Slovenská Technická Univerzita v Bratislave
Fakulta Informatiky a Informačných Technológií
Ilkovičova 2, 842 16 Bratislava 4

Chovateľské centrum

Filip Gális

Študijný program: Informatika
Ročník: 3
Krúžok: Štvrtok, 13:00
Predmet: Databazove systemy
Vedúci projektu: Ing. Ondrej Kachman
Ak. rok: 2015/16

1. Zadanie

Vo vami zvolenom prostredí vytvorte databázovú aplikáciu, **ktorá komplexne rieši minimálne 6 scenárov** vo vami zvolenej doméne. Presný rozsah a konkretizáciu scenárov si dohodnete s Vašim cvičiacim na cvičení. Aplikáciu vytvoríte v dvoch iteráciach. V prvej iterácii, postavenej nad relačnou databázou, musí aplikácia realizovať tieto všeobecné scenáre:

- Vytvorenie nového záznamu,
- Aktualizácia existujúceho záznamu,
- Vymazanie záznamu,
- Zobrazenie prehľadu viacerých záznamov (spolu vybranou základnou štatistikou),
- Zobrazenie konkrétneho záznamu,
- Filtrovanie záznamov spĺňajúcich určité kritériá zadané používateľom.

Aplikácia môže mať konzolové alebo grafické rozhranie. Je dôležité aby scenáre boli realizované realisticky - teda aby aplikácia (a teda aj jej používateľské rozhranie) naozaj poskytovala časť funkcionality tak, ako by ju očakával zákazník v danej doméne.

Scenáre, ktoré menia dáta musia byť realizované **s použitím transakcií** a aspoň jeden z nich musí zahŕňať **prácu s viacerými tabuľkami** (typicky vytvorenie záznamu a naviazanie cudzieho kľúča).

V druhej iterácii do aplikácie pridáte min. 1 scenár postavený na nerelačnej databáze Redis alebo Elasticsearch (dohoda s cvičiacim na inom type nerelačnej db je samozrejme možná). Konkrétny scenár si dohodnete s vašim cvičiacim v závislosti od použitej databázy a domény vašej aplikácie (napr. štatistiky o interakciách s jednotlivými záznamami aplikácie v Redise alebo vyhľadavávanie záznamov cez Elasticsearch).

Bez odovzdanej (teda cvičiacim akceptovanej) prvej iterácie nie je možné odovzdať druhú.

Pre získanie zápočtu je potrebné odovzdať (a cvičiaci musí akceptovať minimálnu úroveň kvality) obidve iterácie projektu.

2. Špecifikácia scenárov

2.1. Vypisanie zvierat, ktore boli objednane zakaznikom za posledne 4 mesiace a meno a kontakt daneho zakaznika

V tomto selecte vyberam z tabuľky zviera, ktorú spájam s tabuľkou objednávka a osoba všetky zvierata, ktore boli objednané zákazníkom za posledne 4 mesiace. Taktiež vypisujem meno a priezvisko zakaznika a kontakt na daného zakaznika.

2.2. Vypisanie mena zvierata a datum kontroly kedy bolo u zverolekara a meno lekara

V tomto selecte vyberam taktiež z tabuľky zviera, ktorú ale spájam s tabuľkou kontrola zvieraťa a osoba aby som vytlačil meno, rasu a plemeno všetkých zvierat, ktoré boli skontrolované zverolakárom z tabuľky kontrola zvieraťa, ako aj meno daného zverolekára.

2.3. Meno, adresa, kontakt a pocet zakaznikov, ktori si objednali psa starsieho ako rok

V tomto selecte pracujem s hlavnou tabuľkou osoba, ktorú spájam s tabuľkou objednávka a zviera. Cez tabuľku osoba si vyberám meno, priezvisko a kontakt na zakazníkov ako aj celkový počet zákazníkov, ktorí si objednali psa staršieho ako jeden rok.

2.4. Pracovnik, ktory ako posledny nakupil krmivo pre zvierata, datum objenavky a suma hmotnosti krmiva

V tomto selecte pracujem s hlavnou tabuľkou osoba, ktorú spájam s tabuľkami objednávka a krmivo. Z tabuľky si počítam najväčší dátum, v ktorom pracovník stanice kupoval krmivo ako aj sumu hmotnosti krmiva ktoré kúpil.

2.5. Pocet objednavok psov manazerom chovatelskej stanice a meno manazera

V tomto selecte pracujem s tabuľkou osoba, z ktorej vyberám konkrétny záznam a to manažérov alebo manažerá, ktorý nakúpil psov a počet týchto objednávok ako aj meno a priezvisko manažéra.

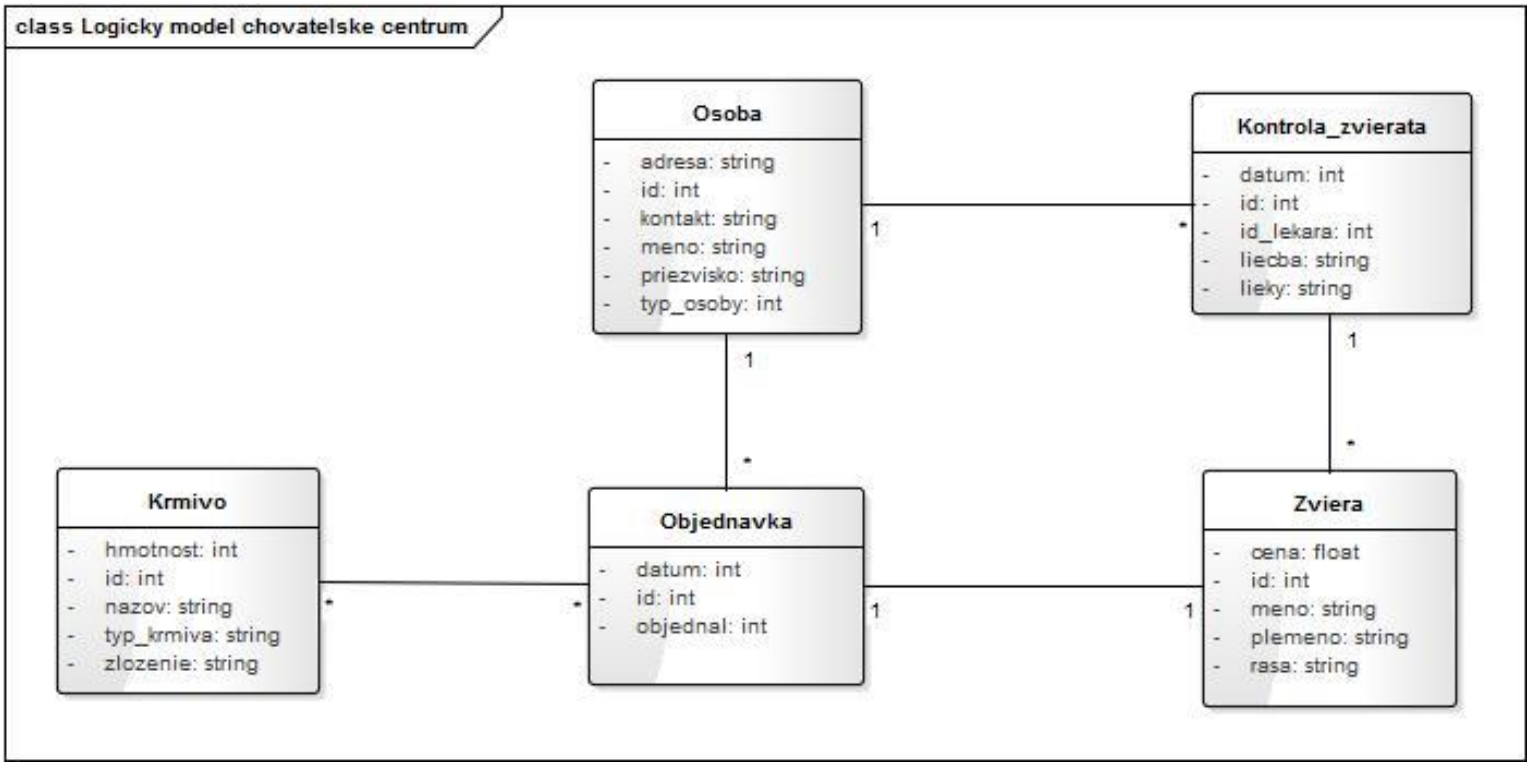
2.6. Objednavky zvierat, ktore vhodne pre rodinu a ich priemerna cena

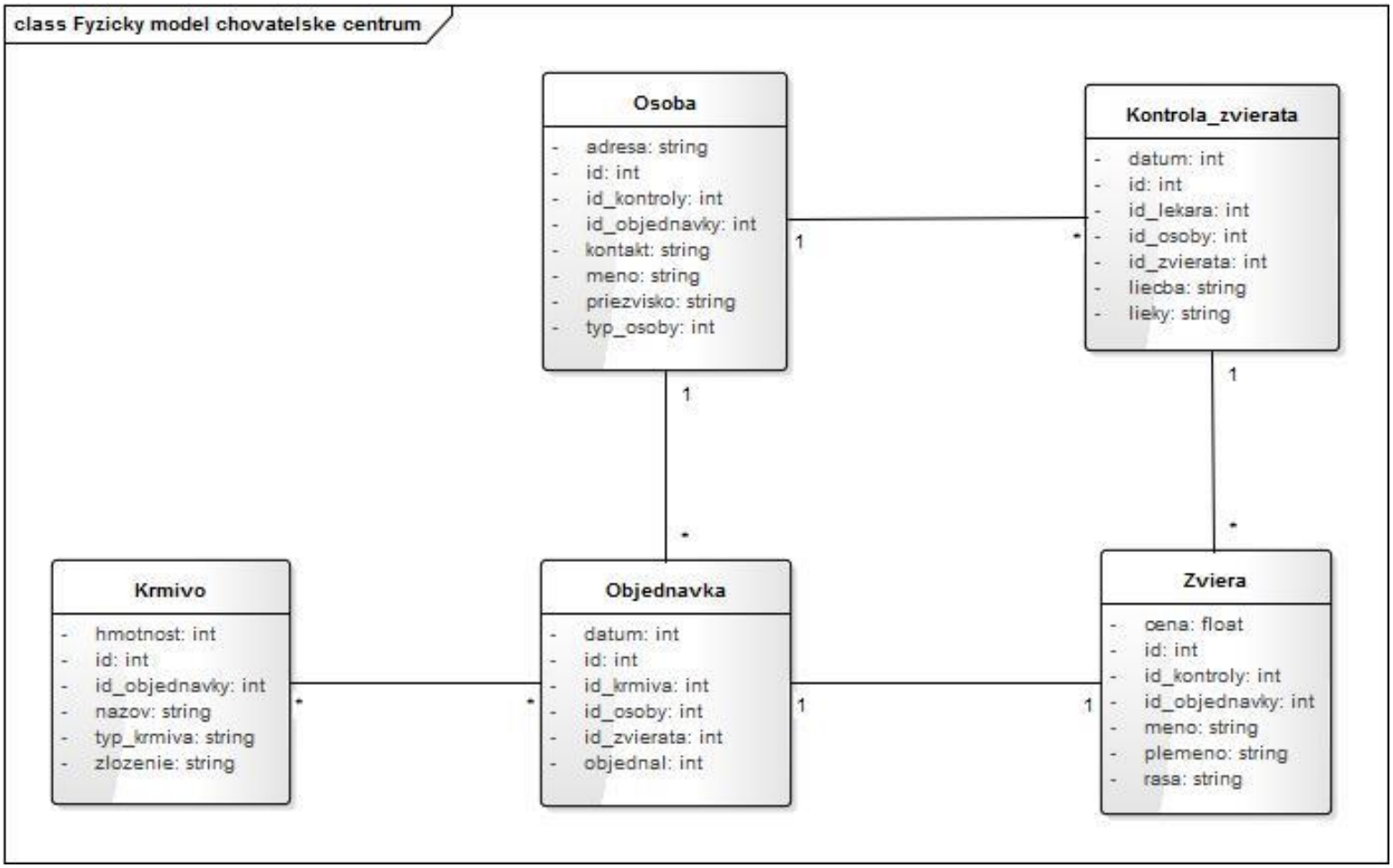
V tomto selecte pracujem hlavne s tabuľkou objednávka a používam len jedno pripojenie a to s tabuľkou zviera. Z nej vyberám id objednávky meno a rasu zvierata a priemernu cenu zvierat ktoré su vhodné pre rodinu podľa atributu v tabuľke zviera.

2.7. Objednavky krmiva, ktore su pre macky a ich hmotnosti je vacsia ako priemerna

V tomto selecte pracujem hlavne s tabuľkou objednávka a používam len jedno pripojenie a to s tabuľkou krmivo. Z nej si vyberam id objednávky, nazov a hmotnosť krmiva. Následne to filtrujem tak, že vypíšem len krmivo čo váži viac ako priemer

3. Diagram logického a fyzického dátového modelu





4. Stručný opis návrhu a implementácie

4.1. Programovacie prostredie

Ako programovanie prostredie som použil jazyk JAVA, konkrétne JAVA 8 a vývojové prostredie eclipse. Ná prácu s databázou používam SQL databázu PostgreSQL a jeho grafické serverové prostredie pgAdmin III.

4.2. Návrhové rozhodnutia

Do PostgreSQL databázy pristupujem cez špeciálny java input java.sql a jeho mnohé podinputy, ktoré mi umožňujú jednoducho pristupovať do databázy týmto kúskom kódu:

```
private Connection connection = null;

private Statement statement = null;

Class.forName("org.postgresql.Driver");

connection = DriverManager.getConnection("jdbc:postgresql://localhost:5432/postgres", "postgres", "databazy2016");

connection.setAutoCommit(false);
```

Najprv si zadeklarujem objekty connection a statement na začiatku triedy a potom s nimi pracujem nasledovne:

- Vytvorím si statement

```
statement = connection.createStatement();
```

- Vložím doňho reťezec z sql príkazom a vyvolam prikaz na jeho vykonanie

```
statement.executeUpdate(sql);
```

- Nakoniec už len zatvorim pripojenie a ukončím statementy

```
statement.close();

connection.commit();

connection.close();
```

4.3. Opis implementácie jednotlivých scenárov

4.3.1 Vypisanie zvierat, ktore boli objednane zakaznikom za posledne 4 mesiace a meno a kontakt daneho zakaznika

Joinujem si tabulky zviera, objednavka a osoba a nasledne si dávam podmienku aby dátum bol od zaciatku roka po april.

```
select p.meno, p.kontakt, o.datum, z.meno, z.plemeno from "
                                                    + "zviera as z join
objednavka as o on z.id=o.id_zvierata join osoba as p "
                                                    +  "on  o.id=p.id_objednavky
where o.datum>'2016-01-01' and o.datum<'2016-04-30'
```

4.3.2 Vypisanie mena zvierata a datum kontroly kedy bolo u zverolekara a meno lekara

Joinujem si tabuľky zviera kontrola zvieraťa a osoba a porovnávam si či typ osoby je 2, čo znamená, že je to zverolakár.

```
select z.meno, p.meno, p.priezvisko, k.datum "
                                                    + "from zviera as z join
kontrola_zvierata as k on z.id=k.id_zvierata join "
                                                    + "osoba as p on
k.id=p.id_kontroly where p.typ_osoby=2 "
                                                    + "group by p.meno, z.meno,
p.priezvisko, k.datum
```

4.3.3 Meno, adresa, kontakt a pocet zakaznikov, ktori si objednali psa starsieho ako rok

Tu idem od tabuľky osoba a joinujem si ju s tabuľkami objednávka a zviera. Potom si porovnávam či zviera je pes, či ma viac ako 1 rok a či osoba má typ 3, to znamená, že je zákazník. Taktiež používam group by meno, priezvisko

```
select p.meno, p.adresa, p.kontakt, count(p.id) "
                                                    + "from osoba as p join
objednavka as o on p.id=o.id_osoby join zviera as z "
                                                    + "on o.id=z.id_objednavky
where z.rasa='pes' and z.vek>1 and p.typ_osoby=3 "
                                                    + "group by p.meno, p.adresa,
p.kontakt
```

4.3.4 Pracovnik, ktory ako posledny nakupil krmivo pre zvierata, datum objenavky a suma hmotnosti krmiva

Taktiež pracujem s tabuľkou osoba ale joinujem s objadnavka a krmivo. Tu už neporovnávam ale si vypisujem najvačší dátum objednávky a cez max(datum) a limitujem na 1

```
select p.meno, p.priezvisko, max(o.datum), "
                                                    + "sum(k.hmotnost) from
osoba as p join objednavka as o on p.id=o.id_osoby "
                                                    + "join krmivo as k on
o.id=k.id_objednavky group by p.meno, p.priezvisko "
```

```
                                                + "order by max(o.datum) desc
limit 1
```

4.3.5 Pocet objednavok psov manazerom chovatelskej stanice a meno manazera

Podobné prvemu selectu v tabuľke osoba, ale kontrolujem či osoba objednávajúca zvierata je manžér, či zviera je pes a sumu týchto objednávok.

```
select sum(o.id), p.meno, p.priezvisko from osoba as p "
                                                + "join objednavka as o on
p.id=o.id_osoby join zviera as z on o.id=z.id_objednavky "
                                                + "where p.pozicia='manazer'
and z.rasa='pes' group by p.meno, p.priezvisko
```

4.3.6 Objednavky zvierat, ktore vhodne pre rodinu a ich priemerna cena

Tu už vychádzam z tabuľky objednávka a joinujem len s tabuľkou zviera. V nej si porovnávam podľa atributu rodinne či je zviera vhodné pre rodinu a taktiež počítam priemernú cenu týchto zvierat.

```
select o.id, z.meno, z.rasa, avg(z.cena) from "
                                                + "objednavka as o join
zviera as z on o.id=z.id_objednavky where rodinne='true' group by o.id,
z.meno, z.rasa
```

4.3.7 Objednavky krmiva, ktore su pre macky a ich hmotnosti je vacsia ako priemerna

Tu si naopak joinujem tabuľku objednávka s tabuľkou krmivo a volám si druhý select ktorým si zistim priemerný hodnotu a vypíšem len tie krmiva ktoré sú pre mačky a vážia viac ako priemer.

```
elect o.id, k.nazov, k.hmotnost from objednavka as o "
                                                + "join krmivo as k on
o.id=k.id_objednavky where k.hmotnost >= (select avg(k.hmotnost) "
                                                + "from krmivo as k) group by
o.id, k.nazov, k.hmotnost order by k.hmotnost desc
```

5. Použitie a implementovanie NoSQL databázy

5.1. Použitá databáza

Na prácu z NoSQL databázou som použil nástroj Elasticsearch verziu 2.3.2. Vývojové prostrenie NetBeans, programovací jazyk Java a server Elasticsearch.

5.2. Implementácia databázy

Napojenie do Elasticsearch databázy riešim cez príkazový riadok tak, že do priečinku elasticsearch cestou elasticsearch-2.3.2/bin a v ňom si spustim príkaz elastisearch. Tento príkaz mi spustí databázu na andrese „localhost“ a porte 9200 ako aj transport na porte 9300. Následne sa cez javu napájam na databázu tak, že som si vytvoril dependency na elasticsearch

```
<dependency>
  <groupId>org.elasticsearch</groupId>
  <artifactId>elasticsearch</artifactId>
  <version>2.3.2</version>
</dependency>
```

a potom som si už len vytvoril transport clienta na porte 9300, ktorým sa pripájam do databázy a vykonávam potrebné akcie.

Client client = TransportClient.builder().build().addTransportAddress (new InetSocket TransportAddress (InetAddress.getByName("localhost"), 9300));

5.3. Opis implementacie scenára - Zoznam krmiv s nazvom pedigri s hmotnostou od 20kg do 80kg a ich priemerna hmotnost

V tomto scenári si vytváram SearchResponse (niečo ako resultset v postgrese) na index „postgres“ (názov databázy). Následne si k SearchResponse pripájam nastavovanie typu na „krmivo“ (názov tabuľky, z ktorej beriem data), samotné query (vyberám len tie záznamy, v ktorých je nazov krmiva pedigri), agregaciu priemernej hodnoty hmotnosti krmiv a rozsah hmotnosti krmiva, ktoré sa budú vyberať.

```
SearchResponse response = client.prepareSearch("postgres")
        .setTypes("krmivo").setSearchType(SearchType.DFS_QUERY_THEN_FETCH)
        .setQuery(QueryBuilders.termQuery("nazov", "pedigri"))                    // Query
        .addAggregation(AggregationBuilders.avg("priem_hmotnost").field("hmotnost"))
        .setPostFilter(QueryBuilders.rangeQuery("hmotnost").from(20).to(80))      // Filter
        .setFrom(0).setSize(60).setExplain(true).execute().actionGet();
```

Záver

Na záver by som len povedal, že pracovať s relačnou postgresql databázou a nerelačnou elasticsearch databázou bolo pre mňa záujimavou skúsenosťou a rozhodne si viem predstaviť pracovať s nimi aj v budúcnosti.