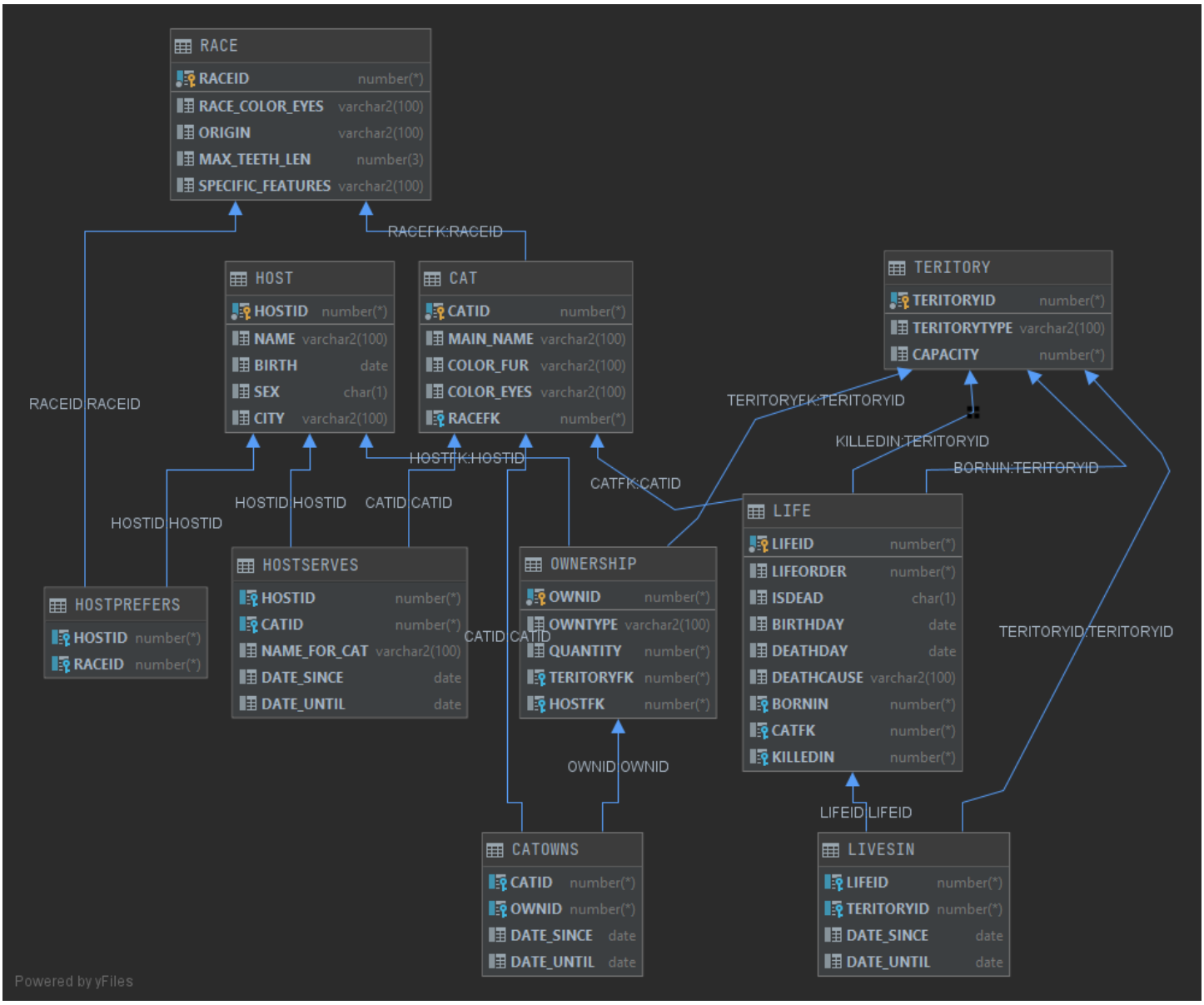
Mačka má 9 životov. Život začne aj ukončí na konkrétnom mieste (teritóriu). Žije v rôznych teritóriách a v atribúte vzťahu si treba uchovať informáciu o intervale žitia v teritóriu. Zaznamenávajú sa aktuálny a minulé životy mačky.

Mačka je práve jednej rasy, pričom hostiteľ môže preferovať rôzne rasy mačiek ktorým slúži.

Viacero hostiteľov môže slúžiť jednej mačke a hostiteľ môže slúžiť viacerým mačkám. Keďže má mačka 9 životov, je nutné si pamätať aj hostiteľovo meno pre mačku a aj interval kedy mačke slúžil a volal ju daným menom.

Mačky vlastnia viacero vecí (vlastníctiev), ale jedno vlastníctvo patrí iba jednej mačke v daný čas. Mačka sa veci ale môže vzdať a neskôr si vec môže iná mačka privlastniť. Je nutné si pamätať aj kedy vlastníctvo mačka vlastnila. Jedno vlastníctvo sa nachádza v jednom teritóriu a v jednom teritóriu môže byť viacero vlastníctiev. Vlastníctva sa dajú prepožičať hostiteľovi.

Schéma databázy



Obr. 1

Generalizácia/Špecializácia

V rámci entitnej množiny *Život* bolo potrebné zaznamenať, či je konkrétny život stále prebiehajúci, alebo už bol ukončený. Okrem tejto informácie zároveň existuje vzťah tejto množiny s množinou *Teritórium,* kde sa zaznamená, kde bol život začatý, a kde skončil. Pre zjednodušenie týchto vzťahov bola zvolená implementácia generalizácie/špecializácie práve pri entitnej množine *Život,* a to takým spôsobom, že máme iba jednu tabuľku a informáciu o začatom/aktuálnom živote máme v stĺpci *isDead*.

Implementácia

V samotnej implementácii je v skripte prvým krokom vytvorenie tabuliek podľa dátového modelu. Tieto sú následne naplnené dátami, aby sa mohli neskôr testovať jednotlivé dotazy. Ako prvé sú SELECT dotazy, ktoré sú zakomentované (projekt 3), následne sa vykoná optimalizácia a jej zobrazenie pomocou EXPLAIN PLAN spolu s INDEXOM ako optimalizáciou. V ďalšom kroku sú vytvorené obmedzenia cez TRIGGER dotazy, ktoré sú hneď aj otestované. Toto testovanie prebieha tak, že sa do tabuľky vkladajú dáta ktoré nespĺňajú vstupné požiadavky požadované v rámci jednotlivých triggerov, pričom toto vkladanie vyvolá chybu (tieto INSERTy sú v kóde zakomentovnané). Ďalej v skripte nasledujú definície procedúry, ktoré sa potom aj spustia. A ako posledné sú pridelené práva a vytváranie materializovaného pohľadu.

Triggery

V rámci projektu sme implementovali 4 databázové triggery. *If_Cat_PK_NULL* a *If_Race_PK_NULL* vyplývajú zo zadania a slúžia iba na automatické generovanie primárneho kľúča. Ako už z názvu vyplýva, pracujú s tabuľkami *Cat* a *Race* a správnosť triggerov sa overila INSERT príkazmi bez hodnoty PK.

Trigger *Proper_life_dates*, bol už komplikovanejší a jeho hlavnou úlohou bolo kontrolovať správnosť dátumov v tabuľke *Life*. Keďže život súvisí s mačkou, najskôr sme si našli najvyšší dátum u danej mačky a potom sme skontrolovali, či nový dátum narodenia sa nachádza neskôr ako tento dátum. Pokiaľ sa tak nestane, vyvoláme chybu. Okrem tejto funkcionality sme sa rozhodli implementovať ešte funkciu *Instantnej reinkarnácie.* Tá spočíva vtom, že keď chceme pridať nový život a predošlý život ešte neskončil, tak predošlý život ukončíme s dátumom smrti rovnakým ako dátum narodenia nového života. Takisto sa nastaví položka *isDead* predošlého života na 'Y' (yes). Aby sa trigger lepšie testoval, pridal sa trigger *If_Life_Attributes_NULL*, ktorý okrem automatického generovania primarného kľuča, automaticky vypočíta aj *lifeOrder* a *isDead*. Pomocou INSERT a SELECT dotazov sme sa snažili demonštrovať funkcionalitu týchto triggerov.

V našom skripte robíme: pridanie nového života so všetkými atribútmi správnymi, pridanie života, ktorý má dátum smrti skôr ako dátum narodenia (pokryté už pri vytváraní tabuľky), pridanie života, ktorý má dátum narodenia skôr ako predchádzajúci život začal (posledné dva zmienené vyvolajú chyby a sú v skripte zakomentované) a 2 INSERTy ktoré majú iba záznamy o dátume narodenia, mačke a mieste narodenia, na ktorých demonštrujeme instantnú reinkarnáciu, rovnako ako *If_Life_Attributes_NULL* trigger.

Explain Plan s Indexom

Príkaz Explain Plan pomohol demonštrovať optimalizáciu spracovania dotazu po pridaní indexu do tabuľky. Dotaz nad ktorým sa optimalizácia vykonávala vyhľadá a spočíta všetky životy, minulé aj aktuálne, prislúchajúce danej mačke.

V prvom zavolaní sa index v tabuľke nenachádzal a výsledok spracovania tohto dotazu je možné vidieť v prvej tabuľke. Pred druhým zavolaním bol do tabuľky zavedený index do tabuľky *Life* na stĺpec *catFK.* Index bol zavedený na toto miesto vzhľadom na to že daný stĺpec obsahuje hodnotu cudzieho kľúča.

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		4	312	7 (15)	00:00:01
1	HASH GROUP BY		4	312	7 (15)	00:00:01
* 2	HASH JOIN		4	312	6 (0)	00:00:01
3	TABLE ACCESS FULL	LIFE	4	52	3 (0)	00:00:01
4	TABLE ACCESS FULL	CAT	5	325	3 (0)	00:00:01

Tabuľka 1

Pri spracovaní dotazu SELECT, sa ako prvá operácia vykoná HASH GROUP BY. Táto operácia zoskupuje položky podľa hashovacieho kľúča. Následne operácia HASH JOIN páruje záznamy spojených tabuliek, využíva hashovací kľúč spojenia. Následne sa operáciou TABLE ACCESS FULL prečítajú celé tabuľky *Life* a *Cat.*

Po zavedení indexu začína vyhodnocovanie podobne, následne pomocou operácie NESTED LOOPS sa prejde každý riadok tabuľky. Na koniec miesto tabuľky *Life,* je operácia INDEX RANGE SCAN, ktoré vzostupne pristupuje k indexovaným záznamom, zároveň využíva vytvorený index v danej tabuľke a neprechádza jej iné stĺpce.

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time
0	SELECT STATEMENT		4	312	4 (25)	00:00:01
1	HASH GROUP BY		4	312	4 (25)	00:00:01
2	NESTED LOOPS		4	312	3 (0)	00:00:01
3	TABLE ACCESS FULL	CAT	5	325	3 (0)	00:00:01
* 4	INDEX RANGE SCAN	MY_INDEX	1	13	0 (0)	00:00:01

Tabuľka 2

Vidíme, že pridanie indexu do tabuľky *Life* znížilo cenu jednotlivých operácií a mierne aj pamäťovú náročnosť, no na druhej strane stúpol procesorový čas.

Procedúry

V skripte sú implementované dve netriviálne procedúry, ktoré používajú implementovaný kurzor, *cat_details()* a *teritory_info()*. Obidve procedúry taktiež využívajú premenné odkazujúce na riadok tabuľky a starajú sa o krajné prípady ktoré by mohli za behu spôsobiť nechcený pád programu.

Prvá procedúra, *cat_details()*, má za úlohu vypísať všetky príslušné dáta patriace mačke. Tento výpis spraví pre všetky položky z tabuľky *Cat.* Začne prechádzať túto tabuľku po jednotlivých riadkoch. Pre každú položku vypíše meno, pôvod a aktuálne prebiehajúci život. Ten získa pomocou dotazu SELECT, ktorý hľadá život s rovnakým cudzím kľúčom ako je primárny kľúč mačky. Následne pre každú mačku prejde po položkách tabuľku vlastníctiev a vypíše tie, ktoré prislúchajú danej mačke. Ako posledné sa vypíšu majitelia ktorí tejto mačke slúžia.

Procedúra využíva dva kurzory, jeden na tabuľke *Cat* a druhý na tabuľke *CatOwns,* ktorá má uložené vlastníctva pre danú mačku. Osobitne sa stará o prípad, keď mačka nezačala ešte žiť žiaden život.

Procedúra *teritory_info()* má podobnú funkciu, a to vypísať pre každé teritórium, koľko mačiek sa v ňom narodilo, a tiež koľko vlastníctiev sa na jeho území nachádza. Procedúra využíva kurzor na tabuľke *Teritory* a *Ownership.*

Prechádza po jednom jednotlivé teritóriá z tabuľky, pre každé spočíta životy ktoré v ňom začali a tento počet vypíše. Taktiež vypíše, pokiaľ žiadne také neexistujú. Pre toto teritórium taktiež prejde každý riadok vlastníctiev, kde hľadá také, ktoré má cudzí kľúč zhodný s primárnym kľúčom teritória. Vlastníctva ktoré sa v ňom nachádzajú po jednom vypíše.

Materializovaný pohľad & Pridelenie prístupových práv

Pre vypracovanie úlohy s materializovaným pohľadom, bolo nutné najskôr prideliť prístupové práva druhému členovi tímu. Druhý člen dostal práva na všetky tabuľky a spúšťanie procedúr a demonštrujeme to na tabuľkách *Cat a Race.* Vytvárame materializovaný pohľad pre všetky mačky z Austrálie (*Australians*) z tabuliek prvého člena tímu. Na SELECT dotaze sme otestovali, či pohľad funguje a pomocou INSERT dotazu sme si overili, že sa pohľad aktualizuje po prevedení zmeny v tabuľke.

Záver

Skript sme vypracovávali v prostredí Visual Studio Code, na školskom serveri Oracle na hostovi gort.fit.vutbr.cz. Projekt sme vypracovávali s pomocou informácií z prednášok, slidov a demonštračných cvičení. Taktiež sme čerpali z Oracle dokumentácie a súvisiacich článkov na internete.

Práca na projekte nás veľmi bavila, vyskúšali sme si prácu s rôznymi (aj komplikovanejšími) databázovými objektmi a sme si istí, že poznatky získané v tomto projekte budeme využívať aj v kariérnom živote.