FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Dokumentácia projektu č. 48 do predmetu *Databázové systémy*

Kočičí Informační Systém

Jakub MÍkvy (xmlkvy00)
Adam Múdry (xmudry01)

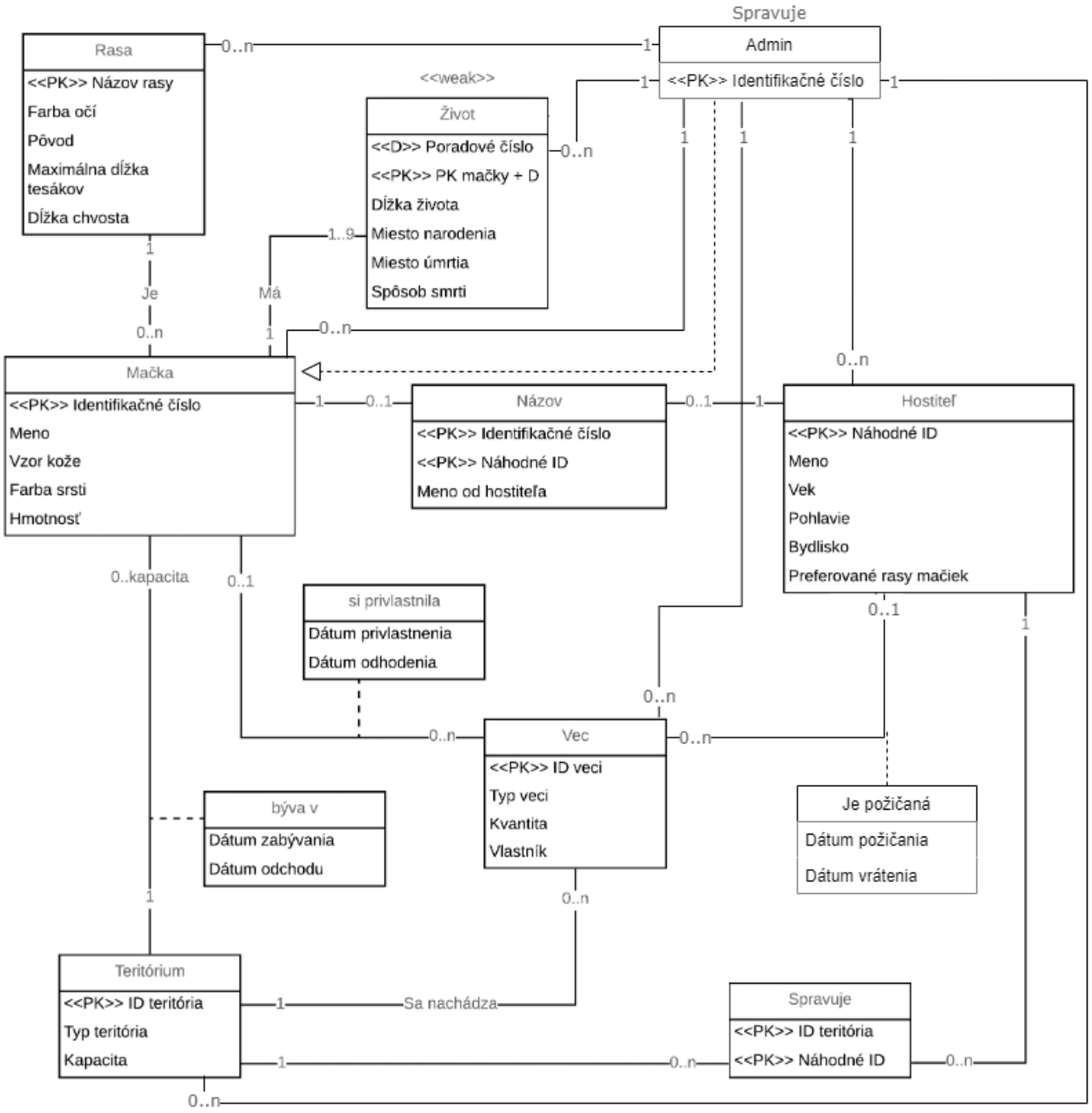
30. apríla 2020

Obsah

1 Zadanie

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.

2 Návrh


Spravuje
Rasa
<<PK>> Názov rasy
Farba očí
Pôvod
Maximálna dĺžka tesákov
Dĺžka chvosta
Admin
<<PK>> Identifikačné číslo
<<weak>>
Život
<<D>> Poradové číslo
<<PK>> PK mačky + D
Dĺžka života
Miesto narodenia
Miesto úmrtia
Spôsob smrti
Je
Má
Mačka
<<PK>> Identifikačné číslo
Meno
Vzor kože
Farba srsti
Hmotnosť
Názov
<<PK>> Identifikačné číslo
<<PK>> Náhodné ID
Meno od hostiteľa
Hostiteľ
<<PK>> Náhodné ID
Meno
Vek
Pohlavie
Bydlisko
Preferované rasy mačiek
si privlastnila
Dátum privlastnenia
Dátum odhodenia
Vec
<<PK>> ID veci
Typ veci
Kvantita
Vlastník
býva v
Dátum zabývania
Dátum odchodu
Je požičaná
Dátum požičania
Dátum vrátenia
Teritórium
<<PK>> ID teritória
Typ teritória
Kapacita
Sa nachádza
Spravuje
<<PK>> ID teritória
<<PK>> Náhodné ID

3 Triggery

Implementované sú 2 typy triggerov. Trigger na generovanie ID nebol potreba, lebo to máme implementované pomocou sekcvencie príkazov

```
GENERATED BY DEFAULT AS IDENTITY (START WITH 1 INCREMENT BY 1) NOT NULL PRIMARY KEY
```

Prvý trigger (*trigger_max_zivotov*) pri vložení alebo aktualizácii dát v tabuľke *Zivot* kontroluje korektné číslo života (1 až 9) a ďalej kontroluje jeho unikátnosť pomocou našej procedúry *zivot_unique_check()*.

Ďašie triggery pri vložení alebo aktualizácii dát do tabuliek *Byva*, *JePozicane*, *JePrivlastnene* kontrolujú správnosť dátumov uložených do dátového typu *DATE* – musí byť zadaný dátum *zabývania / požičania / privlastnenia* a dátum *odchodu / vrátenia / odhodenia* nemôže byť starší ako prvý dátum.

Pri chybe sa vyhadzuje chybová hláška s príslušným kódom a správou pomocou *Raise_Application_Error()*.

4 Procedúry

Implementovali sme procedúry

```
pocet_mackiek_v_teritoriu(typ_teritoria IN VARCHAR2)
zivot_unique_check(macka_id IN int, zivot_no IN int)
```

Prvá procedúra berie ako argument názov teritória – hľadá všetky mačky, ktoré sa v teritóriu momentálne nachádzajú a na *DBMS_OUTPUT* vetou vypíše ich počet.

Druhá procedúra je automaticky volaná triggerom *trigger_max_zivotov* a jej úloha je skontrolovať unikátnosť života mačky – ak život s číslom *zivot_no* pre mačku s id *macka_id* existuje v databáze, procedúra vyhodí *application error -20011*. Ak je procedúra úspešná, na *DBMS_OUTPUT* vypíše výstup oznamujúci správnosť operácie.

5 Prístupové práva

Práva, ktoré sme pridelili druhému členu tímu, by mohli predstavovať *moderátora systému*, ktorý môže upravovať všetky tabuľky okrem tabuľky *Admin*.

Ďalšie možné priradenie práv by mohlo byť napr. pre *supporta / helpera*, ktorý by mohol vidieť iba určité informácie – napr. všetky okrem citlivých informácii.

6 Explain Plan

Dotaz pre Explain Plan

```sql
SELECT M.meno, max(Z.cislo_zivota)
FROM Zivot Z, Macka M
WHERE Z.Macka_id = M.id
GROUP BY Z.Macka_id, M.meno;
```

Plán bez použitia indexu

```
-------------------------------------------------------------------------------------------
| Id  | Operation                     | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                |     1 |    57 |     5  (20)| 00:00:01 |
|   1 |  NESTED LOOPS                 |                |     1 |    57 |     5  (20)| 00:00:01 |
|   2 |   NESTED LOOPS                |                |     1 |    57 |     5  (20)| 00:00:01 |
|   3 |    VIEW                       | VW_GBC_5       |     1 |    26 |     4  (25)| 00:00:01 |
|   4 |     HASH GROUP BY             |                |     1 |    26 |     4  (25)| 00:00:01 |
|   5 |      TABLE ACCESS FULL        | ZIVOT          |     1 |    26 |     3   (0)| 00:00:01 |
|*  6 |    INDEX UNIQUE SCAN          | SYS_C001519870 |     1 |       |     0   (0)| 00:00:01 |
|   7 |   TABLE ACCESS BY INDEX ROWID| MACKA          |     1 |    31 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

Plán s použitím indexu

```
-------------------------------------------------------------------------------------------
| Id  | Operation                     | Name           | Rows  | Bytes | Cost (%CPU)| Time     |
-------------------------------------------------------------------------------------------
|   0 | SELECT STATEMENT              |                |     1 |    57 |     2   (0)| 00:00:01 |
|   1 |  NESTED LOOPS                 |                |     1 |    57 |     2   (0)| 00:00:01 |
|   2 |   NESTED LOOPS                |                |     1 |    57 |     2   (0)| 00:00:01 |
|   3 |    VIEW                       | VW_GBC_5       |     1 |    26 |     1   (0)| 00:00:01 |
|   4 |     HASH GROUP BY             |                |     1 |    26 |     1   (0)| 00:00:01 |
|   5 |      INDEX FULL SCAN          | INDEX_EXPLAIN  |     1 |    26 |     1   (0)| 00:00:01 |
|*  6 |    INDEX UNIQUE SCAN          | SYS_C001519870 |     1 |       |     0   (0)| 00:00:01 |
|   7 |   TABLE ACCESS BY INDEX ROWID| MACKA          |     1 |    31 |     1   (0)| 00:00:01 |
-------------------------------------------------------------------------------------------
```

Explain Plan nám ukazje ako konkrétny dotaz spracováva naša databáza. Dotaz vypíše meno mačky a číslo najnovšieho zaznamenaného života danej mačky.

SELECT STATEMENT – uskutočnenie daného dotazu
NESTED LOOPS – 2-krát loop na pre porovnanie položiek jednej tabuľky s druhou
VIEW – operácia na zobrazenie dotazu
HASH GROUP BY – spojenie položiek podľa hashovacieho kľúča
TABLE ACCESS FULL – zoberie všetky riadky tabuľky
INDEX FULL SCAN – indexácia riadkov
INDEX UNIQUE SCAN – získanie jedinečného riadku podľa primárneho kľúča
TABLE ACCESS BY INDEX ROWID – prístup po riadkoch

Použitím indexu sme znížili záťaž operácií daného dotazu na procesor.

7 Materializovaný pohľad

Pri používaní materializovaného pohľadu redukujeme zaťaženie databázy – používame dáta uložené na disku. Nemusí však vždy obsahovať najaktuálnejšie dáta.

Náš materializovaný pohľad zobrazuje názvy všetkých farieb srsti, aké mačky v databáze majú a počet mačiek s danou farbou srsti. Zmeny v pohľade sa aktualizujú po príkaze *COMMIT*.

Použité optimalizácie:

CACHE	– optimalizácia načítavania už zobrazených dát
BUILD IMMEDIATE	– vytvorenie pohľadu hneď po naplnení
REFRESH FAST ON COMMIT	– optimalizácia rýchleho obnovenia pri commite
ENABLE QUERY REWRITE	– optimalizácia rýchlej aktualizácie uloženého pohľadu

8 Záver

SQL skript sme písali spoločne v textovom editore *VS Code* pomocou rozšírenia *Live Share* a následne sme ho testovali na školskom *Oracle serveri*, na ktorý sme sa pripojili pomocou programu *Jetbrains DataGrip*. Komunikácia prebehla pomocou programu *Discord*, ktorý umožňuje okrem audio komunikácie aj zdieľanie obrazovky, čo nám veľmi pomohlo. Ako zdroj informácií poslúžil najmä *StackOverflow* a oficiálna dokumentácia *Oracle SQL*.