Heterarhije in relacijski podatkovni modeli

Tomaž Lajovic, Iztok Lajovic
KRES Kreativni sistemi, d. o. o., Ljubljana
tomaz.lajovic@kres-ks.si; iztok.lajovic@kres-ks.si

Izvleček

Pri razvoju poslovnih rešitev z lastno informacijsko platformo Panorama smo uspešno presegli vrsto omejitev tradicionalnih relacijskih modelov. V standardni relacijski bazi smo združili prednosti relacijskega na eni in objektnega modela na drugi strani: preglednost in učinkovitost relacijskega smo nadgradili z objektnim načinom oblikovanja informacijskega prostora.

Gradnja relacijskih modelov je praviloma osredinjena na zadovoljitev minimalnih potreb dejanskih poslovnih procesov, in sicer predvsem v luči zahtevanih poročil oziroma poizvedb. Naš pristop se rahlo razlikuje, saj je primarna zahteva pri gradnji podatkovnega modela popolnost opisa (mogočih) medsebojnih odnosov, nato pa podatkovni model dopolnimo z izvedenimi (posrednimi) povezavami, s katerimi je uporabniku omogočen neposreden vpogled v zanj zanimive podatke o opazovanem predmetu. Pri tem smo naleteli na zanimiv izziv kot posledico različnih vrst in vsebine podatkov, ki so lahko uporabljeni kot atribut opazovanih predmetov – izziv heterarhij oziroma uvrščenosti v množico hierarhij, ki definirajo opazovani predmet. Gre za vprašanja interpretacije lastnosti opazovanega predmeta, ki so dejansko lastnosti predmetov, katerim pripada sam, na eni strani ter lastnosti predmetov, ki pripadajo opazovanemu predmetu, na drugi strani.

S preprostim primerom predstavimo kritične zahteve in želeni rezultat ter izhodišča za ureditev zahtevanih podatkovnih struktur, s katerimi podpremo upravljanje heterarhičnih struktur in njihovo pravilno interpretacijo.

Ključne besede: podatkovne baze, objektno-relacijsko mapiranje, sklici, razvejani sklici, hierahije, heterarhije, upravljanje matičnih podatkov.

Abstract

Heterarchies and Relational Data Models

In the development of business solutions with our own IT platform Panorama we have successfully overcome limitations of the traditional relational models. In a standard relational database, we combine the advantages of relational on the one hand and of object-oriented models on the other: the transparency and efficiency of the relational model has been upgraded with an object-oriented method of creating an information space.

Construction of the relational models is generally focused on meeting the minimal needs of real business processes, especially in the light of the required reports or queries. Our approach is slightly different because the primary requirement for building the data model is a complete description of the (potential) relationships and then the data model is amended with derivative (indirect) relations, allowing the user direct insight into the information on the observed object. In doing so, we came across an interesting challenge as a result of different types of content and data that can be used as an attribute of the observed objects – the challenge of heterarchies or multitude of hierarchies, which define the observed object. It is a question of interpretation of the properties of observed objects, that are properties of objects that itself (hierarchically) belongs to, on the one hand and of the properties of objects that (hierarchically) belong to the observed object.

A simple example is used to present critical requirements and the desired outcome as well as underlying principles for the regulation of the required data structures to support management of heterarchical structures and their proper interpretation.

Keywords: databases, object-relational mapping, links, branched links, hierachies, heterarchies, master data management.

1 UVOD

Uporaba enotnih podatkovnih modelov za več informacijskih namenov je zaradi sprotne implicitne medsebojne sinhronizacije vedno bolj aktualna. Dostopnih virov podatkov je namreč vedno več in medsebojno usklajevanje in/ali dostopanje do vsakega vira posebej je nekako neskladno s pričakovanji, ki jih gojimo do informatizacije.

Pri razvoju poslovnih rešitev z lastno informacijsko platformo Panorama smo v podjetju KRES uspešno presegli vrsto

omejitev tradicionalnih relacijskih modelov. V standardni relacijski bazi smo združili prednosti enostavnosti vzdrževanja relacijskega na eni in sistematičnosti gradnje objektnega modela na drugi strani: preglednost in stabilnost relacijske baze smo nadgradili z opisovanjem dejanskosti prilagodljivim objektnim načinom podatkovnega modeliranja.

Tako smo omogočili podatkovno modeliranje neodvisno od končnega cilja – podatkovne strukture naj čim bolj verno odslikavajo realnost, dejanske informacijske potrebe pa zadovoljujemo z opisom zahtev na metanivoju nad enotnim podatkovnim modelom. Še več, zaradi zahtevnosti upravljanja matičnih podatkov (angl. master data management) in z njim povezanega usklajevanja zatečenih oziroma obstoječih podatkovnih baz smo želeli še najmanj enostavno dopolnjevanje enotne podatkovne strukture ter obvladovanje posameznih časovnih in jezikovnih različic atributov predmetov. Vsega naštetega sicer še nismo izpolnili, v tem članku pa smo se osredinili na nekatere probleme, povezane s hierarhijami.

Panorama nastaja od leta 1984 kot nadgradnja projekta izgradnje centralnega podatkovnega slovarja za vse lastne aplikacije tedanjega podjetja Iskra Delta, ki ga je razvijala skupina pod vodstvom avtorja Panorame in soavtorja tega članka, Iztoka Lajovca. Izhaja iz implementacije konceptov asociativnega razmišljanja, kar je neodvisno v svojem delu iz okvirno istega časa opisal Minsky (Minsky, 1986). Razvoj misli in razvoja Panorame je bil večkrat objavljen (Lajovic, 2002; Lavbič, Lajovic & Krisper, 2010), ta članek pa temelji na prispevku, ki sva ga avtorja predstavila na Dnevih slovenske informatike 2012 (Lajovic & Lajovic, 2012).

V nadaljevanju z nekaj primeri predstavimo bistvene prednosti panoramskega objektno-relacijskega pristopa. Na kratko so to štiri:

- **pregledno urejene datoteke metapodatkovnega nivoja**, ki služijo kot podatkovni slovar in vodilo upravljanja s podatki;
- **dosledna dvosmernost povezav** je omogočena s t. i. *vezmi*, ki vsakemu *sklicu* (atribut v tabeli A, ki je posamična vrednost iz tabele B) priredi *seznam* (atribut v tabeli B, ki je seznam vrednosti iz tabele A), in s čimer je omogočena hitra in nedvoumna interpretacija vseh povezav na vsebinskem nivoju;
- **podpora razvejanim povezavam**, ki omogočajo enotnost interpretacije inherentno raznovrstnih medsebojnih odnosov, in sicer v obliki:
  - *razvejanih sklicev*: atribut v tabeli M, ki je posamezna vrednost iz katere koli izmed tabel N, O, P, ter
  - *skupnih seznamov*: atribut v tabeli W je nabor vrednosti iz tabel X, Y, Z;
- **omogočanje *oddaljenih sklicev* in *seznamov***, s katerimi se poenostavi pridobivanje podatkov v sicer kompleksnem visoko strukturiranem podatkovnem modelu.

V sklepu nakažemo smer nadaljnjega razvoja misli o heterarhijah in njene implementacije v panoramskih strukturah.

2 IZHODIŠČA ZA GRADNJO PANORAMSKIH MODELOV

2.1 Osnovno o Panorami

Panorama je podatkovna platforma, ki uporabniku omogoča prosto prilagajanje in nadgrajevanje podatkovnega modela. Vgrajeno ima relacijsko bazo, skladno z ANSI SQL-92 specifikacijo, tako da se vse strukture in datoteke zapisujejo v standardnih relacijskih tabelah s pripadajočimi indeksi, uporabnik pa vse upravlja prek grafičnega vmesnika, ki ne zahteva znanja programskih jezikov.

Kot dinamični sistem sproti in avtomatsko generira vse potrebne tabele, ukaze, poti in interpretacije, in sicer z uporabo slovarskih tabel, s katerimi so opisana vodila za sprotno avtomatsko pripravo potrebnih ukazov SQL. Na eni strani s tem dosežemo, da po kompleksnih podatkovnih strukturah lahko potuje tudi programiranja nevešč uporabnik, na drugi strani pa omogočamo tudi nadaljnjo gradnjo za realizacijo jutrišnjih podatkovnih in informacijskih zahtev brez vplivanja na že obstoječi in delujoči sistem.

Primarno je Panorama namenjena poslovnim aplikacijam in je izvedena v standardni arhitekturi odjemalec/strežnik v okoljih Windows. Dostop z mobilnih platform oziroma prek spleta nameravamo podpreti omejen nabor funkcionalnosti, in sicer predvsem v smislu učinkovite podpore pregledovanja ter ažuriranja podatkov na nivojih II in VI (nivoji so opisani v nadaljevanju tega razdelka).

Večina postopkov priprave ukazov za bazo podatkov in celotno grafično okolje se izvaja na strani odjemalca, na strani strežnika pa se nahajajo datoteke ter izvajajo ažuriranja in obdelave zahtev posameznih odjemalcev. Za upravljanje uporabniških pravic in preferenc posameznikov je uporabljen inovativni

samoreferenčni sistem pravil, ki omogoča učinkovito obvladovanje in izmenjavo strukturiranih podatkov.

Tabele in datoteke panoramskega modela so po funkcijah razvrščene v te ravni:

I. slovarska raven – definicija podatkovnega modela in poizvedb;
II. podatkovna raven – podatki in rezultati poizvedb;
III. pristopna raven – upravljanje s pravicami dostopa, overjanje in avtorizacija uporabnikov ter uvozi in sinhronizacije z zunanjimi viri;
IV. oblikovalni nivo – prikazi, filtri, uporabniške nastavitve, izpisi in izvozi;
V. evidenčni nivo – beleženje dostopov in uporabe;
VI. dokumentni nivo – dokumenti in datoteke;
VII. arhivski nivo – arhivske datoteke celotne baze.

Vse datoteke nivojev I do V so del enotne baze, medtem ko so datoteke dokumentnega in arhivskega (VI in VII) nivoja ločene, in sicer so dokumenti in datoteke shranjeni v obliki posebne odvisne baze, arhivi pa so povsem neodvisne datoteke (posnetki stanja celotne baze v danem trenutku). V članku so predstavljene nekatere rešitve slovarskega (I) in podatkovnega (II) nivoja, ki sta temelja vsebinske prilagodljivosti in enostavnosti upravljanja panoramskih modelov.

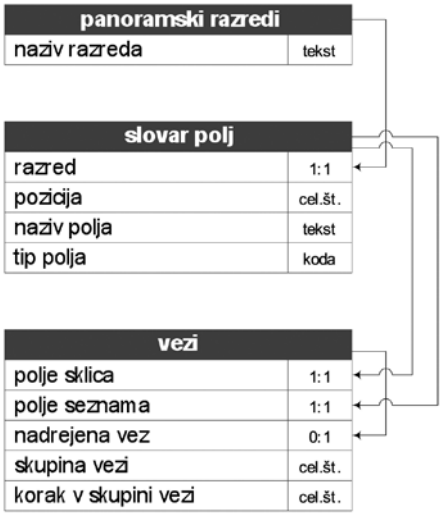
2.2 Upravljanje podatkovnega modela

Tabele podatkovnega modela (nivo II) so opredeljene z zapisi v treh slovarskih (nivo I) tabelah:

1. *panoramski razredi* s poimenovanji tabel;
2. *slovar polj* z navodili za oblikovanje stolpcev posameznih podatkovnih tabel;
3. *vezi*, v katerih so podrobno opisane vse povezave med polji tabel.

Medsebojna razmerja treh temeljnih tabel slovarskega nivoja so predstavljena shematično (slika 1), podrobnejša legenda pa se nahaja v nadaljevanju (slika 2).

Način gradnje podatkovnega modela z opisom logičnega modela zahteva enoličnost interpretacije vseh parametrov fizičnega modela, saj se ta zgradi sam v skladu s pravili, ki so kot temeljna pravila vgrajena v program. Zapisi v tabelah slovarskega nivoja popolnoma izpolnjujejo ta kriterij, istočasno pa zagotavljajo tudi nedvoumnost interpretacij vsebine povezav prek vezi oziroma parov polj *sklic* in *seznam*:



Slika 1: **Panoramske tabele slovarskega nivoja (poenostavljena shema)**

- 'naziv polja' za tip *sklic:*
  naziv atributa predmeta,
- 'naziv polja' za tip *seznam:*
  vsebina povezanosti predmeta s predmeti, navedenimi v pripadajočem seznamu.

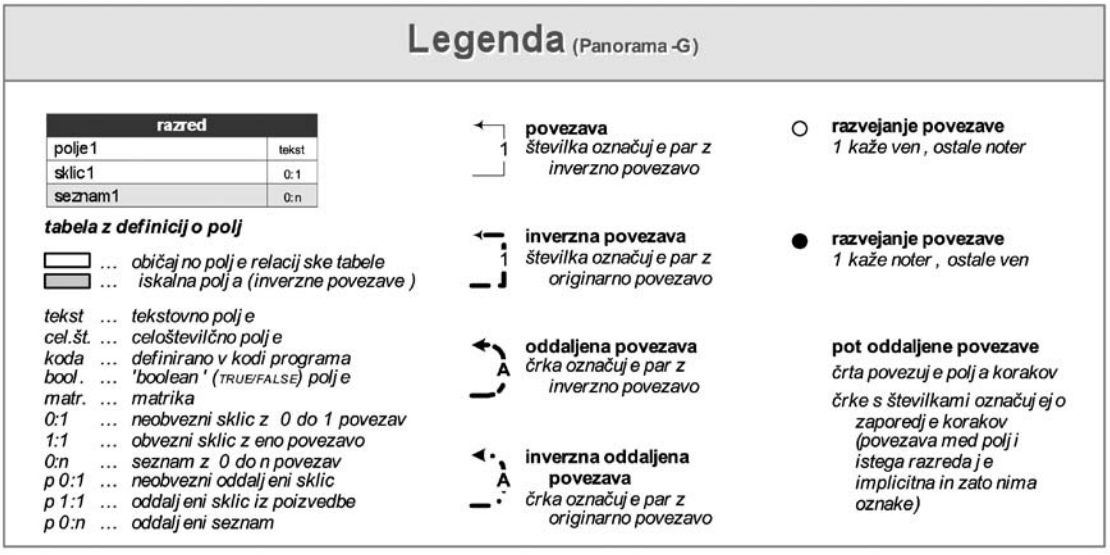
Ob tem namena slovarske datoteke *vezi* še ne moremo razložiti v celoti, saj za opis običajnih relacijskih modelov povsem zadostujeta prvi dve datoteki. Prav slovarski datoteki se bomo v nadaljevanju posebej posvetili; nepogrešljiva je namreč tako za gradnjo razvejanih relacijskih struktur kot za predlagani pristop k formalizaciji heterarhičnih struktur.

V zvezi z normalizacijo oziroma veljavnostjo posameznih normalnih form je treba poudariti, da Panorama sama zase ne zagotavlja katere koli stopnje normalizacije baze – prek samoumevne 1NF (Date, 2012) – ki jo zgradimo z njo. Seveda pa je – poleg tega, da že sama vodi razmišljanje ob modeliranju v bolj »normalno« smer – lahko s svojimi slovarskimi datotekami v veliko pomoč pri za normalizacijo zahtevani analitiki in posledični optimizaciji podatkovnih baz.

3 PRIMERI PANORAMSKIH PODATKOVNIH STRUKTUR

S Panoramo kot informacijsko platformo smo želeli uporabnikom omogočiti čim večjo svobodo pri oblikovanju lastnih podatkovnih modelov. Panoramske podatkovne strukture so omejene na neposredno podprte v implementaciji samega programa, pri čemer smo uvedli tudi nekaj posebnosti, ki so shematično s preprostimi primeri predstavljeni v nada-

ljevanju. Za namene predstavitve panoramskih podatkovnih struktur smo razvili poseben shematski zapis, imenovan Panorama-G, katerega glavni elementi so predstavljeni v legendi (slika 2). V takem shematskem zapisu bodo predstavljeni vsi primeri v nadaljevanju.



Slika 2: **Legenda shematskega zapisa Panorama-G**

Zaradi preglednosti smo ob shemah tabelarično navedli tudi pripadajoče zapise v slovarski tabeli »vezi« (slika 1), v kateri smo za lažje razumevanje dodali zadnji stolpec »tip vezi«, ki ga sicer ni v tabeli. Vrednosti v tem stolpcu so lahko:

- *elementarna* – osnovni gradnik podatkovnega modela;
- *oddaljena – temeljna* – izvedeni gradnik metapodatkovnega modela, in sicer začetna oziroma končna (v primeru oddaljenega sklica je to ena in ista) vez poti skozi podatkovni model;
- *oddaljena – vmesna* – izvedeni gradnik metapodatkovnega modela, in sicer vmesna vez na poti skozi podatkovni model;
- *korak* – opis poti od začetne temeljne oddaljene vezi do končne temeljne oddaljene vezi s koraki prek elementarnih vezi podatkovnega modela.

3.1 Sklic in seznam

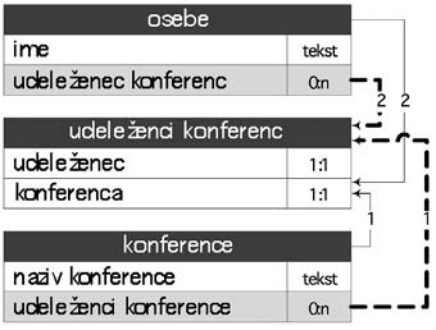
V standardnem relacijskem modelu so *sklici* povezave med dvema tabelama, pri čemer se vsebina polja v prvi tabeli polni s predmeti iz druge tabele. Ker ima tudi obratna pot svojo specifično informacijsko vrednost – kateri predmeti iz prve tabele se sklicujejo na posamezne predmete iz druge tabele in s kakšnim namenom –, smo uvedli izrecno deklaracijo *seznamov*: vsako polje *sklica* ima na drugi strani povezave, v razredu, na katerega se sklicuje, polje navedbe pripadajočega *seznama* (slika 3, tabela 1). Vsebina *seznama* je, skladno s standardi, ki veljajo v relacijskih bazah, rezultat poizvedbe v tabeli *sklica*: polje tipa *seznam* namreč obstaja le kot vodilo v slovarskih datotekah, tabela, ki ji pripada *seznam*, pa nima ustreznega stolpca.

Slika 3: **Povezava [ena : mnogo] z izrecno deklaracijo *seznama* v razredu, na katerega kaže *sklic***

Tabela 1: **Vezi k sliki 3**

VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	barva las	osebe s to b. las				*elementarna*

Na prvi pogled ta pristop prinaša zgolj odstop od obstoječih normalnih form, vendar pa gre pri tem dejansko za posebno dopolnitev, ki ponuja množico dodatnih možnosti in optimizacij. Ena izmed njih je lahko razvidna že iz preprostega primera povezave, pri kateri se poljubno mnogo predmetov prvega razreda povezuje s poljubno mnogo predmeti drugega (slika 4, tabela 2). Če nas zanima, kateri predmeti preostalih razredov (danega podatkovnega modela) se sklicujejo na predmete razreda »osebe«, nam polje tipa *seznam* v definiciji razreda »osebe« navede, da se od obeh ostalih dveh razredov v našem modelu nanj lahko sklicujejo le predmeti razreda »udeleženci konferenc«, zato bomo pri zadani poizvedbi preiskali zgolj ta razred v zadevnem stolpcu (z uporabo pripravljenega indeksa) in preprosto izpustili drugega (ter morebitne preostale).

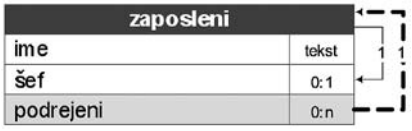


Slika 4: **Povezava [mnogo : mnogo]**

Tabela 2: **Vezi k sliki 4**

VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	konferenca	udeleženci konf.				*elementarna*
2	udeleženec	udeleženec konf.				*elementarna*

Dodatno, zgolj za ilustracijo načina delovanja, podajamo tudi sliko razreda s preprosto rekurzivno povezavo (slika 5, tabela 3) kot eno izmed običajnih relacijskih struktur. Predstavlja zaposlene, pri čemer ima vsak zaposleni lahko samo enega šefa (ki je prav tako eden izmed zaposlenih), vsak šef pa poljubno število neposredno podrejenih.



Slika 5: **Rekurzivna povezava [ena : mnogo]**

Tabela 3: **Vezi k sliki 5**

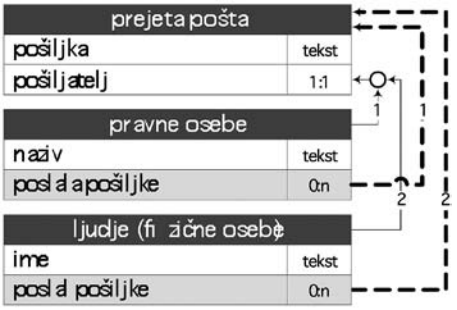
VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	šef	podrejeni				*elementarna*

3.2 Razvejane povezave

Pari polj v povezavah niso nujno ekskluzivni, kar je prvi izmed razlogov za potrebnost slovarske datoteke »vezi« (slika 1). Vsako posamezno polje namreč lahko nastopa v več povezavah. Posamezno polje tipa *sklic* je lahko v paru/-ih z enim ali več polji tipa *seznam* iz istega ali različnih razredov ter vsako polje tipa *seznam* je lahko v paru/-ih z enim ali več polji tipa *sklic*. Takšna polja imenujemo *razvejani sklic* in *razvejani seznam* (za tega je morda bolj ilustrativno poimenovanje *skupni seznam*).

Razvejanost povezav ima za posledico, da je treba pri vsakem sklicevanju oziroma poizvedbah poleg identifikacije zapisa v tabeli identificirati tudi tabelo samo – v Panorami je to urejeno prek slovarske tabele »panoramski razred« (slika 1). Ob tem je treba povedati, da so procedure, ki vodijo izvajanje operacij, povezanih z razvejanimi strukturami, precej kompleksnejše od procedur, ki jih zahtevajo standardne (nerazvejane) povezave.

Razvejani sklic je vrsta povezave, ki se uporabi, kadar je vsebina posameznega polja določenega razreda lahko predmet iz različnih razredov, kakršen je primer prejete pošte (slika 6, tabela 4). Pošto nam namreč lahko pošljejo tako pravne kot fizične osebe, vendar imajo pravne in fizične osebe precej različne atribute, zato jih je smiselno voditi v ločenih razredih.



Slika 6: ***Razvejani sklic* (polje sklica se nanaša na več razredov)**

Tabela 4: **Vezi k sliki 6**

VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	pošiljatelj	poslala pošiljke				*elementarna*
2	pošiljatelj	poslal pošiljke				*elementarna*

Razvejanost lahko nastopa tudi na strani *seznama*, in sicer kadar je posamezno polje tipa *seznam* v parih z več polji tipa *sklic* – polje tipa *skupni seznam*. *Skupni seznami* so malce bolj abstraktna oblika, katere namen in potrebnost postaneta jasnejša ob preiskovanju kompleksnejših podatkovnih struktur.

Redek primer na elementarnem podatkovnem nivoju so rodbinska drevesa, pri katerih nastopa rekurzivna povezava osebe z osebo kar dvakrat, in sicer s svojima materjo in očetom (slika 7, tabela 5). V tem primeru sta povezavi različni (mati, oče), *seznam* obeh pa je skupen (otroci).

Slika 7: **Razvejani ali skupni seznam (dva ločena sklica imata skupni pomen seznama)**

Tabela 5: **Vezi k sliki 7**

VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	oče	otroci				*elementarna*
2	mama	otroci				*elementarna*

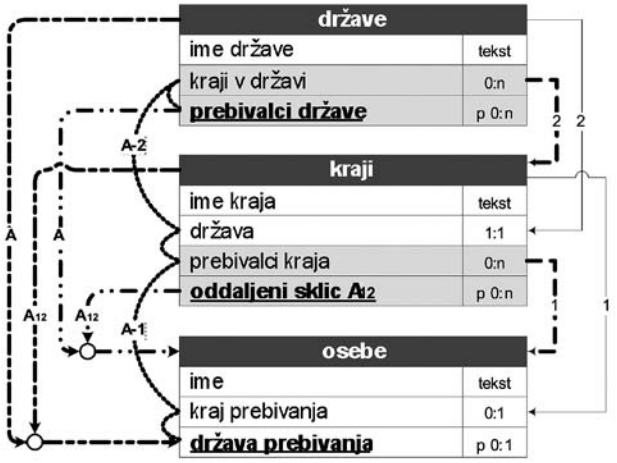
3.3 Oddaljene povezave

Oddaljene povezave so posredne oziroma sestavljene povezave med predmeti dveh oddaljenih razredov, ki obstojijo le, če v danem trenutku obstojijo vsi koraki povezave. Služijo za neposreden prikaz povezav, ki so sicer posredne, in so po obliki materializirani pogledi (*angl.* materialized view). Glede na kompleksnost procedur, ki so potrebne za njihovo vzpostavitev in vzdrževanje, ločimo dva tipa, in sicer *oddaljeni sklic* in *oddaljeni seznam*.

Za obe vrsti velja, da jih ne moremo ažurirati neposredno, saj so vrednosti vedno rezultat obstoja neprekinjene povezave po definirani poti skozi podatkovni model. Njihova vrednost je v preprostosti definicije in razumljivosti za uporabnika, bistvena posledica pa, da z dodajanjem oddaljenih povezav višamo informativnost tako povezanih razredov.

Prvi primer prikazuje vzpostavitev oddaljenega sklica med predmeti razredov »osebe« in »države« prek predmetov razreda »kraji« (slika 8, tabela 6). V tem primeru ima namreč vsaka oseba lahko določen največ en kraj prebivanja, vsak kraj pa ima določeno državo, v kateri se nahaja. Če torej za posamezno osebo obstaja vnos kraja prebivanja, lahko znotraj našega modela pridobimo tudi podatek o državi prebivanja, sicer pa je do tega podatka brez *oddaljenega sklica* mogoče priti le posredno. Za vzpostavitev neposrednega prikaza povezav se med polji razreda

»osebe« definira *oddaljeni sklic* na razred »države« in določi pot te posredne povezave. Kakor je razvidno iz tega primera, v razredu »osebe« pridobimo podatek o državi prebivanja, v razredu »države« pa *seznam* prebivalcev države.



Slika 8: **Oddaljena povezava tipa oddaljeni sklic**

Tabela 6: **Vezi k sliki 8**

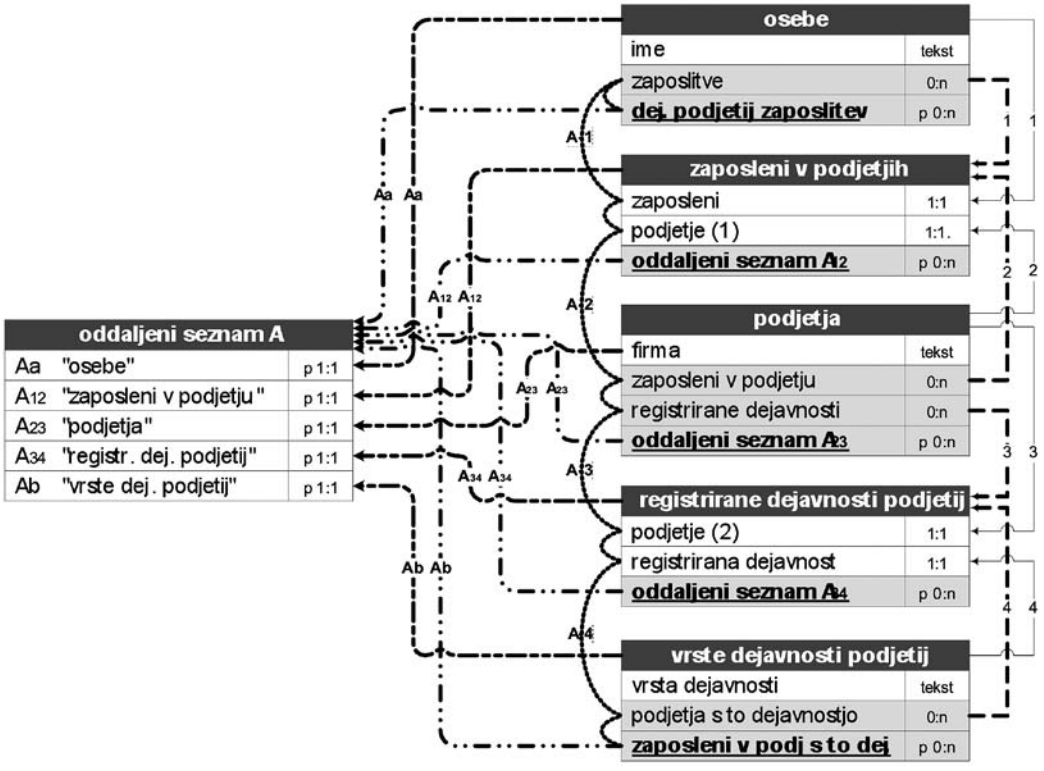
VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	kraj prebivanja	prebivalci kraja				*elementarna*
2	država	kraji v državi				*elementarna*
A	država prebivanja	prebivalci države		1	1	*oddaljena elementarna*
A-1	kraj prebivanja	prebivalci kraja	A	1	2	*korak*
A_{12}	država prebivanja	oddaljeni sklic A_{12}	A-1	1	0	*oddaljena vmesna*
A-2	država	kraji v državi	A-1	1	3	*korak*

Zahteva po strukturiranem zapisu poti je poleg že zgoraj omenjene razvejanosti drugi razlog za potrebnost datoteke »vezi« (slika 1), v katero zapišemo navodila poti oddaljene povezave. Ker je v prehodih med koraki jasno, da gre za prehod med polji znotraj razreda, ni potrebe po izrecnem zapisu teh korakov, saj je popolno in nedvoumno navodilo podano že implicitno. Vsekakor pa ima takšen prehod znotraj razreda za posledico avtomatsko vzpostavitev nove *vmesne oddaljene vezi*, ki služi kot vodilo pri ažuriranju in siceršnjem vzdrževanju pravilnosti vseh zapisov v bazi: tako namreč hitro in nedvoumno najdemo vse zapise, ki bi lahko bili potrebni posodobitve ob posodobitvi katerega koli od predmetov, vključenih v vezi oddaljenega sklica. Takšna vez je v gornjem primeru vez A12, ki si dva parametra (»nadrejena vez« in »skupina vezi«) sicer deli z vezjo A-2, kar jo vse nedvoumno umešča v ustrezno oddaljeno povezavo, vendar pa je njena vrednost parametra »korak v skupini« enaka 0, kar jo opredeli kot *vmesno oddaljeno povezavo*.

Druga oblika oddaljene povezave, *oddaljeni seznam*, sledi isti logiki, vendar pa je zaradi možnosti, da za vsak predmet oddaljeno povezanih dveh razredov obstaja več povezav s predmeti oddaljenega razreda (povezava [mnogo : mnogo]), potrebna vzpostavitev novega povezovalnega razreda, v katerega se zapisujejo vse obstoječe kombinacije povezav oddaljeno povezanih predmetov. Ti metapodatki omogočajo neposredni vpogled in nadaljnje analize zadevnih oddaljenih povezav.

V primeru oddaljenega seznama (slika 9, tabela 7) smo osebam v oddaljeni seznam pripisali združene sezname dejavnosti podjetij vseh njihovih zaposlitev in istočasno vrstam dejavnosti podjetij v oddaljeni seznam pripisali osebe, ki so bile zaposlene v podjetjih z ustreznimi registriranimi dejavnostmi. Z materializacijo »oddaljenega seznama A«, v katerem so posamično zavedene vse obstoječe poti med predmeti razredov »osebe« in »vrste dejavnosti podjetij«, lahko hitro in preprosto pridobimo nove podatke,

npr. navedbo zgolj različnih parov vrednosti začetnega in končnega polja, brez razlikovanja po vrednosti polj na poti med njima (namesto vse množice različnih peterk, ki so zapisane v razredu oddaljeni seznam A, na podlagi transformacije dobimo nov razred vrednosti z vsemi različnimi dvojicami).



Slika 9: **Oddaljena povezava tipa *oddaljeni seznam***

Tabela 7: **Vezi k sliki 9**

VEZI	polje sklica	polje seznama	nadrejena vez	skupina vezi	korak v skupini	tip vezi
1	zaposleni	zaposlitve				*elementarna*
2	podjetje (1)	zaposleni v podj.				*elementarna*
3	podjetje (2)	registr. dejavnosti				*elementarna*
4	registrirana dej.	podjetja sto dej..				*elementarna*
Aa	"osebe"	dej. podjetij zap.		1	1	*oddaljena temeljna*
A-1	zaposleni	zaposlitve	Aa	1	2	*korak*
A12	"zaposleni v p."	odd. seznam A12	A-1	1	0	*oddaljena vmesna*
A-2	podjetje (1)	zaposleni v podj.	A-1	1	3	*korak*
A23	"podjetja"	odd. seznam A23	A-2	1	0	*oddaljena vmesna*
A-3	podjetje (2)	registr. dejavnosti	A-2	1	4	*korak*
A34	"registr. dej.p."	odd. seznam A34	A-3	1	0	*oddaljena vmesna*
A-4	registrirana dej.	podjetja sto dej.	A-3	1	5	*korak*
Ab	"vrste dej. pod."	zaposl. v p. sto d.	A-4	1	6	*oddaljena temeljna*

V primerjavi z *oddaljenim sklicem* so potrebne procedure za vzdrževanje *oddaljenega seznama* bistveno bolj zahtevne, saj ob tem, da terjajo avtomatsko vzpostavitev in vzdrževanje novega razreda (tabele), tudi *temeljna oddaljena vez* med izbranima razredoma razpade na dva dela (v gornjem primeru na vezi Aa in Ab; začetni in končni korak *oddaljenega seznama*).

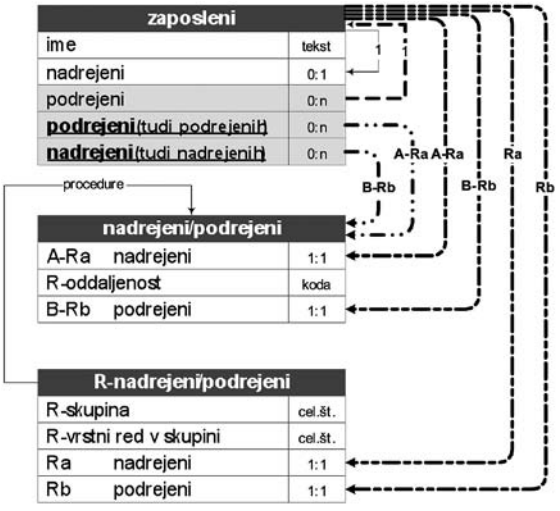
3.4 Obvladovanje rekurzivnih struktur v oddaljenih povezavah

Rekurzivne povezave pri preiskovanju kompleksnih struktur povzročajo težave. Vsak izmed predmetov rekurzivnih povezav ima namreč toliko posrednih povezav s predmeti istega razreda, kolikor je teh istovrstnih predmetov na eni strani njemu neposredno podrejenih in na drugi strani njemu neposredno nadrejenih, zmnoženo z istovrstnimi vrednostmi vsakega izmed teh. Težava, na katero naletimo, je, da se tovrstnih povezav ne da analizirati na podlagi definicije podatkovnega modela (slovarski nivo), ampak na tem nivoju lahko samo ugotovimo, da obstaja rekurzivnost. Da pridobimo podatek o dejanskem številu nadrejenih in podrejenih moramo namreč predhodno analizirati dejanske vrednosti podatkov.

Doslej smo se srečali z dvema primeroma rekurzivnih povezav (sliki 5 in 7), nismo pa se dotaknili posledic tovrstnih struktur na gradnjo oddaljenih povezav. Pri vseh oddaljenih strukturah, predstavljenih zgoraj (sliki 8 in 9), smo namreč z običajnimi strukturami, zapisanimi v slovarskih datotekah, pojasnili vse mogoče oddaljene povezave; rekurzivne strukture pa zaradi svoje narave, ki določene funkcije definicijske (slovarske) ravni privzame tudi na podatkovnem nivoju, v relacijskem modelu terjajo posebno obravnavo. Takšno preiskovanje rekurzivnih struktur poleg razumevanja povezav na slovarskem nivoju zahteva tudi analizo dejanskih povezav na podatkovnem nivoju. Za predstavitev bomo uporabili že uporabljeni preprosti model zaposlenih in njihovih šefov, in sicer z dodano oddaljeno povezavo posredno nadrejenih in podrejenih (slika 10).

Najprej ugotovimo, da gre v primeru podrejenih in nadrejenih za dve plati ene same povezave. V vsakem paru je namreč ena oseba nadrejeni in druga podrejeni. Vse to smo lahko ugotovili iz definicije, za zmožnost nadaljnjih podatkovnih operacij v teh strukturah pa potrebujemo še analizo dejanskih predmetov in njihove povezanosti na podatkovnem nivoju, saj šele tako lahko predvidljivo korakamo skozi podatkovni prostor brez skrbi, da se bomo začeli vrteti v krogu ali bomo nezmožni identificirati pot naprej.

Omejitve modeliranja v relacijskih strukturah pripeljejo do spoznanja, da podatkov o rekurzivnih povezavah ni mogoče obvladati v eni sami tabeli, saj je število korakov vnaprej povsem neznano – števila stolpcev (za zapis nivojev rekurzivnih pod- in nadrejenosti) ne moremo določiti vnaprej. Tako smo rešitev poiskali v sestavljenih relacijskih strukturah, ki neposredno sledijo vzpostavitvi rekurzivne strukture na slovarskem nivoju: z definicijo rekurzivne strukture se vzporedno s podatkovno tabelo vzpostavita še dve metapodatkovni tabeli, v kateri z vsakim vpisom podatka v podatkovno tabelo zapišemo analitične (meta)podatke o rekurzivnih razmerjih.



Slika 10: **Model obvladovanja razmerij v rekurzivnih strukturah**

Tabela R-nadrejeni/podrejeni služi zapisu celotne poti, in sicer z navedbo vsakega posameznega nadrejenega v vrsti, druga (v gornjem primeru nadrejeni/podrejeni) pa je analitična in navaja le pare prvega in zadnjega v vrstah (skupinah) iz prve tabele ter njuno oddaljenost (tj. koliko korakov skozi rekurzivno zanko je bilo opravljenih za povezavo dveh predmetov). Analitična tabela nam omogoča neposredno naslavljanje razmerij iz rekurzivnih struktur in s tem nadaljnje analitične postopke in interpretacije razmerij skozi razrede rekurzivnih povezav.

Ker sta obe tabeli vzpostavljeni in ažurirani povsem avtomatsko, so vezi med njima enosmerne. Povezave so namreč urejane in interpretirane programsko.

4 OD HIERARHIJE DO HETERARHIJ

Pri razvoju podatkovnih rešitev z uporabo platforme Panorama smo ugotovili, da so za dobro uporabniško izkušnjo pri potovanju skozi visoko strukturirane informacijske prostore oddaljene povezave bistvenega pomena. Na abstraktni ravni so to hierarhije, katerim pripadajo posamezni predmeti in so nosilke informacij, ki tem predmetom dodajajo (izvedene) lastnosti oziroma atribute.

Z objektno relacijskim pristopom in zgoraj omenjenimi nadgradnjami smo uspešno razširili vrste struktur, ki jih lahko oblikujemo za zapis dejanskih odnosov med predmeti posameznih vrst v podatkovni model. S tem smo združili podatkovni model in poizvedbe ter odprli prostor za svobodnejši, od dejanskih aplikacij in končnih informacijskih potreb neodvisen pristop k oblikovanju relacijskih baz podatkov. Pri gradnji podatkovnega modela namreč lahko brezkompromisno izhajamo iz dejanskih razmerij v realnosti, ki jo opisuje podatkovni sistem, potem pa znotraj tega verističnega modela oblikujemo izvedene strukture, ki se prilagajajo informacijskim potrebam posamičnih poslovnih procesov oziroma aplikacij.

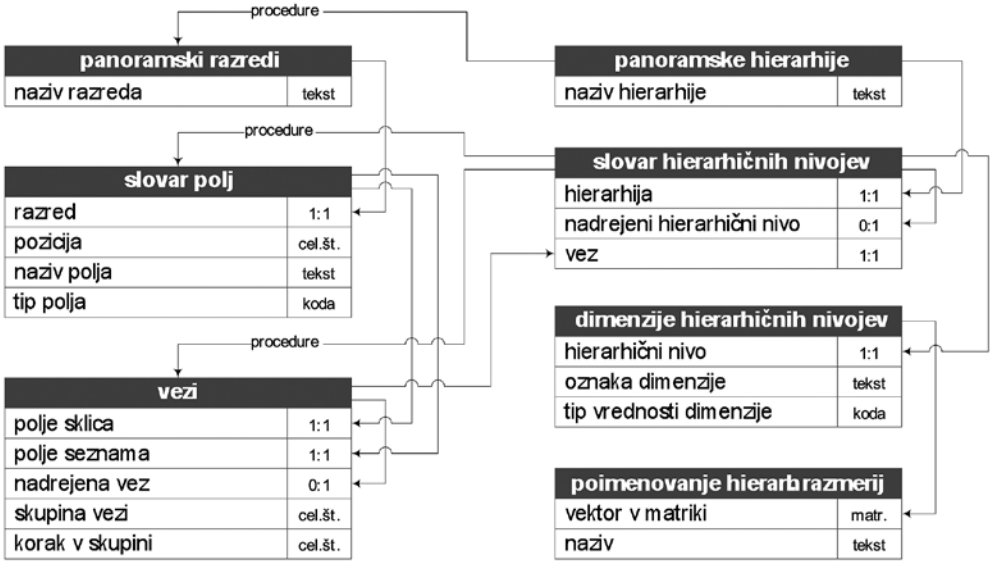
To smo dosegli tako, da smo omogočili definicijo hierarhij, katerim lahko pripada posamezen predmet, ki ima dejansko lastnost povezave prek določene vezi (v smislu vezi iz slovarske datoteke »vezi«; slika 2), s tem pa ta vez pridobiva lastne atribute ter z njimi povezane možnosti njihove interpretacije. Vsaki vezi moramo omogočiti uvrstitev v poljubno mnogo hierarhij oziroma v njihove poljubne kombinacije ali heterarhije.

Preliminarni testi so pokazali, da takšna strategija sicer zagotavlja možnost popolne in pravilne interpretacije vseh medsebojnih razmerij, da pa je lahko precej procesorsko požrešna, kar odpira povsem nova polja hevurističnih pristopov k optimizaciji. Pričakujemo namreč, da bomo morali podpreti dva načina pridobivanja podatkov o hierarhičnih pripadnostih:

- sprotnega, z izvedbo na ukaz ob zahtevi za vpogled v hierarhijo, ter
- preventivnega, ki neprenehoma zagotavlja popolno posodobljenost materializiranega (v)pogleda v namensko datoteko.

V prvem primeru moramo za sprejemljive odzivne čase zagotoviti v kratkem času izvedbo množice ukazov, v drugem pa stalno skrbeti za posodobljenost na eni in dodaten spominski prostor za (z vidika podatkovnega modela podvojene) analitične podatke na drugi strani. Tako poleg samega modela implementacije z dimenzijo hierarhij razširjenih slovarskih datotek razvijamo tudi ustrezne procedure.

Poenostavljena shema razširjenih slovarskih datotek je predstavljena na sliki 11.



Slika 11: **Panoramske tabele slovarskega nivoja z dodanimi vodili hierarhičnih struktur**

5 SKLEP

Panorama je uporabniku skoraj v celoti odprta podatkovna platforma za obvladovanje strukturiranih podatkov. Že danes omogoča oblikovanje visoko kompleksnih podatkovnih struktur ter ponuja precej funkcionalnosti razkrivanja informacij iz kompleksnih podatkovnih modelov.

S predlagano razširitvijo strukture slovarskih datotek in spremljajočimi procedurami bomo omogočili bistveno preglednejše in enostavnejše urejanje hierarhične pripadnosti. Tako bomo uporabnikom še bolj približali sproten vpogled v sestavljene informacije poljubne natančnosti.

Pred implementacijo se bomo posvetili še uporabniškemu vmesniku – preprosta dostopnost in razumljivost je pač edina učinkovita pot do sprejetosti pri uporabnikih. Abstraktnost, ki ga zahteva implementacija heterarhij v relacijskem podatkovnem modelu, k razumljivosti sama zase ne pripomore kaj dosti.

6 VIRI IN LITERATURA

[1] Date, C. J. (2012). Database Design and Relational Theory: Normal Forms and All That Jazz. Sebastopol, CA: O'Reilly Media, Inc.

[2] Lajovic, I. (2002). Urejanje in preiskovanje baz podatkov na asociativni osnovi. Uporabna informatika, 10 (3), 174–178.

[3] Lajovic, T. & Lajovic, I. (2012). Združevanje podatkovnega modela in poizvedb: izziv heterarhij. V Slovensko društvo informatika. (2012), *Ustvarimo nove rešitve! / 19. konferenca Dnevi slovenske informatike – DSI*, Portorož–Ljubljana: Slovensko društvo Informatika. Pridobljeno iz Zbornika predavanj DSI 2012 na CD-ROM-u.

[4] Lavbič, D., Lajovic, I. & Krisper, M. (2010). Facilitating information system development with panoramic view on data, ComSIS, 7, 737–767.

[5] Minsky, M. (1986). The Society of Mind. New York, NY: Simon and Schuster.

▪

Tomaž Lajovic je direktor družinskega podjetja KRES Kreativni sistemi. Pred tem je bil direktor energetske borze BSP SouthPool, potem ko je bil sedem let na mestih svetovalca direktorja energetske borze BSP SouthPool oziroma odgovornega pravnika pri slovenskem organizatorju trga z električno energijo Borzen odgovoren za pravne zadeve in razvoj trgovalnih sistemov v razmerah omejenih prenosnih zmogljivosti. V preteklosti je bil zaposlen kot pravni svetovalec v avdio/video inženiring podjetju TSE, kot specialist za pogodbe in pogajanja v družbi IBM Slovenija ter kot sekretar evropskega združenja borz električne energije EuroPEX.

▪

Iztok Lajovic je lastnik in dolgoletni direktor podjetja KRES Kreativni sistemi. Od samega začetka se ukvarja z razvojem programske opreme za finančno področje in objektno orientiranih podatkovnih baz. Pod njegovim vodstvom je bil že leta 1979 izdelan prvi celoviti in delujoči on-line sistem v gospodarstvu v Jugoslaviji. Zatem je deset let vodil razvoj programske opreme v Ljubljanski banki. Leta 1989 je ustanovil svoje podjetje za razvoj programske opreme, ki živi izključno od lastnega razvoja. Podjetje načeloma ne razvija programske opreme na podlagi naročil, ampak jo razvija na podlagi lastnih vizij za neznanega kupca. Ko se porodi ideja za nov produkt, ga izdelajo od začetne zasnove do končnega izdelka in ponudijo na trg, ki se odzove glede na svoje potrebe in možnostirešitev in informacijskih sistemov.