

Asoft okvir za klasifikacije



# Sadržaj

- Opis rešenja
- Model podataka
- Java biblioteka
- Mogućnosti primene
- Planovi za dalji razvoj
- Pitanja

# Opis rešenja

**A-Klas** je okvir za univerzalnu i dinamičku klasifikaciju zapisa u bazi podataka

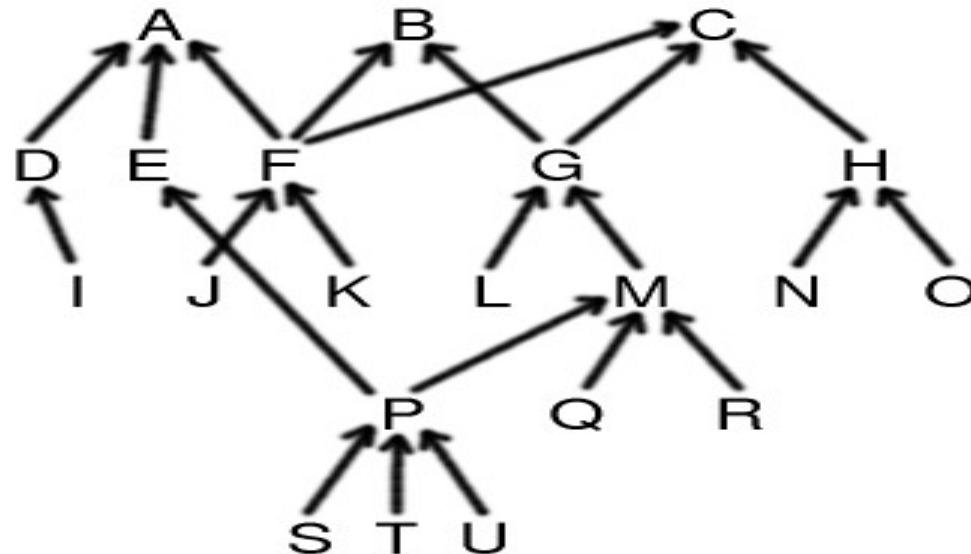
**A-Klas** omogućava tipiziranje, grupisanje, razvrstavanje, kategorizaciju i formiranje ostalih klasifikacija objekata kao što su:

`vrsta_dokumenta, tip_računa, namena_konta,  
grupa_konta, tip_kursa, vrsta_lokacije,  
kategorija_partnera, vrsta_artikla`

• • •

# Opis rešenja

**A-Klas** omogućava uspostavljanje vertikalnih i horizontalnih asocijacija između objekata i na taj način formira multi-tree strukturu



# O čemu je reč

Ideja potiče sa MIT-a, a reč je o novom hitu koji se zove

<Tagged data structures>

To su strukture podataka kojima su dodeljeni ljudski atributi, tj tagovi.

Sa ovim je iskombinovana ideja iz jednog Ruby&Rails plugin-a koji je napisao neki IBM developer.

# U čemu je problem

Rešenje pokušava da pruži odgovore kako dodati nova razvrstavanja bez menjanja baze. Recimo na primeru tabele konto:

The diagram illustrates a SQL code snippet within a 'SQL pane' window. To the left of the pane, there is a vertical double-headed arrow labeled 'Kandidati' (Candidates). A large blue thought bubble is positioned to the right of the code, containing the text: 'Šta kada je potrebno dodati nova podešavanja za konta, npr: prihod\_zakup, prihod\_proizvodi, prihod\_renta, porez\_dobit, kupci-povezana ...'.

```
SQL pane
CREATE TABLE asoft_finansije.konto
(
    id bigint NOT NULL DEFAULT nextval('a
    sifra character varying NOT NULL,
    datum_isknjizenja date NOT NULL,
    kolicina numeric(15,2) NOT NULL,
    vrednost numeric(15,2) NOT NULL,
    opis character varying NOT NULL,
    zatvaranje character varying NOT
    pocetnostanje character varying NOT
    partner boolean NOT NULL DEFAULT true,
    opisob boolean NOT NULL DEFAULT true,
    datopis boolean NOT NULL DEFAULT true,
    kontrastav boolean NOT NULL DEFAULT false,
    vezdok boolean NOT NULL DEFAULT true,
    devkonto boolean NOT NULL DEFAULT false,
    orgjed boolean NOT NULL DEFAULT false,
```

# Klasifikacije = Tagovi

domaća, inostrana, povezana,  
matična, PDV0, PDV8, PDV18,  
oslobodjena, zavisna, prihodi,  
rashodi, troškovi, ulaganja,  
kamate, kompenzacije, sredstva,  
obrtna, fiksna, nekretnine,  
reprezentacija, zavisna, zbirno,  
pojedinačno, dugovno, devizni  
analitički, dinarski, američki,  
2010, 2011 ...

# Razmišljanje..

Recimo da smo Amasis i da imamo račun za nabavku polovne proizvodne linije od Pionira.

Za Amasis je to **ulazni račun** koji stvara **eksterni trošak** za **nabavku proizvodnog osnovnog sredstva** od **matičnog preduzeća** koje je u **sistemu PDV-a**.

# Razmišljanje..

Ukoliko tagujemo:

- Pionir sa  
dobavljač, domaće, matično,  
u\_sistemu\_pdv
- Dokument sa  
račun, ulazni, nabavka
- Stavku sa  
sredstvo, osnovno, proizvodno

# Razmišljanje..

Ukoliko istim ovim tagovima označimo šemu knjiženja koju želimo da primenjujemo na ovakve slučajeve -

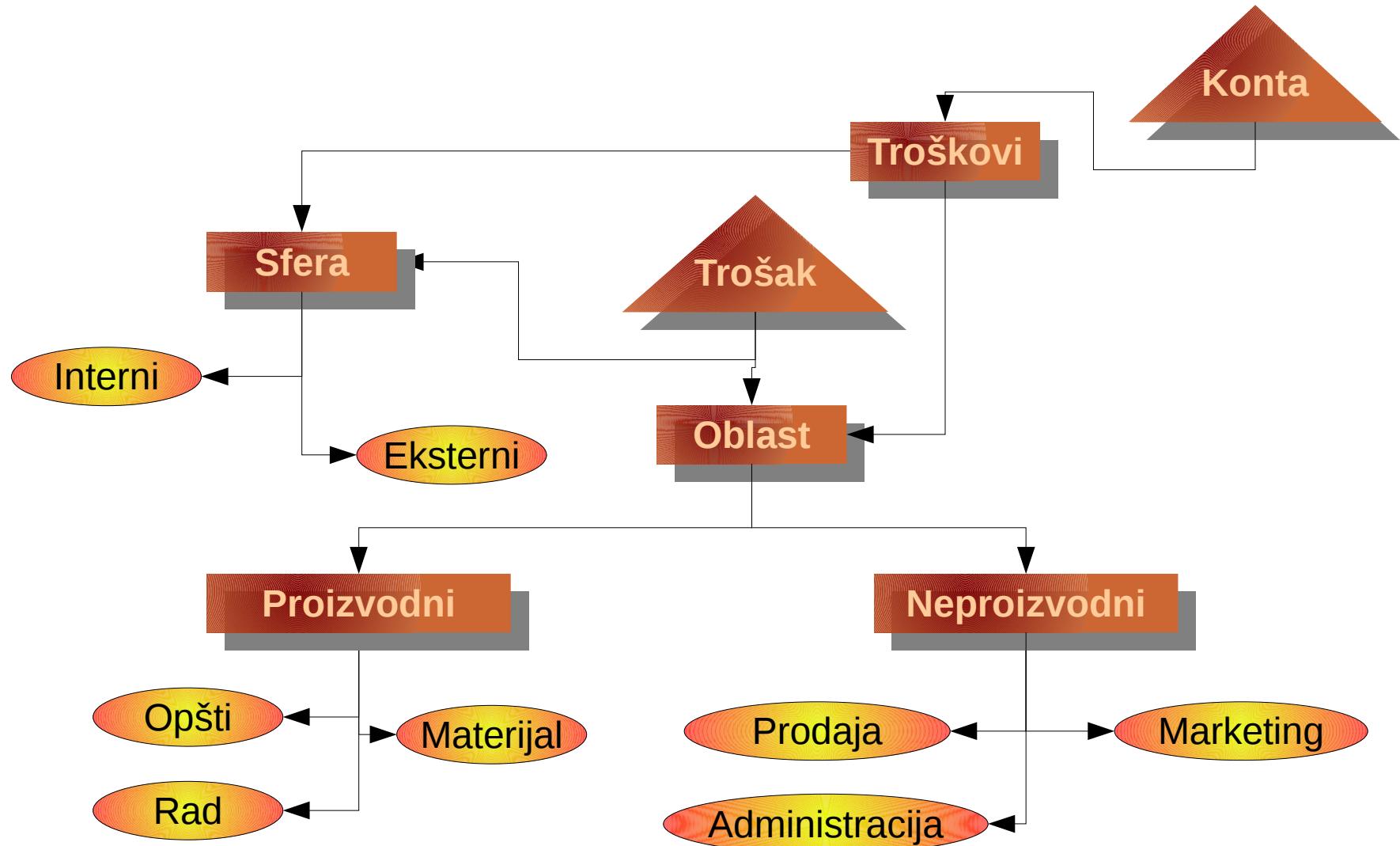
**Automatizovao bi se korak izbora šeme knjiženja nakon evidencije narednih dokumenata.**

# Razmišljanje..

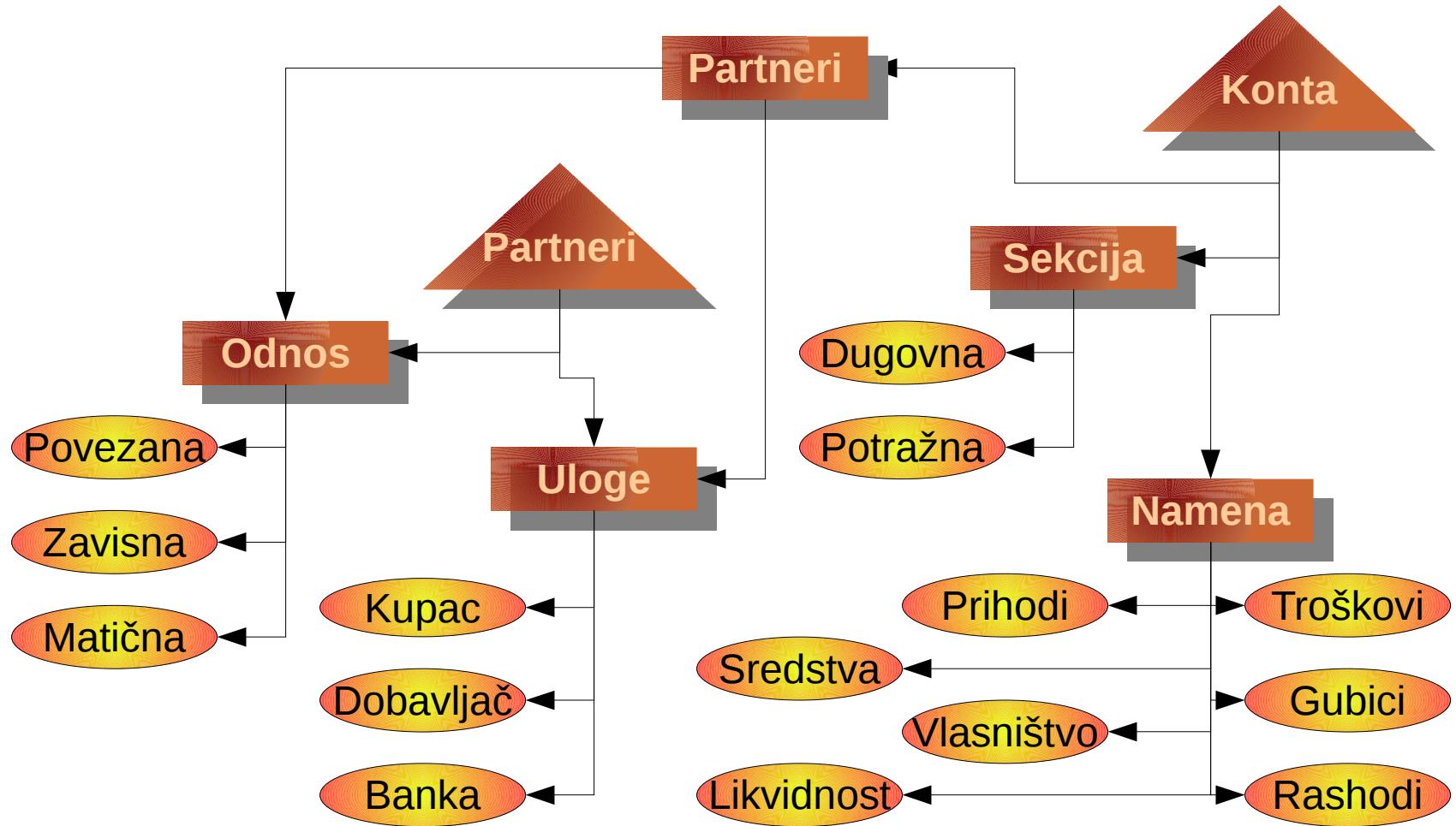
Ukoliko ovim i drugim tagovima označimo **konta** -

**Ukupan broj šema knjiženja bi mogao da se svede na svega nekoliko, jer bi se potrebna konta nalazila automatski.**

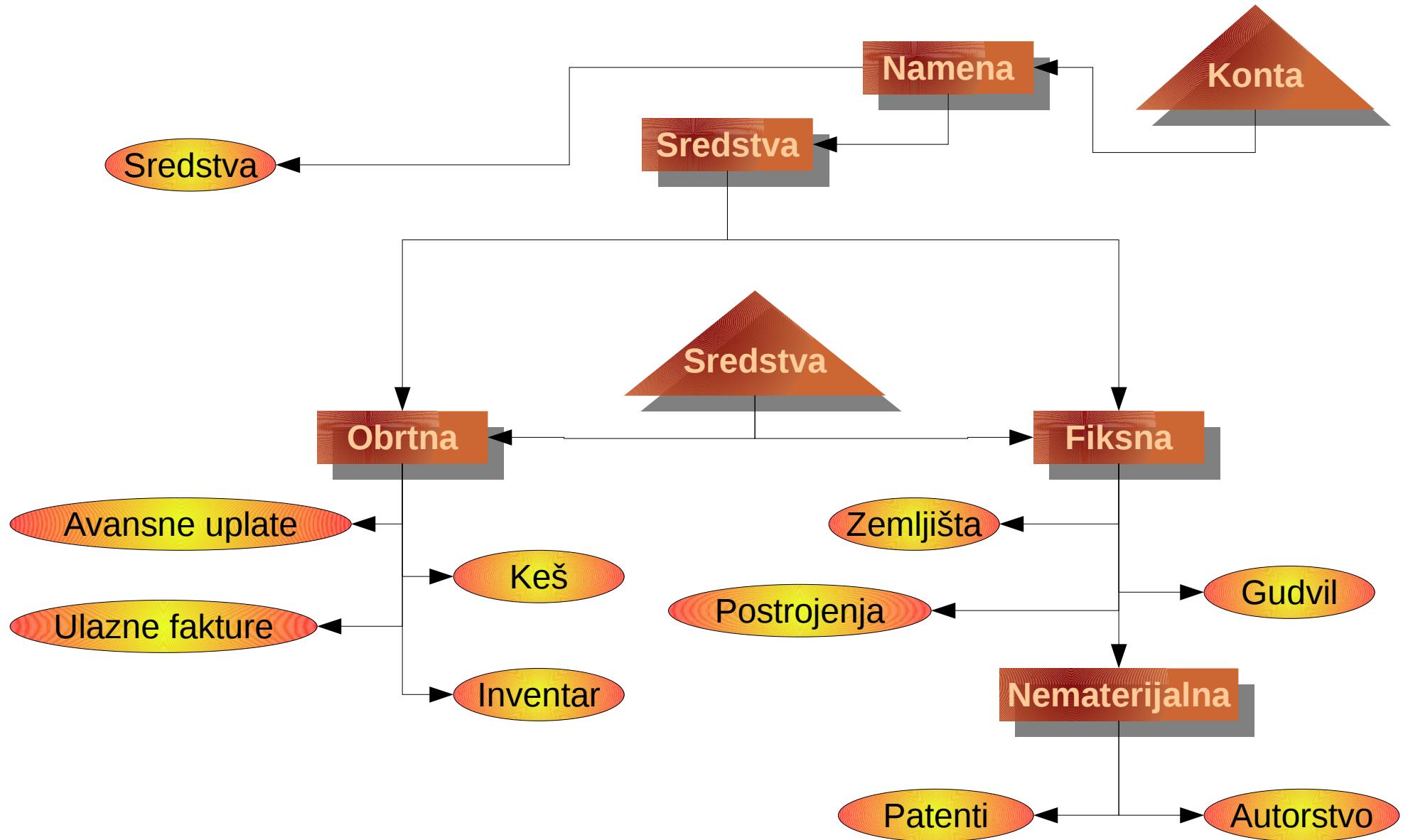
# Klasifikacija troškova



# Klasifikacija partnera i konta



# Klasifikacija sredstava

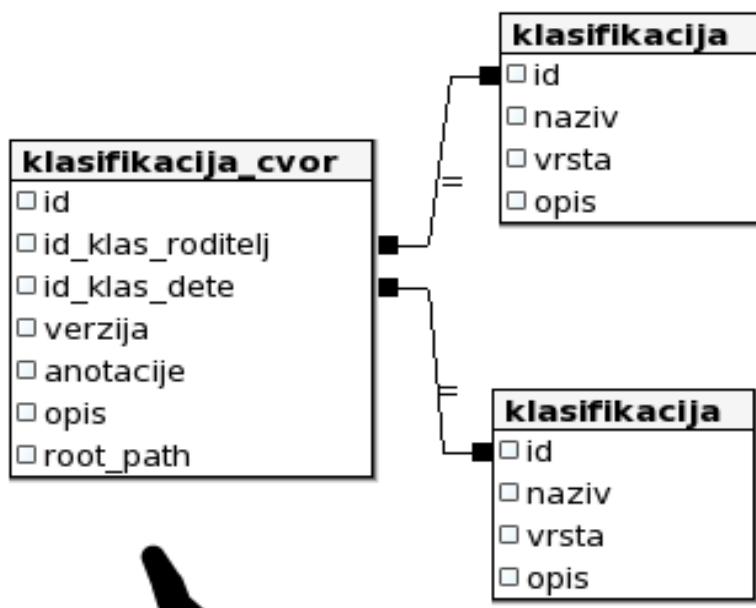


# Koja je korist

- 1-**Univerzalno razvrstavanje**, svih entiteta koji se koriste u sistemu
- 2-**Dinamičko razvrstavanje**, bez potrebe za izmenom fizičke strukture baze
- 3-**Automatsko prepoznavanje**, srodnih entiteta koje je potrebno primeniti u implementaciji nekog konkretnog slučaja korišćenja (??Experimental??)

# Model podataka

# Formiranje klasifikacionog grafa



root	default
partneri	top_kategorija
dokumenta	top_kategorija
konta	top_kategorija
narudzbenica	default
maticno	default
zavisno	default
povezano	default
dobavljenici	default
kupci	default
banke	default
odnos	kategorija

	<b>id</b> [PK] <b>bigint</b>	<b>id_klas_roditelj</b> <b>bigint</b>	<b>id_klas_dete</b> <b>bigint</b>	<b>verzija</b> <b>integer</b>	<b>anotacije</b> <b>character varyin</b>	<b>opis</b> <b>character</b>	<b>root_path</b> <b>character</b>
1	0		1	0	"	opisite asoc	"
2	0		2	0	"	opisite asoc	"
3	0		3	0	"	opisite asoc	"
4	1		12	0	mi, vi	Uloge partnera	"
5	2		13	0	"	opisite asoc	"
6	3		26	0	"	opisite asoc	"

# DbSchema

```
wget http://www.dbschema.com/dbschema.zip
```

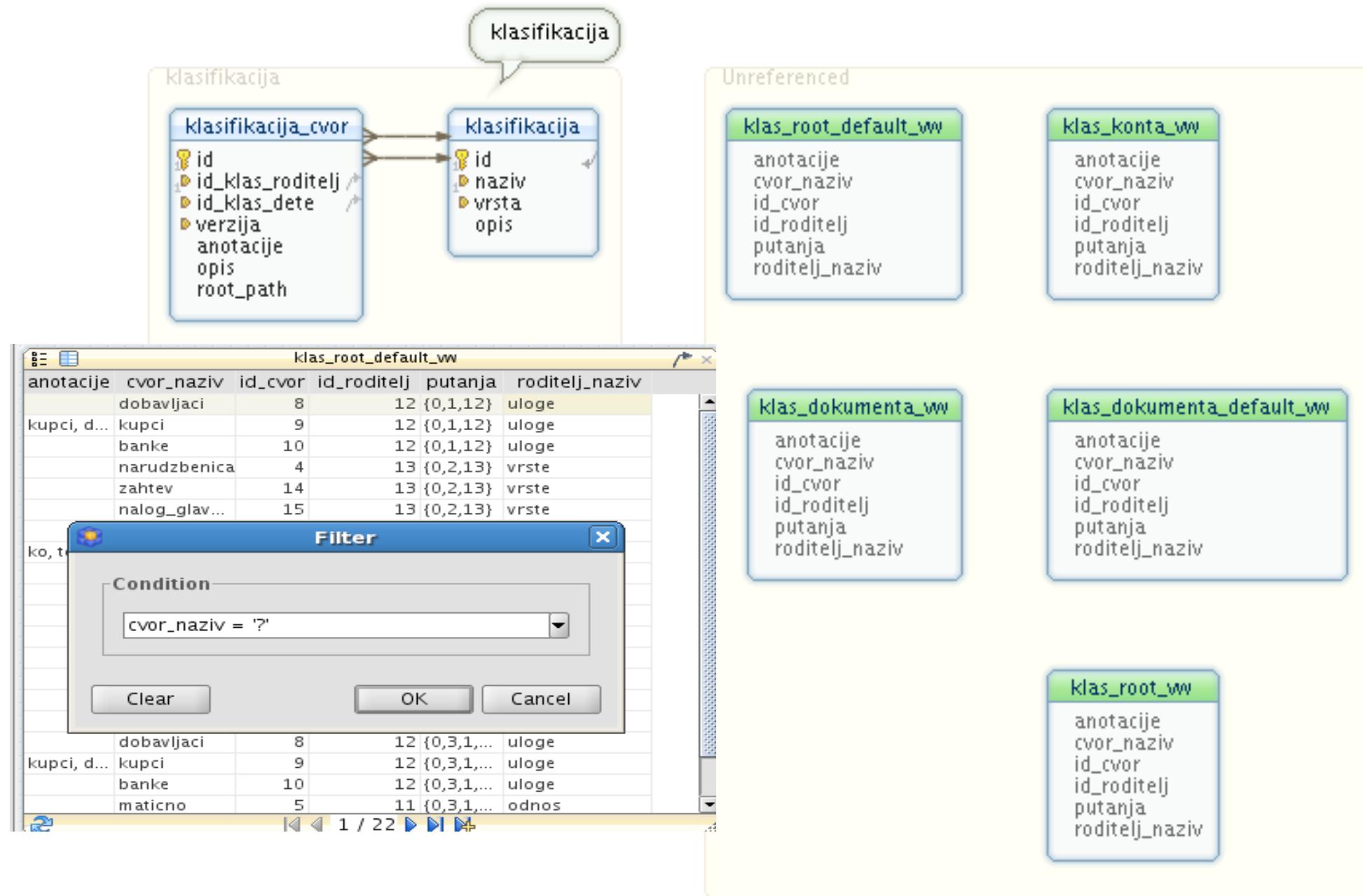
```
unzip dbschema.zip
```

```
cd DbSchema
```

```
chmod +x dbschema.sh
```

```
./dbschema.sh
```

# Osnovni graf sa db view-ima



# Power of Postgres Views

## 36.2.3. The Power of Views in PostgreSQL

The benefit of implementing views with the rule system is, that the planner has all the information about which tables have to be scanned plus the relationships between these tables plus the restrictive qualifications from the views plus the qualifications from the original query in one single query tree. And this is still the situation when the original query is already a join over views. The planner has to decide which is the best path to execute the query, and the more information the planner has, the better this decision can be.

## Db views - implementacija

```
CREATE OR REPLACE VIEW asoft_klas.klas_konta_vw AS
SELECT klasifikuj_od_korena.id_cvor, klasifikuj_od_korena.cvor_naziv,
       klasifikuj_od_korena.id_roditelj, klasifikuj_od_korena.roditelj_naziv,
       klasifikuj_od_korena.putanja, klasifikuj_od_korena.anotacije
  FROM asoft_klas.klasifikuj_od_korena(3, '%'::character varying)
klasifikuj_od_korena (id_cvor, cvor_naziv, id_roditelj, roditelj_naziv, putanja, anotacije);
```

```
CREATE OR REPLACE VIEW asoft_klas.klas_konta_vw_default AS
SELECT klasifikuj_od_korena.id_cvor, klasifikuj_od_korena.cvor_naziv,
       klasifikuj_od_korena.id_roditelj, klasifikuj_od_korena.roditelj_naziv,
       klasifikuj_od_korena.putanja, klasifikuj_od_korena.anotacije
  FROM asoft_klas.klasifikuj_od_korena(3, 'default'::character varying)
klasifikuj_od_korena (id_cvor, cvor_naziv, id_roditelj, roditelj_naziv, putanja, anotacije);
```

## Db views – central function

```
CREATE TYPE asoft_klas.aklas AS (id_cvor bigint, cvor_naziv character varying,
                                   id_roditelj bigint, roditelj_naziv, character varying, putanja integer[], anotacije text);
CREATE OR REPLACE FUNCTION asoft_klas.klasifikuj_od_korena(integer, character varying)
RETURNS SETOF asoft_klas.aklas AS
$BODY$
DECLARE
    r asoft_klas.aklas;
BEGIN
    FOR r IN EXECUTE 'WITH RECURSIVE prolaz AS (
        SELECT klasifikacija_cvor.id_klas_dete, klasifikacija_cvor.id_klas_roditelj,
               1 AS dubina, ARRAY[klasifikacija_cvor.id_klas_roditelj::integer] AS putanja,
               false AS ciklicni, klasifikacija_cvor.anotacije
        FROM asoft_klas.klasifikacija_cvor WHERE klasifikacija_cvor.id_klas_roditelj = ' || $1 || '
        UNION ALL SELECT b.id_klas_dete, b.id_klas_roditelj, a.dubina + 1,
                        a.putanja || ARRAY[b.id_klas_roditelj::integer],
                        b.id_klas_roditelj = ANY (a.putanja), b.anotacije
        FROM prolaz a, asoft_klas.klasifikacija_cvor b
        WHERE a.id_klas_dete = b.id_klas_roditelj AND NOT a.ciklicni )
        SELECT p.id_klas_dete AS id_cvor, klas1.naziv AS cvor_naziv,
               p.id_klas_roditelj AS id_roditelj, klas2.naziv AS roditelj_naziv,
               p.putanja, p.anotacije
    FROM prolaz p
    JOIN asoft_klas.klasifikacija klas1 ON p.id_klas_dete = klas1.id AND klas1.vrsta = "" || $2 || ""
    JOIN asoft_klas.klasifikacija klas2 ON p.id_klas_roditelj = klas2.id;
    LOOP
        RETURN NEXT r;
    END LOOP;
    RETURN;
END;
$BODY$
LANGUAGE plpgsql
```

# **Model podataka 2**

## **Povezivanje sa sistemom**

ubuntu®