# 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**

interval

**Žije**

**Rasa**

<PK> ID

farba očí

pôvod

max dĺžka tesákov

špecifické rysy

**Mačka**

<PK> Hlavné meno

vzorka kože

farba srsti

farba očí

**Život**

<PK> ID

poradové číslo

**Aktuálny**

dátum narodenia

**Minulý**

spôsob smrti

dátum smrti

dátum narodenia

**Hostiteľ**

<PK> ID

meno

dátum narodenia

pohlavie

bydlisko

**Vlastníctvo**

<PK> ID

typ

kvantita

**Teritórium**

<PK> ID

typ

kapacita

0..n

je

1

1

má

1..9

1

začal

1

0..n

skončil

1

1

je

0..n

0..n

preferuje

0..n

interval

**Slúži**

meno

0..n

0..n

**Vlastní**

interval

0..1

0..n

1

0..n

prepožičať



0..n

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.

**UCD**

Odobrať život / Narodiť sa

Čas

Zaslať novinky

Výpis vlastníctva a sluhov

Presídliť

Prijať hostiteľa

Nadobudnúť vlastníctvo

Prepožičať vlastníctvo

Mačka

**Generalizácia/Špecializácia**

V rámci entitnej množiny *Život* bolo potrebné zaznamenať, či je konkétny 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 sú vytvorené dve osobitné tabuľky na ňu naviazané. Jedna ukladá minulé životy a druhá aktuálne prebiehajúce. Obe tabuľky sú ďalej naviazané na entitnú množinu *Teritórium,* kde existujúci vzťah ukazuje kde ten daný život začal/skončil. Tieto tabuľky zdieľajú primárny kľúč ID.

**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 sa vykonajú SELECT dotazy, 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. Ako posledné sú v skripte definované procedúry, ktoré sa potom aj spustia.

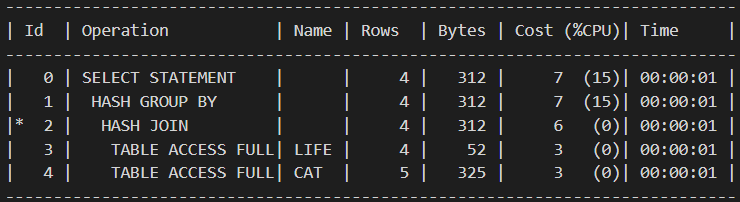
**Triggery**



**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.

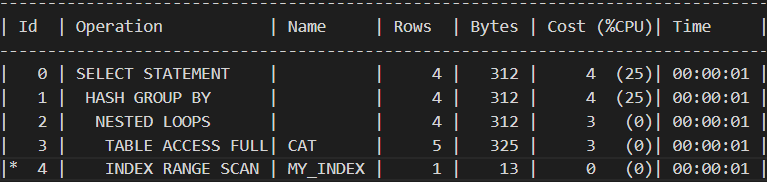
V prvom zavolaní sa index v tabuľke nenachádzal a výsledok spracovania tohto dotazu je možné vidieť v prvej tabuľke. Pred druhým zavolaním bol do tabuľky zavedený index do tabuľky *Life* na stĺpec *catFK.* Index bol zavedený na toto miesto vzhľadom na to že daný stĺpec obsahuje hodnotu cudzieho kľúča.



Tabuľka 1

Pri spracovaní dotazu SELECT, sa ako prvá operácia vykona HASH GROUP BY. Táto operácia zoskupuje položky podľa hashovacieho kľúča. Následne operácia HASH JOIN spáruje 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.



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 posledný výpis nájde majiteľov 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**

**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.