Vilniaus universitetas
Gamtos mokslų fakultetas
Kartografijos centras

Giedrė Beconytė

DUOMENŲ BAZIŲ PROJEKTAVIMAS

Mokomoji knyga geomokslų specialybių studentams

Vilnius
2012

Aprobuota VU Gamtos mokslų fakulteto tarybos 2012 m. rugsėjo 18 d.

Recenzavo:
Habil. dr. Eimuntas Kazimieras Paršeliūnas
Dr. Antanas Brazauskas
Dr. Olga Suboč
Dr. Albertas Šermokas

Knygos rengimas iš dalies finansuotas Žmonių išteklių plėtros veiksmų programos projekto „Aukštos kvalifikacijos specialistų, atitinkančių valstybės ir visuomenės poreikius biologinių ir žemės gelmių išteklių naudojimo srityje, rengimo tobulinimas (BIOGEONAUDA-A)“ lėšomis.

ISBN 978-609-459-105-1

TURINYS

ĮVADAS

Jei vairuotojams būtų keliami reikalavimai, panašūs į tuos, kurie keliami programuotojams...
Skelbimas: Siūlome darbą vairuotojui.
Reikalavimai: aukštasis išsilavinimas, profesionali darbo patirtis vairuojant troleibusus, tramvajus, traukinius, laivus, funikulierius, metro, ekskavatorius bei atomines elektrines. Ralio varžybų bei lėktuvų dispečerio patirtis būtina. Kandidatas privalo turėti dyzelinių variklių bei turbinų gamybos patirtį. Taip pat būtina turėti „BMW", „Mercedes", „Kamaz" ir „Boeing" inžinieriaus sertifikatus.
www.delfi.lt

Šiais laikais praktiškai bet kurios srities specialistui keliami dideli reikalavimai – jis privalo išmanyti informacines technologijas, dokumentų rengimo taisykles, turėti sistemų projektavimo bei projektų valdymo žinių. Deja, tokios žinios sistemiškai pateikiamos tik nedaugelio specialybių studijų programose. Srityse, kuriose tvarkoma ir naudojama geografinė informacija, sudaromi žemėlapiai, akivaizdus dar didesnis kompetencijų poreikis – jų specialistai turi būti susipažinę ir su geodezija, geografija, kartografija. O kartografija šiandien – tai ne tik žemėlapių ir atlasų sudarymas bei parengimas spaudai, bet ir interaktyvių žemėlapių bei žemėlapių paslaugų Internete kūrimas ir nuolatinis palaikymas, t.y., darbas su geografinių duomenų bazėmis ir jų vaizdais – skaitmeniniais žemėlapiais. Pirmasis šios knygos variantas buvo skirtas būtent kartografijos specialybės studentams, tačiau keleto metų dėstymo patirtis rodo, kad bendrosios žinios apie duomenų modeliavimą, duomenų bazių sudarymą ir jų valdymo principus yra naudingos kiekvienam. Todėl dabartinis tekstas yra išplėstas ir orientuotas ne tik į kartografų, bet ir kitų aplinkos, geografinės informacijos mokslų poreikius.

Knygoje išdėstyta medžiaga apima vieną svarbiausių informatikos aspektų – duomenų bazes, jų projektavimo principus ir teisingą naudojimą. Nors tai nėra knyga apie geografinių duomenų bazes, ji suteikia pagrindines žinias, kurios leidžia lengvai suprasti ir įsigilinti į bet kokių duomenų bazių, tarp jų ir geografinių, specifiką.

Duomenys – tai faktai, kurie gali būti saugomi tam tikru pavidalu: kaip skaičiai, tekstinės eilutės, simboliai ir pan. Duomenys gali būti panaudoti praktiniams tikslams tik suteikus jiems prasmę, t.y., operuojant jais kuriame nors kontekste pagal nurodytas taisykles – taip jie paverčiami informacija. Tam, kad duomenis būtų galima naudoti optimaliai, jie sujungiami į sistemą, vadinamą duomenų baze. Organizacijos duomenų bazė visada yra didesnio struktūrinio vieneto – tos organizacijos informacinės sistemos – dalis. Duomenų bazėje, kurią dabar dažniausiai įsivaizduojame tik skaitmeninę, visų pirma turi būti galima efektyviai rasti ir panaudoti reikiamus duomenis. Galima išskirti pagrindine duomenų valdymo funkcijas, kurias labai palengvina ir naujas galimybes suteikia skaitmeninių duomenų bazių valdymo sistemos:

- duomenų surinkimas ir paruošimas,
- perdavimas,
- įvedimas (kodavimas),
- saugojimas,
- rūšiavimas, paieška ir operacijos su duomenimis (duomenų analizė),
- rezultatų pateikimas galutiniam naudotojui.

Šios knygos pagrindu Vilniaus universitete skaitomo kurso „Duomenų bazių projektavimas" tikslas – supažindinti studentus su svarbiausiomis duomenų bazių sąvokomis, matematiniais pagrindais, duomenų bazių projektavimo principais bei ypatumais ir šiuo metu populiariausia reliacinių duomenų bazių valdymo sistemų technologija. Kursas skirtas magistratūros studentams, išklausiusiems aukštosios matematikos (turintiems diskrečiosios matematikos pagrindus) ir

informatikos pagrindų kursus. Tikimasi, kad skaitytojai aiškiai įsivaizduoja šiuolaikinių kompiuterių sistemų ir dokumentų tvarkymo programų sistemų galimybes, turi supratimą apie geografinio objektų geometrijos ypatumus, žemėlapių sudarymo principus.

Duomenų bazių projektavimas yra svarbi informacijos valdymo dalis bet kokiuose projektuose, todėl gautos žinios ir įsisavinta technologija (reliacinės duomenų bazių valdymo sistemos projektavimas ir kūrimas *MS Access* programų sistemos pavyzdžiu) bus naudinga tolesnėse studijose ir praktiniame darbe. Perskaitę šią knygą studentai turės išsamų supratimą apie reliacinį duomenų modelį, mokės sukurti savo reliacinę duomenų bazę pagal sudarytą koncepcinį modelį, ją sutvarkyti, optimizuoti, suformuluoti nesudėtingas užklausas matematiškai, jas užrašyti SQL kalba ir įvykdyti.

Išdėstyta medžiaga apima tarpusavyje susijusias temas, kurių kiekvienai skirta po vieną skyrių. Kai kuriuose skyriuose minimi dalykai, išdėstyti tolesniuose skyriuose, kur apie juos galima paskaityti plačiau. Tais atvejais, kai tai nėra akivaizdu, tekste yra nuorodos į atitinkamą knygos vietą.

Pirmajame skyriuje aptariama duomenų, informacijos ir duomenų bazės samprata, pateikiama trumpa duomenų bazių valdymo sistemų raidos apžvalga, aptariami pagrindiniai duomenų tipai. Plačiau nagrinėjami geografiniai duomenys, jų modeliai, geografinės informacijos sistemos. Supažindinama su kartografiniais duomenimis ir išplėstai paaiškinama metaduomenų samprata. Šią temą tęsia antrasis skyrius, kuriame nagrinėjama, kaip keitėsi duomenų bazės samprata ir funkcijos geografijoje, kokias svarbiausias naujoves kartografijoje atnešė plintančios skaitmeninės technologijos. Aptariamos svarbiausios Lietuvos geografinių duomenų bazės. Trečiasis skyrius skirtas duomenų ir duomenų bazės vietai organizacijos struktūroje ir veikloje, strateginiams duomenų naudojimo klausimams. Šiuose trijuose skyriuose metodinės priemonės medžiaga yra glaudžiai siejama su geografijos mokslu.

Ketvirtajame skyriuje skaitytojai supažindinami su koncepcinio duomenų modeliavimo metodika, kuri yra duomenų bazių projektavimo pagrindas. Šiame skyriuje pristatomos pagrindinės esybių ryšių modeliavimo sąvokos ir pateikiami jo taikymo įvairiose srityse pavyzdžiai. Penktajame skyriuje pristatomi dažniausiai pasitaikantys duomenų bazių modeliai.

Likusioje knygos dalyje pristatoma bendroji duomenų bazių projektavimo teorija, kuri tiesiogiai nesiejama su jokia taikomąja sritimi, tačiau kurią būtina išmanyti siekiant profesionaliai taikyti informacijos technologijas ir duomenų bazių valdymo sistemas ten, kur tenka kaupti, tvarkyti ir analizuoti duomenis. Dėstant fundamentalią reliacinių duomenų bazių teoriją ir kai kuriuos papildomus duomenų bazių projektavimo aspektus, ypač 7, 8 ir 9 skyriuose remtasi C.J. Date knyga „An Introduction to Database Systems“, išleista 2003 metais (8-tasis leidimas). Šeštajame skyriuje aptariamos duomenų bazių teorijos pagrindinės sąvokos ir sistemos architektūros pagrindai. Septintajame skyriuje išsamiai aprašytas reliacinis modelis – pateiktas matematinis modelio pagrindimas, manipuliavimo duomenimis funkcijos, bei išdėstyta duomenų vientisumo samprata. Aštuntajame skyriuje nagrinėjamos konkrečios pagrindinės struktūrizuotos užklausų kalbos SQL operacijos su pavyzdžiais. Devintasis skyrius apima svarbią duomenų bazių projektavimo temą – duomenų bazių norminimo metodiką. Likusiuose skyriuose aptariami duomenų lygiagretaus naudojimo ir paskirstymo klausimai, duomenų sauga ir papildoma informacija, svarbi, norint geriau suprasti reliacinio modelio matematinį pagrindą: duomenų klasifikavimo principai, įvadas į predikatų logiką, pagrindinės aibių sąvokos ir aibių algebros operacijos.

Kiekvieno skyriaus pabaigoje yra klausimai, leidžiantys pasitikrinti, ar gerai įsisavinta teorinė medžiaga, taip pat įvairaus pobūdžio savarankiško darbo ir praktinės darbo su duomenų bazėmis

užduotys. Didžioji praktinių užduočių dalis gali būti atliekama kompiuteriu naudojant *MS Access* ar kitą duomenų bazių valdymo programą. Kaip ir daugumoje informacinių technologijų sričių, literatūros apie duomenų bazių sistemas gausa yra tiesiog neįtikėtina, todėl neįmanoma apžvelgti net ir fundamentalių veikalų šia tema. Knygos pabaigoje pateiktas rekomenduojamos papildomos literatūros sąrašas bei kelios svarbiausios Interneto nuorodos.

Nuoširdžiai dėkoju visiems, padėjusiems parengti šią mokomąją knygą, ypač recenzentams iš Vilniaus universiteto ir Vilniaus Gedimino technikos universiteto už pastabas, labai padėjusias pagerinti jos kokybę, taip buvusiems studentams ir kolegoms iš VĮ „GIS-Centras“, kurių patarimų dėka knyga tapo įdomesnė ir išsamesnė. Tikiuosi, kad jos skaitytojais taps ne vien būsimieji geografai ir kartografai, bet ir kitų specialybių studentai. Juk nuo to, ar tinkamai kaupiame ir tvarkome duomenis, priklauso bet kokio tyrimo kokybė!

Knygoje naudojamos santrumpos

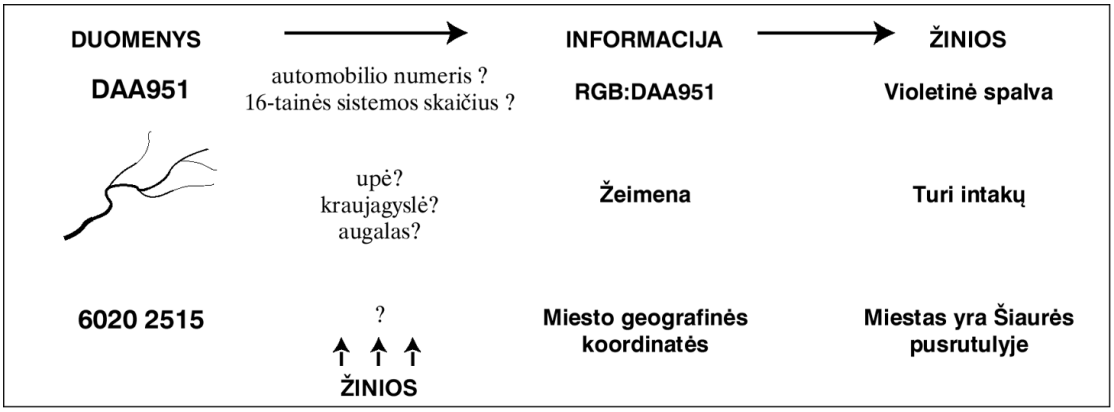
CK – potencialus raktas (angl. *Candidate Key*);
DB – duomenų bazė;
DBVS – duomenų bazių valdymo sistema;
DP – daugiareikšmė priklausomybė;
ER – esybių ryšių (angl. *Entity-Relationship*) modelis;
FK – išorinis raktas (angl. *Foreign Key*);
FP – funkcinė priklausomybė;
GDB – geografinių duomenų bazė;
GDBVS – geografinių duomenų bazių valdymo sistema;
GI – geografinė informacija;
GIS – geografinės informacijos sistema;
GPS – globali padėties nustatymo sistema;
ID – (unikalus) objekto identifikatorius;
IS – informacinė sistema;
LEI portalas – Lietuvos erdvinės informacijos portalas *www.geoportal.lt*;
NF – norminė forma (angl. *Normal Form*);
NMDP – nacionalinis metaduomenų profilis – faktinis Lietuvos metaduomenų standartas;
PDB – paskirstyta duomenų bazė;
PDBVS – paskirstyta duomenų bazių valdymo sistema;
PK – pirminis raktas (angl. *Primary Key*);
RDB – reliacinė duomenų bazė;
RDBVS – reliacinė duomenų bazių valdymo sistema;
SP – sąjungos priklausomybė;
SQL – struktūrizuota užklausų kalba (angl. *Structured Query Language*).

I. DUOMENYS, GEOGRAFINIAI DUOMENYS IR METADUOMENYS

Labai liūdna, kad mūsų dienomis tiek mažai nenaudingos informacijos.
Oscar Wilde

I.1 DUOMENYS IR DUOMENŲ BAZĖS

Duomenys – tai faktai, egzistuojantys nepriklausomai nuo to, kaip jie interpretuojami, pavyzdžiui, simboliai ar piešinys I-1 paveiksle. Be paaiškinimo tokie faktai iš esmės nieko nereiškia. Tačiau galima pastebėti, kad kartais, jei duomenų kiekis pakankamai didelis ar jie kokiu nors būdu struktūrizuoti, kiekvienas žmogus remdamasis savo sukauptomis žiniomis, patirtimi gali pasiūlyti vienokią ar kitokią duomenų interpretaciją. Vis dėlto, duomenys gali būti tinkamai panaudoti tik tada, kai yra pateikta tiksli, aiški ir išsami papildoma informacija, leidžianti juos suprasti.



I-1 pav. Duomenys ir informacija

Taigi, duomenys patys savaime vertės dar neturi. Tam, kad būtų galima panaudoti sukauptus duomenis, reikia sugebėti juos interpretuoti, t.y., suteikti jiems prasmę. Pavyzdžiui, tas pats skaičius 2398297 gali būti interpretuojamas kaip telefono numeris, pinigų suma sąskaitoje ar studijų knygelės numeris. Interpretuoti duomenys vadinami **informacija**. Tai lotyniškos kilmės žodis, reiškiantis kam nors suteiktą „formą“, idėją, mintį. Be informacijos negali vykti jokia sąmoninga ir organizuota veikla; ji yra pagrindas ir priemonė **žinioms** kaupti. Tai didžiausias žmonijos turtas.

Procesas, kurio metu informacija virsta žiniomis, nėra visiškai ištirtas. Šiuo metu formuojasi naujas požiūris į duomenis, pagrįstas jų tiriamąja analize (angl. *data mining*), kuri suprantama kaip metodas, procesas ar algoritmo taikymas aptikti erdviniams ryšiams bei dėsningumams abstrakčiuose (taip pat geografiniuose) duomenyse ir juos suprantamai pavaizduoti. Taip iš pirmo žvilgsnio tarpusavyje nesusiję duomenys yra sutvarkomi tokiu būdu, kad būtų atskleistas erdvėje egzistuojantis reiškinys ir sukurtos naujos žinios. Dažnai naujų žinių kūrimas iš geografinių duomenų yra paremtas kartografinio ar kitokio vaizdo (vizualizacijos) panaudojimu, nes ypač kartografinis vaizdas pasižymi dideliu ir dar ne iki galo suvoktu interpretavimo potencialu. Tiriamosios analizės tikslas yra visiškai išnaudoti turimus duomenis ir juos interpretuojant papildyti žinių bazę.

Dauguma duomenų žemiausiu lygmeniu yra gana paprastos struktūros – t. y., vieno iš pagrindinių duomenų tipų reikšmės arba iš tokių reikšmių sudaryti rinkiniai. Duomenų tipai yra **pirminiai**,

pavyzdžiui, skaičiai, tekstinės eilutės, ir **sudėtiniai**, dar vadinami duomenų struktūromis, kurie gaunami įvairiais būdais jungiant pirminius duomenų tipus. Sudėtinių tipų pavyzdžiai yra lentelės, sąrašai, medžiai ir pan. Daugiau informacijos apie duomenų tipus pateikiama kitame poskyryje.

Neįmanoma saugoti visos turimos informacijos. Vienas iš duomenų bazių projektuotojų uždavinių yra atrinkti, kuri informacija yra svarbiausia, kurią yra prasminga perkelti į duomenų bazę. Tinkama informacija turi pasižymėti keliomis savybėmis.

1. Reikšmingumas. Informacija turi būti reikalinga konkrečiam tikslui pasiekti.
2. Aktualumas. Informacija turi būti nepasenusi, neperteklinė bei gauta iš šaltinio, tinkamo numatytam uždaviniui spręsti.
3. Tikslumas. Informacija neturi būti klaidinga ar dviprasmiškai interpretuojama.
4. Išsamumas. Turi būti saugomi visi reikalingi faktai apie objektus ar reiškinius bei jų tarpusavio sąsajos.
5. Pasiekiamumas. Informaciją prireikus turi būti galima rasti ir gauti reikiamu pavidalu.
6. Patikrinamumas. Turi būti galima patikrinti ir įvertinti informacijos kokybę.
7. Efektyvumas. Informacijos surinkimo, tvarkymo ir atnaujinimo kaštai turi atitikti jos teikiamą (ar potencialią) naudą įmonei ar organizacijai, svarbą priimant sprendimus.

Kol kas apibrėšime, duomenų bazę (angliškas terminas *database* pirmą kartą panaudotas 1963 metais) kaip rinkinį duomenų elementų (faktų), kurie saugomi kompiuterio atmintyje susisteminti taip, kad kompiuterinės programos galėtų efektyviai gauti reikiamus duomenis **užklausai** (klausimui apie duomenis, užrašytam sutartiniu kodu) atsakyti. Atsakymas interpretuojamas kaip informacija, kuri naudojama sprendimams priimti. Sistema ar programa, naudojama duomenims tvarkyti ir užklausoms vykdyti, vadinama **duomenų bazių valdymo sistema (DBVS)**. Duomenų bazė kartais suprantama tik kaip fizinė duomenų saugykla. Atsiribojus nuo interpretavimo, duomenų saugykla yra grynai informatikos tyrimų objektas, kuris nagrinėjamas saugomų duomenų pasiekimo greičio, fizinės apimties, saugumo ar panašiais technologiniais aspektais. Tačiau platesnė duomenų bazės samprata, kuria toliau vadovausimės, apima ne tik saugomus faktus, bet ir papildomą informaciją, kuri leidžia tuos faktus interpretuoti, t.y., susieti su realaus pasaulio objektais ar reiškiniais. Taip suprantama duomenų bazė paprastai nagrinėjama keliais pagrindiniais aspektais:

- duomenų surinkimas, paruošimas ir įvedimas,
- duomenų saugojimas ir perdavimas,
- duomenų rūšiavimas, paieška ir įvairūs kiti veiksmai su duomenimis,
- rezultatų pateikimas.

Duomenų bazių valdymo sistemos paprastai naudojamos įvairių **taikomųjų programų**, kurių daugelis pasiekiama naudotojui Internetu.

Dar viena svarbi su duomenimis ir duomenų bazėmis siejama sąvoka yra **metaduomenys** – duomenys apie duomenų bazę ar rinkinį. Ši duomenų rūšis aprašyta I.7 poskyryje.

I.2 SVARBIAUSI DBVS RAIDOS BRUOŽAI

Pirmosios DBVS buvo pradėtos vystyti apie 1960-tuosius metus. Šios srities pradininku laikomas amerikiečių mokslininkas Čarlzas Bachmanas (Charles Bachman), kuris savo darbuose pagrindė poreikį efektyviau naudoti atsirandančias naujas technines duomenų saugojimo priemones ir sukūrė tinklinę duomenų bazę IDS (angl. *Integrated Data Store*), maksimaliai išnaudojančią to meto

techninės įrangos galimybes. Bachmano idėjų pagrindu CODASYL konsorciumas išvystė tinklinį duomenų bazės modelį, panaudotą IDMS (angl. *Integrated Database Management System*) sukurti. Maždaug tuo pačiu metu *Rockwell* kompanija kūrė hierarchinį modelį, vėliau tapusį IBM sukurtos Informacijos valdymo sistemos IMS (angl. *Information Management System*) pagrindu. IDMS ir IMS buvo pirmosios reikšmingos duomenų bazių valdymo sistemos. Kiek vėliau buvo sukurtos ir kitos panašios sistemos, kurios vėliau išsivystė į operacines sistemas, programavimo kalbas ar šiuolaikines duomenų bazių valdymo sistemas. 1970-aisiais IBM dirbęs britų mokslininkas Edgaras Kodas (Edgar Frank Codd) straipsnyje "Reliacinis duomenų modelis dideliems bendro naudojimo duomenų bankams" ("A Relational Model of Data for Large Shared Data Banks") pasiūlė **reliacinį duomenų modelį**, tapusį mums įprastų DBVS pagrindu.

DB termino atsiradimas sutapo su tiesioginės prieigos (angl. *direct access*) laikmenomis – magnetiniais būgnais (http://en.wikipedia.org/wiki/Drum_memory), HDD, duomenų ląstelėmis (angl. *data cells*, http://en.wikipedia.org/wiki/IBM_2321_Data_Cell). Tiesioginė prieiga užtikrino interaktyvaus ir bendro naudojimo galimybę.
Pirmosios DB buvo navigacinės (http://en.wikipedia.org/wiki/Navigational_database)

20 a. devintajame dešimtmetyje moksliniai tyrimai DBVS srityje koncentravosi ties **paskirstytų duomenų bazių** problema, o amžiaus pabaigoje paplito objektinis požiūris ir **objektinės duomenų bazės**, taip pat vis labiau naudojamos erdvinių duomenų bazės. Pirmieji **atviro kodo** duomenų bazių produktai, tokie kaip objektinė *PostgreSQL* ir reliacinė *MySQL*, taip pat sukurti 20 a. pabaigoje.

21-ajame amžiuje duomenų bazių tyrimų akcentas yra taip vadinamos **XML duomenų bazės**, kuriose bandoma apjungti į vieną sistemą anksčiau buvusias dvi skirtingas klases – dokumentus ir duomenis, taip palengvinant keitimąsi informacija tarp šių klasių. Vis svarbesnis tampa labai didelių duomenų bazių panaudojimas išaiškinti tiesiogiai nesuvokiamiems ryšiams tarp duomenų, kurie padeda priimti geresnius verslo ir valdymo sprendimus (**duomenų gavyba**). Duomenų gavybos technologija naudojama keliose komercinėse DBVS, tokiose kaip *Oracle*, *IBM DB2* ir *Microsoft SQL Server*. Duomenų bazių valdymo sistemos, kurios jungia duomenų tvarkymo ir jų naudojimo sprendimams priimti priemones vienoje aplinkoje, naudojant paskirstytas duomenų bazes, bendradarbiaujant, dažnai vadinamos **induktyviosiomis duomenų bazėmis**.

I-1 lentelė. Svarbiausių RDBVS gamintojų pelnas pasaulyje 2006–2013 metais, mln. JAV dolerių[1]

Gamintojas	**2006**	**2010**	**2011**	**2013**	**Rinkos dalis 2011 m. (%)**	**Rinkos dalis 2013 m. (%)**
Oracle	7166	9990	11787	11200	48.8	49
IBM	3500	4300	4870	12000	20.2	20
Microsoft	3052	3641	4100	20800	17.0	17
Teradata	558	755	882	2692	3.7	3
SAP/Sybase	492	744	1101	2800	4.6	4
Kiti					5.8	

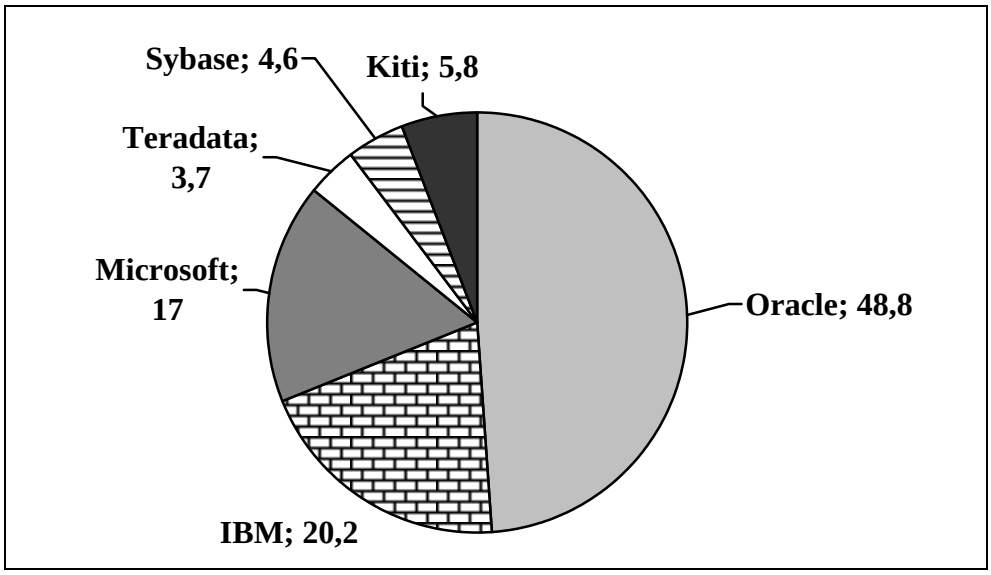
Labiausiai paplitusių reliacinių DBVS rinka 2009 metais sudarė 763 milijardus JAV dolerių. Paveiksle (I-2 pav.) parodyta, kokią šios rinkos dalį užėmė pagrindiniai gamintojai. Pateikti 2008 m.

[1] Pagal *IDC Market Scape*, http://www.idc.com.

duomenys; 2009 m. *Oracle* dalis jau sudarė 48%, o *Sybase* aplenkė *Teradata*. Pabrėšime, kad DBVS atveju (duomenys – brangiausias organizacijos turtas, todėl labai daug investuojama į jų saugumo užtikrinimą ir efektyvų naudojimą) komerciniai produktai kol kas nepalyginamai lenkia atviro kodo sistemas, kurių yra sukurta gana daug (2011 m. balandį *Vikipedija* skelbė 75 produktų sąrašą).

Oracle korporacija pirmąją taip pat pavadintą komercinę reliacinę duomenų bazių valdymo sistemą išleido 1979 m. Tai pirmoji komercinė SQL pagrįsta ir su ANSI SQL standartu suderinama DBVS. *Oracle* korporacija pirmoji sukūrė paskirstytų duomenų bazių technologiją (1986 m.) ir Interneto duomenų bazę (1997 m.), bei XML palakančią duomenų bazę (1999 m.). Šiuo metu tai labiausiai paplitusi DBVS, beveik visur naudojama bankų, finansinėse ir mokslinėse sistemose duomenims saugoti, apdoroti ir analizuoti, išleista 63 kalbomis. 2007 m. *Oracle* korporacija po *Microsoft* ir *IBM* buvo trečioji pagal dydį programinės įrangos gamintoja pasaulyje.

Sybase yra antrosios pagal reikšmę DBVS sistemos kūrėja po *Oracle*. *Sybase IQ* reliacinis produktas – į stulpelius orientuota DBVS, kurioje duomenys saugomi stulpeliais, todėl daug efektyvesnės duomenų skaitymo operacijos, tačiau lėtesnės atnaujinimo operacijos. Pagal Gineso rekordų knygą, 2008 m. gegužės mėn. *Sybase IQ* technologija remiasi didžiausias pasaulyje kada nors buvęs duomenų bankas (1000 terabaitų – petabaitas – virš trilijono eilučių bendros duomenų apimties). *Sybase IQ* naudojama unikali duomenų suspaudimo technologija, leidžianti taupyti didžiulius kiekius vietos duomenų saugyklose, o taip pat saugoti aplinką, nes mažiau energijos naudojama aušinimui ir pan. *Sybase* yra ir mobilių duomenų bazių sistemų lyderė. Šiuo metu tai SAP (angl. *Systems Applications Products in Data Processing*) korporacijos dalis.



I-2 pav. Pagrindiniai komercinių RDBVS gamintojai ir jų rinkos dalis procentais (2011 m.).

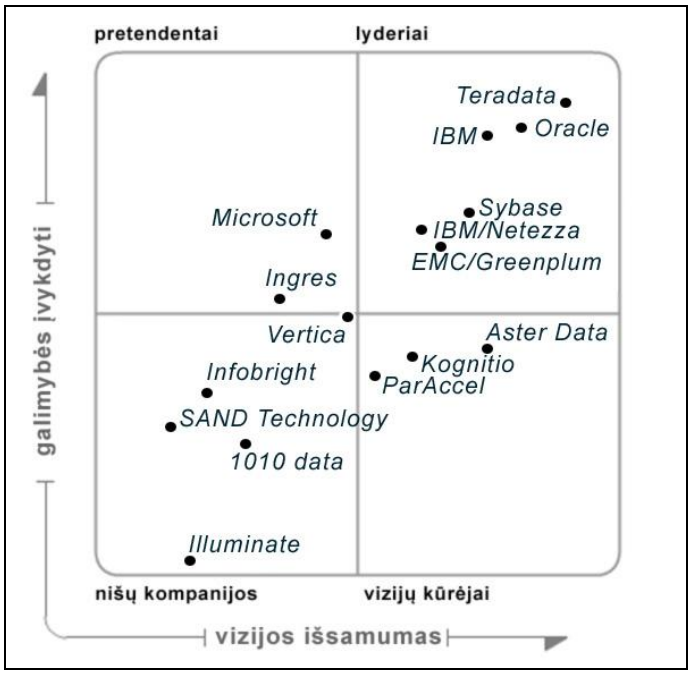
IBM DB2 duomenų bazių valdymo sistema yra viena seniausių DBVS, šiuo vardu žinoma nuo 1983 metų, sukurta dalyvaujant dar reliacinio DBVS modelio kūrėjui E. F. Kodui. Vis dėlto IBM neįvertino Kodo idėjų ir kuriant sistemą buvo pažeista keletas fundamentalių teorinio modelio principų. Vis dėlto rezultatas – SEQUEL, vėliau tapusi SQL kalba – tapo pagrindine DBVS užklausų kalba. 2001 m. IBM nusipirko *Informix* ir inkorporavo jų technologiją į savo produktą, kuris dabar yra objektinės-reliacinės DBVS pavyzdys. *Oracle* ir *IBM* dažnai derina savo produktus ir technologijas (reiškinys, angliškai vadinamas *coopetition – cooperation* / *competition*).

Microsoft SQL Server duomenų bazių valdymo sistema sukurta 1989 metais *Microsoft* korporacijos kartu su *Sybase*, kurios konkurentu ji vėliau tapo (atsiradus *Windows NT* versijai *Microsoft* pradėjo šį produktą vystyti atskirai, jis ir dabar dirba tik su *Windows* operacine sistema) .

Teradata korporacija taip pat įsikūrusi JAV ir turi daugiau kaip 1000 klientų, tarp kurių yra *Wal-Mart* parduotuvių tinklas, *AT&T* telekomunikacijų kompanija, Amerikos bankas, *Coca Cola, FedEx* pervežimų kompanija ir kiti. 1996 m. *Wal-Mart Teradata* duomenų bankas (angl. *warehouse*) apėmė 11 terabaitų ir buvo didžiausias pasaulyje, o iki 1999-ųjų *Teradata* kūrė didžiausią pasaulyje paskirstytą duomenų bazę, apimančią 130 terabaitų duomenų ir 176 mazgus. Išskirtiniai *Teradata* bruožai – sprendimų priėmimo palaikymas ir *Shared-Nothing* paskirstytų DBVS architektūra (itin lengvai plečiama savarankiškų mazgų architektūra, garantuojanti, kad pridėjus naujus procesorius, efektyvumas padidės tiesiškai). Ši sistema gali efektyviai apdoroti daug ir sudėtingų skirtingų klientų užklausų vienu metu. Teigiama, kad sukurti 100 000 įrašų reikia tiek pat sąnaudų, kiek norint sukurti šimtą.

Tarp vidutinio galingumo komercinių DBVS, paprastai turinčių išvystytą naudotojo sąsają, galima paminėti *FileMaker Pro* ir *Microsoft Office Access* sistemas. Tokios DBVS yra skirtos įmonėms ar organizacijoms, kurių duomenų bazės yra palyginti nedidelės, nepaskirstytos erdvėje, turi nedaug lygiagrečiai dirbančių naudotojų.

Svarbi sąvoka, susijusi su DBVS yra **duomenų saugykla** (angl. *data warehouse*). Duomenų saugykla – konkrečios paskirties, vientisas duomenų rinkinys, kuriame saugomi apibendrinti duomenys, ataskaitos, analizių duomenys, reikalingi duomenų gavybai, greitai verslo analizei, rinkos tyrimams ir sprendimams priimti. Duomenų saugyklos pagrindinė paskirtis – gauti, transformuoti, perduoti ir analizuoti duomenis ir metaduomenis. Nuo įprastos duomenų bazės duomenų saugykla skiriasi tuo, kad joje saugomi duomenys surenkami iš daugelio šaltinių, įskaitant įvairius automatinius daviklius, apdorojami, susiejami ir apibendrinami siekiant sąveikumo. Be to, duomenų saugyklose kiekvienas įrašas turi laiko žymą, kuri rodo laiko momentą kada įrašas yra gautas. Duomenų saugykloje duomenys nekinta, tai reiškia, kad nauji duomenys įrašomi, juos pridedant prie jau esamų duomenų, o ne juos pakeičiant. Taigi, duomenų saugykloje kaupiami dalykinės srities istoriniai duomenys. 2011 m. sausio pabaigoje pateiktas *Gartner* informacinių technologijų analitikų atliktas duomenų saugyklų sprendimų vertinimas (I-3 pav.) atspindi DBVS gamintojų lyderystę ir inovatyvumą šioje svarbioje srityje.



I-3 pav. Duomenų saugyklų DBVS „magiškas kvadratas“[1].

Duomenų saugyklų DBVS turi palaikyti labai dideles ir labai mažas duomenų bazes, įvairius duomenų tipus, skirtingą techninę įrangą ir operacines sistemas, operacijas su duomenimis, kurių aibė nuolat kinta ir didėja bei reikalauja nuolatinio DBVS administravimo. Duomenų saugyklų DBVS turi efektyviai valdyti tokias apkrovas, kurios neprognozuojamai kinta nuo labai didelių iki labai mažų, bei turėti ypač kokybiškas duomenų apsaugos ir atkūrimo priemones.

Reikia pabrėžti, kad duomenys praktiškai visada gyvuoja ilgiau negu jų valdymui naudojama programinė įranga. Dažnai programinė įranga būna naudojama keletą ar keliolika metų, o po to ją pakeičia kita, geresnė ar populiaresnė. Kiekvieną kartą keičiant DBVS programinę įrangą, gali tekti iš naujo kurti struktūras visiems saugomiems duomenims. Taip atsitinka kai skirtingos programų sistemos naudoja skirtingas duomenų struktūras. Duomenų perkėlimas iš vienos programinės įrangos į kitą dažniausiai yra ilgas ir sudėtingas procesas, todėl visaip siekiama jo išvengti. Šiuo metu taip nutinka rečiau, nes kuriamos specialios duomenų perkėlimo tarp skirtingų DBVS priemonės, tačiau visiškai išvengti papildomo darbo neįmanoma.

I.3 PAGRINDINIAI DUOMENŲ TIPAI

Visi duomenys skaitmeniniuose elektroniniuose įrenginiuose žemiausiu lygmeniu vaizduojami kaip **bitų** (angl. *Binary Digit*) rinkiniai. Bitas yra elementas, galintis turėti tik dvi (0 ir 1) reikšmes. Mažiausias adresuojamas duomenų vienetas yra **baitas**, sudarytas iš 8 bitų. Programų kodo instrukcijos apdoroja **žodžius** (angl. *word*), kurių ilgis skiriasi, o šiuo metu paprastai yra 32 arba 64 bitai. Dauguma atvejų žodis interpretuojamas kaip dvejetainis skaičius, pavyzdžiui, 32 bitų žodis gali vaizduoti nuo 0 iki 2^{32} -1 sveikųjų reikšmių be ženklo arba nuo -2^{31} iki 2^{31} -1 sveikųjų reikšmių su ženklu (beje, atlikti operacijoms su ženklais naudojamas įdomus dvejetainio papildinio

[1] Pagal *Gartner's global research organization . Magic Quadrant for Data Warehouse Database Management Systems*. http://www.gartner.com/technology/media-products/reprints/teradata/vol3/article1/article1.html

(angl. *two's complement*) metodas. Specifinės aritmetinės instrukcijos gali interpretuoti tą patį žodį kaip slankaus kablelio formato (angl. *floating point*) skaičių.

In the earliest non-electronic information processing devices, such as Jacquard's loom or Babbage's Analytical Engine, a bit was often stored as the position of a mechanical lever or gear, or the presence or absence of a hole at a specific point of a paper card or tape. The first electrical devices for discrete logic (such as elevator and traffic light control circuits, telephone switches, and Konrad Zuse's computer) represented bits as the states of electrical relays which could be either "open" or "closed". When relays were replaced by vacuum tubes, starting in the 1940s, computer builders experimented with a variety of storage methods, such as pressure pulses traveling down a mercury delay line, charges stored on the inside surface of a cathode-ray tube, or opaque spots printed on glass discs by photolithographic techniques.

In the 1950s and 1960s, these methods were largely supplanted by magnetic storage devices such as magnetic core memory, magnetic tapes, drums, and disks, where a bit was represented by the polarity of magnetization of a certain area of a ferromagnetic film, or by a change in polarity from one direction to the other. The same principle was later used in the magnetic bubble memory developed in the 1980s, and is still found in various magnetic strip items such as metro tickets and some credit cards.

In modern semiconductor memory, such as dynamic random access memory or flash memory, the two values of a bit may be represented by two levels of electric charge stored in a capacitor. In programmable logic arrays and certain types of read-only memory, a bit may be represented by the presence or absence of a conducting path at a certain point of a circuit. In optical discs, a bit is encoded as the presence or absence of a microscopic pit on a reflective surface. In one-dimensional bar codes, bits are encoded as the thickness of alternating black and white lines.

Duomenų tipas – tai rinkinys galimų reikšmių, kurias gali įgyti duomenų elementas (t.y., to tipo kintamasis). Pavyzdžiui, numeris paprastai būna sveikas neneigiamas skaičius, didesnis už nulį: 1, 2, 3, 4 ir t.t. Tipo aprašymas gali būti ir labai paprastas (išvardinama baigtinė reikšmių aibė), ir sudėtingas. Tipą galima laikyti duomenų savybe, informacija, kuri naudotojui pasako, kokie tai duomenys, ir (netiesiogiai, o abstraktiems duomenų tipams – tiesiogiai) kokias operacijas su jais galima atlikti.

Duomenų tipo sąvoka susijusi su kompiuterių programavimu ir pirmą kartą pradėta naudoti aukšto lygio programavimo kalbose, kai atsirado poreikis naudoti struktūrizuotus duomenis. Kintamųjų tipizavimas pirmą kartą įvestas *Fortran* kalboje (1953 m.), kurioje naudojami skaitiniai ir duomenų masyvo tipai. Su *Algol-60* buvo išplėstas masyvų panaudojimas neribojant masyvų matmenų. Dar vėliau įvesti tipai simbolių eilutei, įrašui saugoti. *PL/1* (1965 m.) jau buvo galimybė laisvai kurti sudėtingas duomenų struktūras, naudoti nuorodos tipą. Simula-67 – pirmoji eksperimentinė kalba, kurioje įvesta klasės sąvoka. Apie 1970 metus *Algol* ir *Pascal* kalbose tipams jau buvo suteikiami vardai, jie sisteminami. Šiuo metu duomenų tipai išreikštai naudojami praktiškai visose programavimo kalbose, o pagrindiniai paprasti skaitiniai ir tekstiniai duomenų tipai suprantami vienodai, nors gali būti vadinami skirtingai. Dauguma kalbų leidžia apibrėžti naujus duomenų tipus, kurie paprastai sudaromi apjungiant įvairų skaičių kitų tipų elementų, ir aprašyti operacijas su

naujais tipais. Tokio naujo tipo pavyzdys gali būti „Knyga“ – rinkinys, sudarytas iš keleto tekstinių eilučių, kuriose saugomas knygos pavadinimas, autoriaus pavardė, leidimo vieta ir ISBN kodas, bei sveiko skaičiaus – leidimo metų. Tipų sistema leidžia automatiškai kontroliuoti vykdomų operacijų korektiškumą.

Aštuntajame 20 a. dešimtmetyje mokslininkai Entonis Horas (C.A.R. Hoare) ir Barbara Liskov (Barbara Liskov) suformulavo sąrašą savybių, kurias turi tenkinti duomenų tipai:

- vienas duomenų tipas nusako aibę reikšmių, kurias gali įgyti kintamasis ar reiškinys;
- kiekviena reikšmė gali priklausyti vienam ir tik vienam duomenų tipui;
- nepriklausomai nuo reikšmių, kintamojo ar reiškinio tipą galima nustatyti iš konteksto arba iš kintamojo ar reiškinio pavidalo;
- kiekvienos operacijos operandų ir rezultato tipai yra fiksuoti;
- vienodais simboliais gali būti žymimos operacijos su skirtingais tipais. Tada jos laikomos daugiareikšmėmis ir interpretuojamos skirtingai (pavyzdžiui, aritmetinė sudėtis „+“: 1+2=3 ir eilučių sujungimas „1“+„2“=„12“);
- duomenų tipo reikšmių savybės ir su reikšmėmis atliekamų operacijų savybės apibrėžiamos aksiomomis;
- abstrakčiųjų duomenų tipų operacijos turi atitikmenis matematikoje;
- duomenų tipo aprašas turi apimti visas to tipo reikšmėms leistinas operacijas;
- naudotojas neprivalo žinoti, kaip reikšmės vaizduojamos kompiuterio atmintyje;
- naudotojas su tipo reikšmėmis gali atlikti tik apibrėžtas to tipo operacijomis, o ne tiesiogiai operuoti reikšmių vaizdais kompiuterio atmintyje.

Iš pradžių duomenų tipai naudoti tik kaip reikšmių aibės, Devintajame 20 a. dešimtmetyje pradėti naudoti **abstraktieji duomenų tipai** (ADT), kurie apibrėžia ne tik galimas reikšmes, bet ir galimas operacijas su tomis reikšmėmis. Pavyzdžiui, vienas iš dažniausiai naudojamų duomenų tipų *Integer* (sveikas skaičius), *Java* programavimo kalboje žymimas „int“, reiškia 32 bitais (4 baitais) koduojamų sveikųjų skaičių aibę nuo -2147483648 iki 2147483647, o taip pat tai, kad su šio tipo reikšmėmis galima atlikti sudėties, atimties ir daugybos veiksmus. Sudėtinis tipas, aprašantis spalvą, koduotą, pavyzdžiui, RGB modelyje, užima tris baitus, kurių kiekvienas skirtas atitinkamai raudonai R (angl. *Red*), žaliai G (angl. *Green*) ir mėlynai B (angl. *Blue*) spalvos dedamajai. Viename baite saugomas sveikas skaičius nuo 0 iki 255, reiškiantis dedamosios intensyvumą. Su spalvomis galima atlikti sudėties, atimties, invertavimo operacijas, bet negalima daugyba. Be to, šioje struktūroje dar gali būti skiriama vietos tekstinių simbolių eilutei, kurioje saugomas spalvos pavadinimas. Tuo atveju leistinų operacijų aibė apribojama. Gali būti, kad vienintelė leistina operacija su duomenų tipu yra reikšmės priskyrimas.

Pagal sudėtingumą duomenų tipai skirstomi į paprastuosius ir struktūrinius. Paprastųjų tipų reikšmės yra sąlyginai nedalomos, kaip, pavyzdžiui, sveikas skaičius ar loginis tipas. Struktūriniai tipai yra sudaryti iš kelių paprastųjų ar kitų struktūrinių tipų elementų, kaip anksčiau minėtuose „Knygos“ ir RGB spalvos pavyzdžiuose.

Pagrindiniai paprastieji tipai:

- loginis tipas – paprasčiausias duomenų tipas, turintis dvi reikšmes (teisinga ar klaidinga), su kuriomis galima atlikti logines operacijas;
- vardinis tipas – diskretus tipas, kai tipo apraše išvardinamos visos galimos reikšmės;
- simbolinis tipas – diskretus tipas, kurio galimos reikšmės – simboliai ar jų eilutės;

- atkarpos tipas – aprašomas apatinis ir viršutinis rėžiai tam tikroje aibėje (pavyzdžiui, sveikųjų skaičių, simbolių);
- sveikųjų skaičių tipas – galimos reikšmės iš sveikųjų skaičių aibės;
- realiųjų skaičių tipas – teoriškai galimos reikšmės iš realiųjų skaičių aibės, tačiau praktiškai jis realizuojamas reikšmėmis iš mažesnės racionaliųjų skaičių aibės.

Lentelėje išvardinti *MS Access* ir daugelio kitų sistemų palaikomi pagrindiniai duomenų tipai.

I-2 lentelė. Pagrindiniai duomenų tipai

Duomenų tipas	**Paaiškinimas**	**Užimama atmintis**	**Apimamas intervalas**
Byte	Mažas sveikas skaičius	1 baitas	0 – 255
Boolean	Loginis tipas	2 baitai	*True* arba *False* (teisinga arba klaidinga)
Integer	Sveikas skaičius	2 baitai	-32768 – 32767
Long (long integer)	Didelis sveikas skaičius	4 baitai	-2147483648 – 2147483647
Single (single-precision floating-point)	Racionalus skaičius	4 baitai	$-3.402823*10^{38}$ – $-1.401298*10^{-45}$ (neigiamiems skaičiams); $1.401298*10^{-45}$ – $3.402823*10^{38}$ (teigiamiems skaičiams);
Double (double-precision floating-point)	Didelis (dvigubo tikslumo) racionalus skaičius	8 baitai	$-1.79769313486231*10^{308}$ – $-4.94065645841247*10^{324}$ (neigiamiems skaičiams); $4.94065645841247*10^{-324}$ – $1.79769313486232*10^{308}$ (teigiamiems skaičiams);
Currency (scaled integer)	Valiuta (išvengiama apvalinimo klaidų)	8 baitai	-922337203685477.5808 – 922337203685477.5807
Decimal	Dešimtainis (išvengiama apvalinimo klaidų)	14 baitų	+/-79228162514264337593543950335 (be dešimtainio skirtuko); +/-7.92281625142643375935439503 35 su 28 pozicijomis į dešinę nuo dešimtainio skirtuko; mažiausias neneigiamas skaičius +/-0.0000000000000000000000000001
Date	Datos tipas	8 baitai	0100-01-01 – 9999-12-31
Object	Nuoroda į objektą	4 baitai	Nuoroda į bet kokį objektą
String (variable-length)	Tekstinių simbolių eilutė, kurios ilgis kintamas	10 baitų + eilutės ilgis (simbolių skaičius)	0 – maždaug 2 milijardai
String (fixed-length)	Tekstinių simbolių eilutė, kurios ilgis pastovus	eilutės ilgis	1 – maždaug 65400
Variant (with numbers)	Kintamo dydžio skaičius	16 baitų	Bet kokia skaitinė reikšmė, patenkanti į *Double* intervalą
Variant	Kintamo ilgio eilutė su	22 baitai +	Toks pat kaip kintamo ilgio tekstinių

(with characters)	skaičiais ir simboliais	eilutės ilgis	simbolių eilutės (*String, variable-length*)
User-defined (using Type)	Naudotojo apibrėžtas duomenų tipas	Tiek, kiek užima kompo-nentai	Atitinkamai kiekvieno komponento duomenų tipo

Sudėtingi (struktūriniai) duomenų tipai dar vadinami **duomenų struktūromis**. Duomenų struktūra apibrėžia struktūrines reikšmes ir jų sujungimo būdą. Dažniausiai naudojamos duomenų struktūros yra:

- masyvas – daugelio to paties tipo reikšmių fiksuoto dydžio matrica, kurioje elementai indeksuojami pagal masyvo matmenis – jų gali būti 1, 2, 3, o kartais ir daugiau;
- įrašas (struktūra) – įvairių tipų duomenų grupė;
- tiesinis sąrašas – to paties tipo reikšmių kintamo ilgio seka. Toks sąrašas gali būti dviejų pagrindinių tipų: eilė ir stekas Eilės komponentai saugomi ta tvarka, kuria buvo į eilę įdėti, o išimti iš eilės galima tik pirmą įdėtą komponentą (angl. *FIFO – First In First Out*). Steko komponentai saugomi taip pat, o išimti iš jo galima tik paskutinį įdėtą komponentą (angl. *LIFO – Last In First Out*);
- medis – yra paprastai vieno tipo elementų hierarchinė duomenų struktūra, kurioje elementus sieja „tėvų-vaikų“ santykiai. Kiekvienas „tėvas“ yra susietas su vienu ar daugiau „vaikų“. Medžio elementai yra vadinami medžio viršūnėmis. Elementas, kuris neturi tėvo, vadinamas šaknimi, o elementai, neturintys vaikų, vadinami lapais;
- grafas – paprastai vieno tipo elementų matematinio grafo duomenų struktūra, modeliuojama matrica, kurioje grafo viršūnės saugomos kaip stulpelių ir eilučių pavadinimai, o matricos elementai įgauna teigiamą ar klaidingą reikšmę priklausomai nuo to ar atitinkamos viršūnės yra susietos briauna.

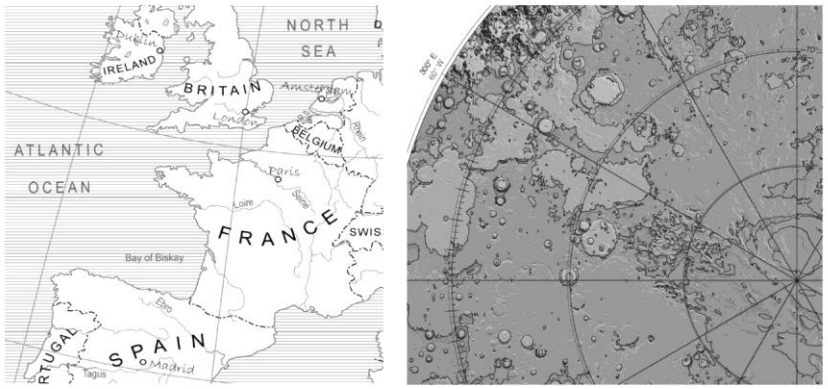
Kaip jau minėta, sudėtiniai tipai gali būti labai įvairūs. Objektiniame programavime vietoje duomenų tipo naudojama klasės sąvoka. Klasė – tai aprašas, nusakantis tam tikros rūšies objektų duomenų struktūrą ir manipuliavimo jais taisykles (metodus). Klasė skiriasi nuo duomenų struktūrų tuo, kad jos apibrėžimas apima ir duomenų vientisumo taisykles, kurios turi būti tenkinamos po kiekvieno metodo iškvietimo. Tai yra, galima ne bet kuri įmanoma klasės objekto būsena. Be to, klasėse apibrėžiamos specialios kūrimo ir naikinimo funkcijos, klasės siejamos paveldimumo ryšiais.

Savitas duomenų tipas yra tiesiog “didelis objektas” BLOB (angl.: *binary large object* arba *basic large object*). Tai dvejetainiu kodu saugomas duomenų rinkinys, DBVS traktuojamas kaip vientisas objektas. Dažniausiai tai paveikslas, garso, vaizdo įrašas ar interaktyvus objektas. BLOB nėra universalus duomenų tipas, jo palaikymas priklauso nuo duomenų bazių valdymo sistemos. Dabartinis akronimas kilo iš angliško žodžio *blob* (gniutulas, amorfinė masė) ir buvo siejamas su duomenų perkėlimu iš vienos duomenų bazės į kitą netikrinant jų struktūros. Toks duomenų tipas tapo praktiškai panaudojamas duomenų bazėse tik labai atpigus disko vietai.

I.4 GEOGRAFINIAI DUOMENYS IR GIS

Erdviniai duomenys – tai duomenys, kurių būtinas komponentas yra informacija apie padėtį erdvėje. Toks komponentas praktiškai visada yra rinkinys koordinačių tam tikroje atskaitos

sistemoje. **Geografiniai** (dar kartais sakoma: geoerdviniai, angl. *geospatial*) **duomenys** – tai erdviniai duomenys, kurių koordinatės nustatomos Žemės paviršiaus atžvilgiu. Dažniausiai kalbant apie duomenų bazes šie terminai naudojami kaip sinonimai. Lietuvoje 2009–2011 m. paplito netikslus termino „erdvinis“ vartojimas turint omenyje geografinius duomenis ar sistemas, kilęs iš teisės aktų vertimų. Erdviniai duomenys plačiąja prasme gali neturėti nieko bendro su geografija – tai gali būti molekulės, žmogaus kūno ar kitokie dviejų, trijų ar daugiau matavimų duomenys.



I-4 pav. Erdviniai geografiniai (a) ir erdviniai negeografiniai[1] (b) duomenys

Euklido geometrija (4 a. pr. m. e.) yra pagrįsta aksiomų sistema, kuri apibūdina taško, tiesės ir plokštumos sąvokas bei ryšius tarp jų. Aksiomų sistemoje, nusakančioje priklausomumą, tvarką, kongruentumą, tolydumą ir lygiagretumą svarbiausias yra penktasis Euklido postulatas: per tašką ne tiesėje galima išvesti vieną ir tik vieną tiesę lygiagrečią duotajai. Buvo bandoma šią aksiomą įrodyti, bet taip ir nepavykus, suabejota jos teisingumu.

19 a. pradžioje rusų matematikas N. Lobačevskis suformulavo šios aksiomos alternatyvą: per tašką ne tiesėje galima išvesti bent dvi tieses lygiagrečias duotajai. Tiesių, kertančių duotąją, kaip ir nekertančių, yra be galo daug. Tai taip vadinama hiperbolinė geometrija.

19 a. viduryje vokiečių matematikas B. Rymanas pateikė dar vieną variantą: per tašką ne tiesėje negalima išvesti nė vienos tiesės, lygiagrečios duotajai, t.y., visos tiesės kertasi. Tokia geometrija atitinka sferos paviršiaus geometriją – ji ir vadinama sferine.

Lobačevskio ir Rymano erdvėse galioja kiti dėsniai, negu Euklido erdvėje, pavyzdžiui, trikampio kampų suma yra mažesnė arba didesnė už 180 laipsnių.

[1] Marso žemėlapio fragmentas. © U.S. Geological Survey (2001)

Laikoma, kad apie 80% visų duomenų, naudojamų viešajame sektoriuje, sudaro duomenys, kurių dalis yra vienu ar kitu būdu saugoma informacija apie objektų padėtį erdvėje, arba kurie gali būti su tokia informacija susiejami. Tikri geografiniai duomenys yra taškų koordinatės, teritorijų ribos, upių, kelių linijos, ežerų kontūrai ir pan. Geografiškai susiejamų duomenų pavyzdys yra adresai – nors adreso informacija pati savaime yra tik tam tikros struktūros teksto eilutė, ji visada gali būti susieta su vieninteliu tašku ar plotiniu objektu Žemės paviršiuje. Be abejo, dauguma geografinių duomenų apima kur kas daugiau, negu vien informaciją apie objektų padėtį. Padėties informacija dažniausiai siejama su papildomais duomenimis apie negeografines objekto savybes, pavyzdžiui, upės pavadinimas, kelio kategorija ir dangos tipas, miesto gyventojų skaičius ir pan.

Nesunku pastebėti, kad duomenų „geografiškumas“ atspindi ne esmines duomenų ypatybes, o jų prasmę, t.y., tam tikrą interpretaciją įprasto pavidalo faktų, kokie yra, pavyzdžiui sveikieji ar realūs skaičiai. Tik žinant, kad tie skaičiai yra geografinės koordinatės, duomenys įsivaizduojami kaip geografiniai. Tačiau duomenis apdorojant automatiškai, jų interpretacija nėra svarbi. Gali kilti klausimas, kodėl tada duomenų bazių valdymo sistemos specializuojamos būtent geografiniams duomenims apdoroti.

Geografinių duomenų aibės laikui bėgant darosi vis sudėtingesnės. Be to, duomenų iš skirtingų gyvenimo sričių integravimas tampa būtinybe – skirtingas duomenų aibes tenka susieti teritoriškai, pagal mastelį, laiką, temą ir kitus parametrus. Atsiranda poreikis ir techninės galimybės formuluoti naujus, anksčiau neišsprendžiamus, klausimus, į kuriuos svarbu teisingai atsakyti, kad būtų galima priimti svarbius teritorijų planavimo ir valdymo sprendimus. Didžioji dalis sukauptų geografinių duomenų saugoma valstybės kadastruose: žemės ir kito nekilnojamojo turto, žemės gelmių, miškų, saugomų teritorijų, upių, ežerų, kelių, kultūros vertybių ir kt.

Geografinių duomenų bazė (GDB) – tai fizinė saugykla duomenų, kurie atitinka konkrečiu momentu ir tam tikram tikslui sukauptas žinias apie geografinę tikrovę, tai yra, tam tikras būdas saugoti geografinius duomenis DBVS. GDB pagrindą sudaro duomenys apie realių ar sutartinių objektų padėtį Žemės paviršiaus atžvilgiu. Kartais geografinių duomenų bazių samprata išplečiama iki **erdvinių** duomenų bazių. Erdvinių duomenų bazėje saugomi duomenys apie padėtį **erdvėje**, t.y., bet kokioje trimatėje ar net didesnio matavimų skaičiaus erdvėje, skirtingai nuo geografinės erdvės, nebūtinai siejamoje su Žemės paviršiumi. Erdvinių, tačiau negeografinių duomenų bazių pavyzdžiai – Mėnulio paviršiaus, dangaus skliauto, menamos Interneto erdvės duomenų bazės. Tačiau visos geografinės duomenų bazės kartu yra ir erdvinės. Šioje knygoje apsiribosime geografinėmis duomenų bazėmis, kurių konkrečių pavyzdžių įvairovė leidžia susidaryti pakankamai išsamų vaizdą apie bet kokių galimų erdvinių duomenų bazių turinį ir struktūrą.

Nors geografinė padėtis techniškai yra aprašoma ne kokio nors ypatingo tipo, o tais pačiais skaitiniais, grafiniais ar tekstiniais duomenimis, tokiais aprašymais nepatogu operuoti geografinių duomenų naudotojui. Būtent dėl erdvėje lokalizuotų duomenų naudojimo specifikos geografinių duomenų bazių valdymo sistemoms (GDBVS) keliami papildomi reikalavimai, atsispindintys visuose jau anksčiau minėtuose duomenų valdymo etapuose.

Duomenų surinkimas, paruošimas ir įvedimas. Technologijos geografiniams duomenims gauti jau dabar yra labai įvairios ir jų teikiamos galimybės darosi vis didesnės. Geografinių duomenų pagrindiniai šaltiniai gali būti visų žinomų tipų:

- skaitiniai koordinačių duomenys (GPS ar antžeminių matavimų duomenys),
- grafiniai šaltiniai (kosminiai ir aerofotovaizdai, žemėlapiai, planai);

- tekstiniai šaltiniai (vietovardžiai, aprašymai).

Tai reiškia, kad GDBVS turėtų būti priemonės efektyviai naudoti visų tipų duomenis, taigi, ir galimybė atlikti pirminį duomenų apdorojimą, pavyzdžiui, statistinius skaičiavimus ar vaizdo šviesumo bei kontrasto koregavimą. Norint sistemoje išsaugoti geografinio objekto, pavyzdžiui, upės ar kelio duomenis, būtina įvesti bent būdingus jų taškus su koordinatėmis. Kiekvienam taškui nurodomos dvi arba trys koordinatės reikiamu tikslumu. Tai galima atlikti automatiškai, pavyzdžiui, nuskaitant GPS imtuvo duomenis. Tačiau dažnai neįmanoma visiškai automatizuoti koordinačių įvedimo proceso. Pusiau automatinis arba žmogaus atliekamas geografinių koordinačių įvedimas vadinamas *vektorizavimu* arba geografinių duomenų skaitmeninimu. Akivaizdu, kad suvedant daug taškų, koordinačių skaičiavimas ir įrašymas į lentelę kiekvienam taškui nėra efektyvus procesas, be to, nematant vektorizuojamo objekto, lengva suklysti. Įprastas erdvinių objektų vektorizavimo būdas yra jų kontūrų „perpiešimas“ naudojant tam skirtus įrankius, kurių analogai yra daugumoje grafinio redagavimo programų. Geografinių duomenų bazių valdymo sistemos teikia panašią galimybę – įvesti taško koordinates vienu klavišo (pelės ar kito skaitmeninimo įrenginio) paspaudimu. Naudotojas pasirenka norimus taškus ant ekrane matomo ar popierinio žemėlapio, o sistema turi mokėti apskaičiuoti jų koordinates pagal nuskaitymo įrenginio poslinkį ir nurodytus koordinačių sistemos atskaitos taškus. Dar daugiau, vektorizavimas gali būti automatizuotas, t.y., naudojama taikomoji programa, turinti galimybes atpažinti vektorinius objektus skaitmeniniame rastriniame vaizde ir apskaičiuoti jų kontūrų taškų koordinates pagal naudotojo įvestus koordinuotų to paties vaizdo taškų duomenis. Tai labai palengvina žmogaus darbą, tačiau reikalauja papildomų specifinių funkcijų iš taikomosios programos, kurią jis naudoja.

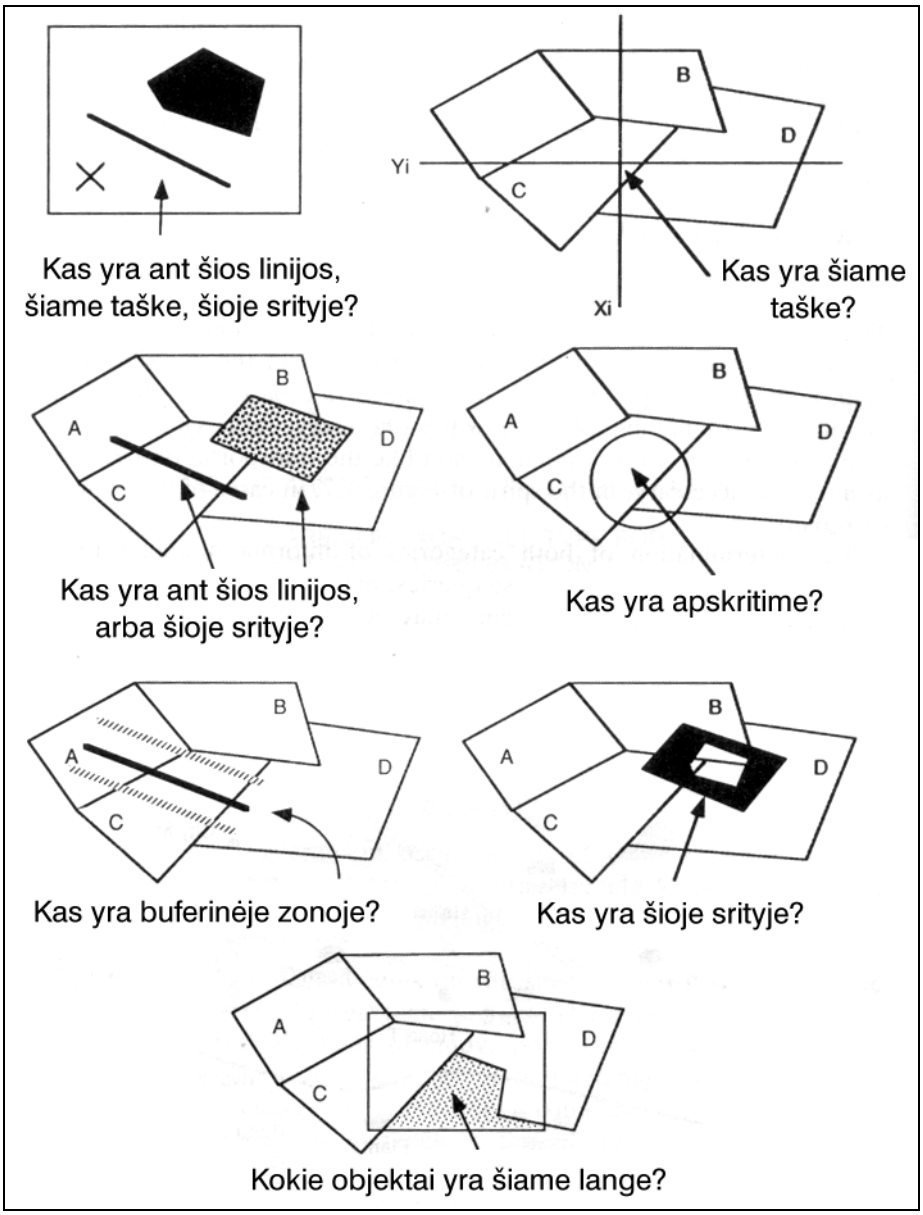
Duomenų saugojimas ir perdavimas. Geografiniai duomenys paprastai pasižymi didele apimtimi bei sudėtingais tarpusavio ryšiais. Pavyzdžiui, nedidelio regioninio parko išteklių valdymo projekto duomenų bazės apimtis siekia 10-20 megabaitų, vidutinio dydžio projekto rastrinių ir vektorinių GIS duomenų bazė gali užimti kelis šimtus megabaitų, nacionalinė GIS duomenų bazė – nuo keliolikos iki kelių šimtų gigabaitų, o ortofotografinių vaizdų nacionalinis archyvas – terabaitus. Įsivaizduojant, kad įvairių tipų duomenys naudojami sudarant žemėlapį, jie turėtų būti saugomi taip, kad prireikus visas žemėlapis būtų greitai atkurtas, tačiau taip pat turi būti galima išskirti vienos rūšies geografinius objektus, pavyzdžiui, upes, kelius, miestus, arba įvairias objektų grupes (dažnai atitinkančias žemėlapio sluoksnius, kuriuose gali būti skirtingų geometrinių savybių objektai), pavyzdžiui, hidrografinius objektus, vienos savivaldybės teritorijoje esančius inžinerinius statinius ir pan. Todėl GDBVS privalo turėti efektyvias priemones operuoti dideliais duomenų masyvais, o keitimuis duomenimis būtinas didelės spartos Interneto ryšys. Geografiniams duomenims greičiau perduoti dažnai naudojami podėlių (angl. *cache*) komponentai, dubliuojantys originalius duomenis, sukurti ir paruošti naudoti iš anksto. Prireikus duomenų, jei jie yra podėlyje, juos galima gauti greičiau, jei nėra – tenka atlikti skaičiavimus ar pasiekti duomenis iš originalios saugyklos. Pavyzdžiui, kuriant žemėlapių podėlį sistemoje išsaugomi skirtingo mastelio to paties žemėlapio vaizdai, kurie ir perduodami naudotojui taip padidinant efektyvumą dviem aspektais:

a) nereikia kiekvieną kartą pastūmus žemėlapį ar pakeitus jo mastelį iš naujo vaizduoti duomenų;
b) vaizdo perdavimo greitis nepriklauso nuo to, kiek objektų yra žemėlapyje.

Podėlyje duomenys dalinami lapais (angl. *tile*), bet tai nėra vienintelis duomenų *paskirstymo* būdas. Didelės apimties duomenys neretai dalinami į lapų ir (arba) teminių sluoksnių blokus, kartais net saugomus skirtingose vietose. Tokius duomenis paprasčiau dalimis atnaujinti ir platinti. Paskirstytoms geografinių duomenų bazėms svarbus duomenų bazės vientisumo, turint omenyje duomenų sutapimą lapų kraštuose reikalavimą (angl. *seamless database*).

Duomenų rūšiavimas, paieška ir operacijos su duomenimis. Geografiniai duomenys ypatingi tuo, kad naudotojus dažnai domina objektų tarpusavio santykiai erdvėje, tokie, kaip įdėtumas, sankirta, atstumai ir pan. (I-5 pav.) Tokios erdvinės operacijos žemiausiame lygmenyje atliekamos su taškų koordinatėmis, t.y., su elementariais dvejetainiais duomenimis, kaip ir visos kitos operacijos. Pagrindinė problema yra ta, kad norint rasti duomenis pagal erdvinius kriterijus, pavyzdžiui, teritorijos plotą ar perimetro ilgį, tokių skaičiavimų reikia atlikti labai daug ir tai trunka ilgai net naudojant šiuolaikinius kompiuterius. Todėl nuolat ieškoma naujų efektyvesnių geografinės informacijos apdorojimo algoritmų, naudojami lygiagretūs skaičiavimai ir pan.

Rezultatų pateikimas. Pagrindinis GDBVS produktas yra žemėlapis, iš kurio tikimasi didelio tikslumo ir geros kokybės, įskaitant ir gerą skaitomumą bei estetišką išvaizdą. Akivaizdu, kad nėra taip paprasta gražiai pateikti geografinių objektų vaizdus ant popieriaus ar kompiuterio ekrane. Tam reikalingos labai įvairios kompiuterinės grafikos galimybės, kurių vis daugiau turi šiuolaikinės GIS sistemos. Kita problema – išsaugoti tų vaizdų (paprastai daug kartų sumažintų, projektuotų plokštumoje) ryšį su realiais duomenimis – dar tik sprendžiama. Populiaria geografinių duomenų teikimo naudotojams aplinka tapo Internetas, kurio dėka daugelis žmonių gali dirbti su tais pačiais geografiniais duomenimis. Galimybę naudoti duomenis tokius, kokie jie yra (ne žemėlapio paveikslėlis, o atskiri objektai) teikia geografinių elektroninių paslaugų (angl. *Web services*) technologija.



I-5 pav. Geografinių uždavinių (užklausų) pavyzdžiai

Geografinių duomenų apdorojimo technologijos, o siauriau jas suprantant – programų sistemos, iš esmės yra ne kas kita, kaip duomenų bazių valdymo sistemos, pritaikytos atlikti anksčiau išvardintus pagrindinius duomenų valdymo etapus, kai dauguma duomenų sistemoje yra geografiniai. Dėl šios priežasties geografinių duomenų bazių valdymo sistemos paprastai nenagrinėjamos atskirai, o laikomos neatsiejama **geografinės informacijos sistemų** (GIS) dalimi. Žinoma, kaip ir kiekviena informacijos sistema, GIS yra ne tik GDBVS, bet ir kiti informacinei sistemai funkcionuoti būtini komponentai, iš kurių ypač svarbūs yra **geografinės analizės** įrankiai. GIS ir tradicinės DBVS dažnai yra pagrįstos skirtingais duomenų sisteminimo principais.

GIS programinė įranga yra labai įvairi. Galima suskaičiuoti keliasdešimt daugiau ar mažiau populiarių GIS pakraipos paketų, dažnai specializuotų tam tikro tipo erdvinėms operacijoms. Iš universalesnių GIS galima paminėti *GRASS*, *MapServer, Quantum GIS, SAGA GIS* (atviro kodo),

ESRI ArcGIS, *Pitney Bowes MapInfo*, *Intergraph GeoMedia* ir *GeoMedia WebMap, Autodesk MapGuide, ERDAS IMAGINE*, *IDRISI* (komercinės).

Reikia pastebėti, kad GIS sistemos savo efektyvumu ir saugumu neprilygsta ilgą vystymosi istoriją turinčioms reliacinėms ar objektinėms DBVS. Didelės GIS sistemos dažniausiai naudoja tradicines reliacines DBVS didžiajai daliai informacijos saugoti ir tvarkyti. GIS sistemos apskritai vis dažniau asocijuojasi ne su duomenų valdymu, bet su specifinėmis sudėtingomis duomenų analizės ir vaizdavimo funkcijomis. Iš kitos pusės, 21-ojo amžiaus pradžioje jau nemažai komercinių ir atviro kodo DBVS paketų turi specialius plėtinius efektyviam darbui su reliacinėje duomenų bazėje saugomais erdviniais duomenimis. Tai *Oracle Spatial, IBM DB2 Spatial Extender, PostGIS, Informix Spatial DataBlade* ir kiti.

Šioje knygoje neaptarinėsime DBVS aspektų, specifinių tik geografiniams duomenims, nes ši tema pati savaime yra pakankamai plati ir sudėtinga, be to, yra nemažai vien tik jai skirtos literatūros.

I.5 GEOGRAFINIŲ DUOMENŲ MODELIAI

Informacija apie realų pasaulį yra tolydi, t.y., kiekviename taške kažkas yra. Net ir pažangiausios šiuolaikinės technologijos neleidžia saugoti tolydžios informacijos, nes tai reikštų begalinę duomenų apimtį. Todėl realaus pasaulio informacija, taip pat ir vaizdinė, yra supaprastinama iki baigtinės aibės duomenų elementų. Priklausomai nuo to, kaip toks supaprastinimas atliekamas, sakome, kad informacija koduojama tam tikru būdu, naudojant pasirinktą modelį.

Geografinėse duomenų bazėse galimi skirtingi loginiai įvairių tipų duomenų organizacijos būdai:

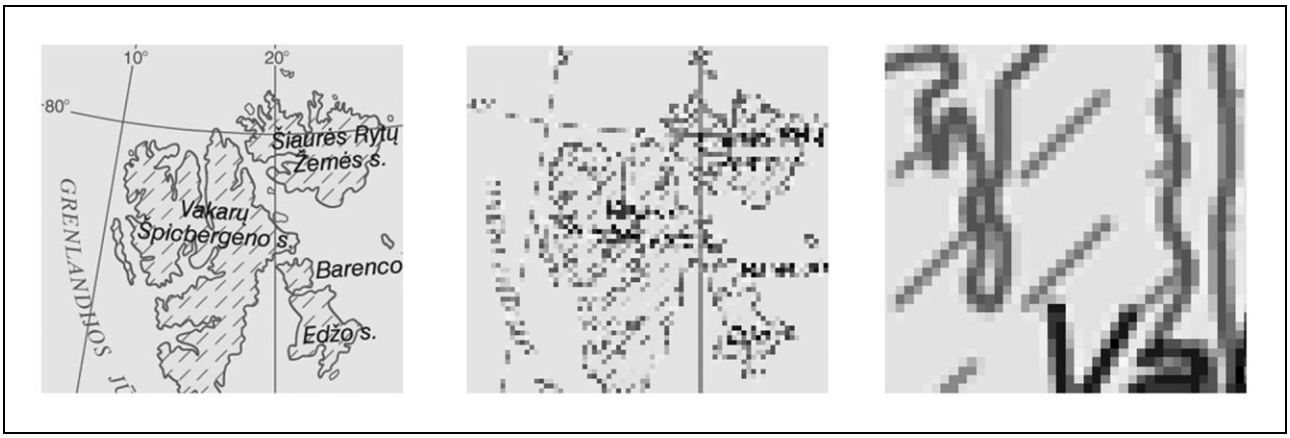
1. Pagal struktūrą. **Sluoksnių** modelis yra viena labiausiai paplitusių struktūrų. Dažniausiai vienam sluoksniui priskiriami vieno tipo objektai. Tai teminis požiūris, orientuotas į naudojimą, jis įprastas kai kalbama apie geosferas. Bet objektų skirstymas į sluoksnius negali būti vienareikšmis, kaip ir "vieno tipo" apibrėžimas. Sluoksniai nebūtinai išskiriami pagal temą, jie gali atitikti skirtingus laikotarpius, pastato aukštus ar reljefo aukščius, 4 spaudai naudojamas spalvas ir pan. Struktūrizavimas **pagal temas ir erdvės sritis**, pavyzdžiui, topografinio žemėlapio lapais, buvo naudojamas daugiausia analoginėse informacinėse sistemose, bet dar vis taikomas ir skaitmeniniams produktams.
2. Pagal informacijos prigimtį. **Objektinis modelis** sudaromas geografinei vietai, kurioje yra skirtingų objektų, be to, visi objektai laikomi svarbiais kuriame nors kontekste. Objektai saugomi viename sluoksnyje, jei reikia, vertikalias jų padėties variacijas rodant trečiuoju – aukščio – matavimu. Šis modelis yra labiausiai „žmogiškas“, tačiau nelabai tinka modeliuoti įvairiems tolydiems paviršiams, tokiems kaip reljefas, statistiniai paviršiai ir pan. Erdvę nebūtinai sudaro atskiri objektai. Joje gali būti paplitęs tolydus reiškinys, pavyzdžiui, atmosfera arba miškas. Tokį reiškinį aprašyti objektų rinkiniu nepatogu. Egzistuoja ir kita – **tolydaus lauko** koncepcija, kai tolydumas išreikštai laikomas svarbiausia savybe. Tai reiškia, kad kiekvienas taškas (padėtis) dvimatėje ar trimatėje koordinačių erdvėje turi kokio nors atributo reikšmę. Bendruoju atveju toks taškas aprašomas kaip aibė reikšmių: {x, y, z, t, a}; čia t – laiko momentas, a – neerdvinis atributas ar jų aibė, kurių reikšmės turėtų būti iš principo empiriškai patikrinamos, t.y., kiekvienoje vietoje galima stebėti paplitusio reiškinio savybes. Tuo šis požiūris iš esmės skiriasi nuo objektų modelio, kuriame a nėra visur apibrėžtas, o reiškinys vaizduojamas kaip {e, s, t, a}; čia e yra erdvėje dislokuotas objektas, o jo padėtį nusako s, kuris gali būti kiek reikia sudėtingas.

Žemesniu lygmeniu geografiniai ir apskritai bet kokie grafiniai duomenys gali būti koduojami ir saugomi pagal tris pagrindinius modelius – rastrinį, vektorinį ir mišrų (gardelės ar netaisyklingų trikampių tinklo). Su geometriniais objektais šiuose modeliuose siejama atributinė geografinių objektų informacija.

I.5.1 Rastro duomenų modelis

Rastrinis, arba taškinis, vaizdas – tai vaizdas, sudarytas iš vienas po kito eilutėmis ir stulpeliais išdėstytų vienodo dydžio vaizdo elementų, pakankamai mažų, kad stebėtojui jie susilietų į tolydų vaizdą. Rastrinį vaizdą galima įsivaizduoti kaip stačiakampį tinklelį, kuriame elemento padėtis nurodoma poslinkiu nuo sutartinio pradžios taško. Teoriškai rastrą gali sudaryti įvairių formų elementai, pavyzdžiui trikampiai ar šešiakampiai, be tarpų užpildantys plokštumą. Praktiškai dažniausiai naudojami stačiakampiai elementai.

Vienas įprastas vaizdo elementas nuo kito skiriasi tik padėtimi ir spalva. Ekrane toks vaizdo elementas yra šviečiantis ekrano taškas, vadinamas pikseliu (angl. *PixEl – Picture Element*). Rašaliniu spausdintuvu spausdintame vaizde vaizdo elementas yra apvalus taškas, gaunamas spausdintuvo rašalo adatai susiliečiant su popieriumi. Tradicinėje fotografijoje toks elementas yra emulsijos, dengiančios fotografinę juostelę ar popierių, grūdelis. Geografiniame tinklelyje elementas yra gardelė. Visa rastrinio vaizdo informacija koduojama tuo pačiu principu „taškas–spalva“, o vaizdo kokybė priklauso nuo to, kokio didumo ir kaip arti vienas nuo kito yra vaizdo elementai. Ši charakteristika vadinama rastrinio vaizdo *skiriamąja geba* arba *rezoliucija* (angl. *resolution*) ir kompiuterinėse sistemose tradiciškai matuojama vaizdo elementų („taškų“) skaičiumi viename eilutės colyje (angl. *dpi – Dots Per Inch*).



I-6 pav. Rastrinio žemėlapio fragmentai: 300 dpi; 72 dpi; padidintas tiek, kad matomi jį sudarantys vaizdo elementai.

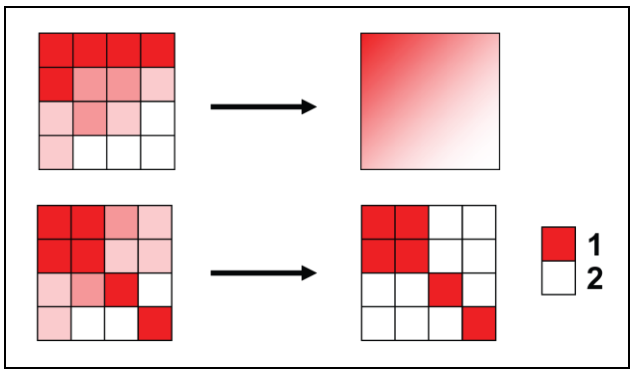
Kompiuterinėje grafikoje yra skirtingi būdai vaizdui koduoti, paverčiant jį vaizdo elementų seka. Nuo kodavimo metodo priklauso skaitmeninio vaizdo *formatas*. Pagrindinis rastrinio vaizdo formatas yra **bitų žemėlapis** (angl. *BMP – Bit MaP*). Tokio formato failai atpažįstami pagal plėtinį „.bmp“, pavyzdžiui, *zemelapis.bmp*.

Bitų žemėlapyje kiekvienam vaizdo elementui skiriamas tam tikras elektroninės atminties vienetų – bitų skaičius, kuris saugo informaciją apie elemento spalvą dvejetainiu kodu. Dažniausiai vienam elementui skiriama 16 arba 32 bitai, t.y., du arba keturi baitai. Kuo daugiau skiriama atminties vienam elementui, tuo tikslesnę informaciją apie elemento spalvą (atspalvį) galima saugoti.

Bitų žemėlapis yra išsamus bet neefektyvus duomenų saugojimo metodas. Vaizduose dažnai pasitaiko vienodos spalvos elementų grupės – linijos ir plotai. Neracionalu kiekvienam iš šimtų vaizdo elementų, sudarančių vienos spalvos jūros plotą I-6 paveiksle skirti keturis baitus saugoti tai pačiai informacijai. Todėl metodas patobulinamas – atliekama tai, kas vadinama vaizdo suspaudimu (glaudinimu). Paprasčiausia yra vienodų vaizdo elementų sekos pradžioje nurodyti, kad tam tikras skaičius iš eilės einančių elementų bus vienodi ir saugoti vienintelę spalvos reikšmę jiems visiems. Taip sutaupoma daug atminties ir neprarandama nė kiek pradinės informacijos. Yra įvairių rastrinių vaizdų formatų, kurie leidžia saugoti vaizdus efektyviau, bet neprarandant originalios informacijos.

Vis dėlto skaitmeniniams vaizdams dažniausiai reikia daug vietos, todėl nuolat ieškoma būdų, kaip dar labiau sumažinti jų apimtį. Tą galima padaryti tik prarandant tam tikrą dalį pradinės informacijos. Yra sukurta įvairių glaudinimo algoritmų, kurie leidžia pasiekti kompromisą tarp failo dydžio ir prarandamos informacijos kiekio. Yra du iš esmės skirtingi glaudinimo metodai.

- **Fotografinių vaizdų** glaudinimas yra pagrįstas ta jų savybe, kad, nors gali būti daug spalvų ir atspalvių, ribos tarp jų yra neryškios, persiliejančios, todėl vaizdo fragmento pikselių grupę dažnai galima pakeisti stačiakampiu, nuspalvinant jį pereinančiomis viena į kitą spalvomis pagal tam tikrą schemą, parinktą taip, kad sukuriamas įspūdis būtų kuo panašesnis į pradinį vaizdą. JPEG (angl. *Joint Photography Experts Group*) glaudinimo algoritmas yra labai efektyvus ir išsaugo pagrindines įprasto fotografinio vaizdo savybes, nors tiksli informacija prarandama.



I-7 pav. Glaudinimo skirtingais metodais rezultatai.

- **Brėžiniai ir piešiniai** iš esmės skiriasi nuo fotovaizdų – jiems būdingos plonos ryškios linijos (kurios nutrūktų taikant JPEG glaudinimo algoritmą) ir griežtai apibrėžtos vienos spalvos dėmės. Tokių vaizdų glaudinimo principas yra visiškai kitoks. Pavyzdžiui, GIF (angl. *Graphic Image Format*) ar PNG (angl. *Portable Network Graphics*) formatu išsaugotame vaizde yra nedidelis skaičius spalvų, kurios sunumeruotos, pavyzdžiui, jei spalvų yra 128, kiekvieno pikselio spalvai saugoti užtenka skirti ne 4 baitus, o vos 6 bitus. Taip glaudinant vaizdą artimos spalvos pakeičiamos viena iš riboto dažniausiai pasitaikančių spalvų rinkinio, o vaizdo elementų skaičius nesumažėja.

Ortofotografiniams vaizdams glaudinti naudojamas specialiai GIS tikslams sukurtas *MrSid* (angl. *Multi-resolution Seamless ImageDatabase*) formatas, kuris leidžia saugoti ir skirtingos skiriamosios gebos vaizdo duomenis[1].

Ortofotografinis žemėlapis yra įprastas rastrinių geografinių duomenų pavyzdys. Tačiau dar dažniau geografiniams duomenims saugoti naudojamas geografinio rastro (gardelių) modelis, kuriame vaizdo elementas siejamas ne su spalva, o su kokio nors atributo reikšme paviršiaus taške. Jis plačiau aprašytas I.5.3 skyrelyje.

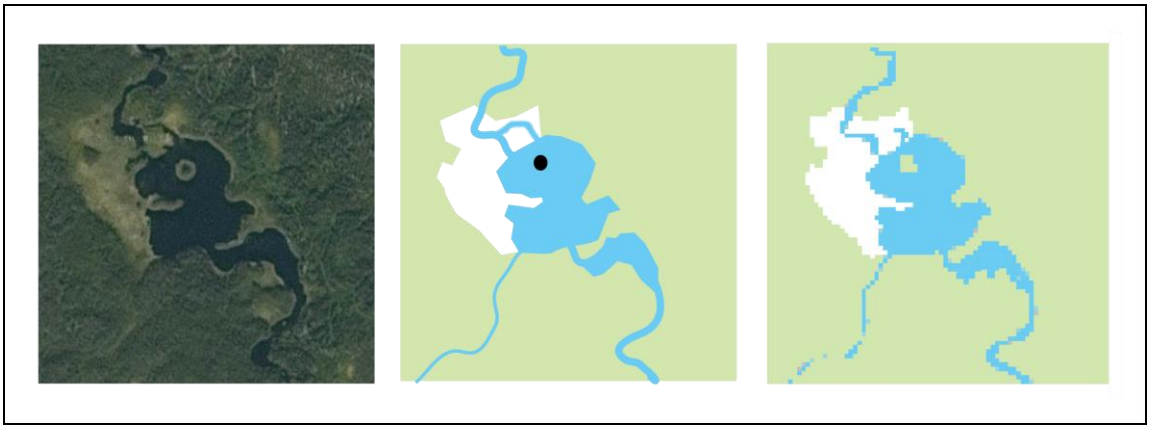
Pagrindiniai rastrinio modelio trūkumai yra šie:

- neįmanoma išskirti geografinių objektų ir su jais atlikti transformavino ar analizės operacijų;
- labai ribotos galimybės kaupti atributinę informaciją;
- jei duomenys detalūs, reikia daug vietos jiems saugoti, bet kuriuo atveju tikslumas visada vienodas ir apribotas vaizdo elemento dydžio;
- jei duomenys skirtingo detalumo (pavyzdžiui, gyvenvietė su įvairiais objektais ir vientisi miškų plotai), saugojimas tampa neefektyvus.

Šis modelis gerai tinka saugoti duomenims, kurie tolydžiai dengia paviršių, pasižymi dideliu heterogeniškumu, yra dažnai analizuojami geostatistiniais metodais.

I.5.2 Vektorinis geografinių duomenų modelis

Vektorinė informacija – tai diskrečių objektų skaitmeninių vaizdų rinkinys, gaunamas įvedant objektų koordinates erdvėje. Specialiai darbui su geografiniais duomenimis skirtose sistemose informacija apie objektų savybes saugoma atskirta nuo informacijos apie jų padėtį erdvėje ir laike, o padėtis ir erdvinės savybės modeliuojamos geometriniais objektais Euklido erdvėje. Toks modelis vadinamas **vektoriniu duomenų modeliu.** Tai sudėtingiausias geografinių duomenų modelis.



I-8 pav. Žemės paviršiaus objektų (kairėje) vaizdavimas vektoriniu (viduryje) ir rastriniu (dešinėje) modeliu.

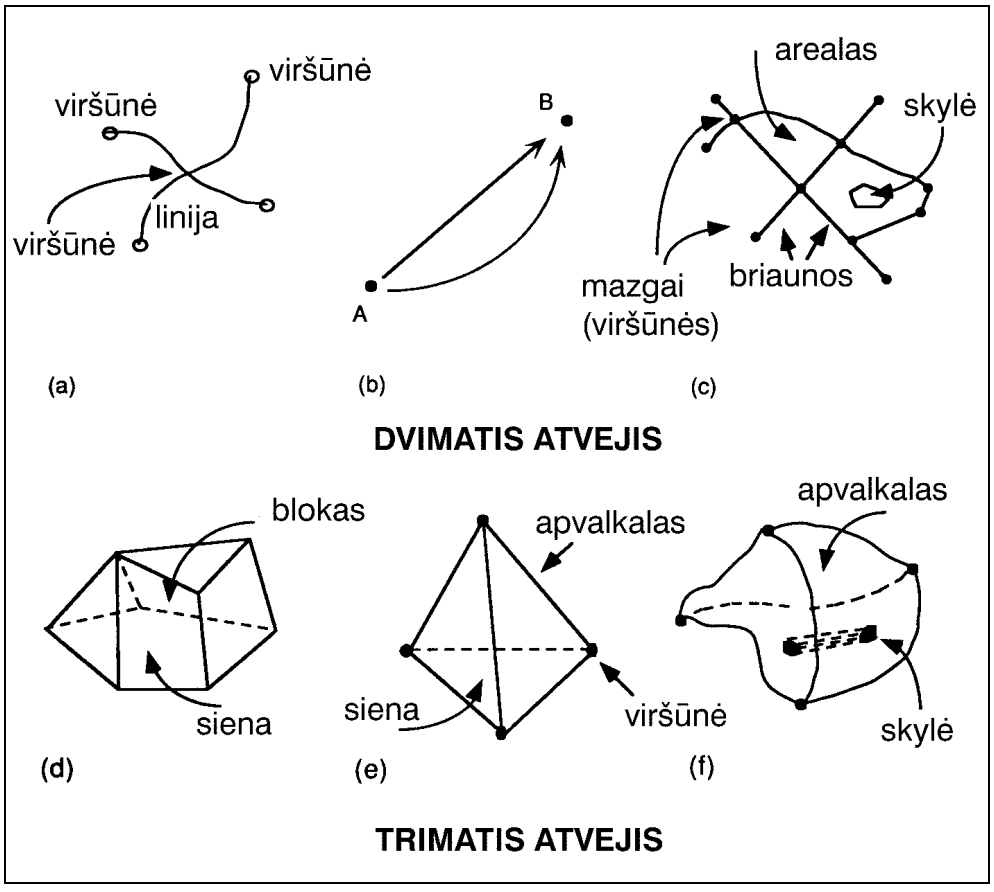
Geografiniai objektai gali būti nagrinėjami pagal erdvės matavimų skaičių, erdvinių savybių tipą arba derinimo būdus. Įprasta juos klasifikuoti pagal matavimų skaičių (ne erdvės, nes praktiškai visi geografiniai objektai yra trimačiai, bet pagal objekto matavimo galimybę). Taško, linijos, arealo ir bloko sąvokas atitinka 0, 1, 2 ir 3 matavimai: nematuojamas, tik ilgis; ilgis ir plotis; ilgis, plotis ir gylis/aukštis. Praktiškai taškas turi dydį žemėlapyje, tiesiog į jį nekreipiama dėmesio.

[1] http://en.wikipedia.org/wiki/MrSID

Stebėjimo nuotolis ar kartografinis mastelis gali lemti suvokiamų matavimų skaičių, be to, kai kurie objektai gali išnykti.

Taškai apibrėžti duotam stebėjimo masteliui aprašomi bent dviem padėties charakteristikomis, identifikatoriumi ir papildomais atributais. Žemėlapyje taškas gali būti paprasta lokacija, arba reikšti kitus objektus – užrašą ar ikoniniu sutartiniu ženklu žymimą objektą. Tašką sukuria linijų sankirta arba galai. Jis neturi dydžio.

Būdingi taškų pavyzdžiai:
- taškas-objektas (vaizduoja realų objektą);
- taškas-žymė (tekstinio objekto padėtį žymintis taškas);
- centroidas (arealo atributų nešėjas);
- viršūnė ar mazgas (topologinė jungtis ar galinis taškas tinkle). Tai taškas, kuriame linija baigiasi ar kertasi su kita linija. Šis objektas pasižymi jungumo su linijomis savybe.



I-9 pav. Vektorinio duomenų modelio sąvokos

Linijos (angl. *polyline, arc, edge*) – tai keliai, upės, ryšių linijos. Jos gali egzistuoti kaip atskiri objektai arba būti sujungtos į tinklą. Būdingi linijos atributai – ilgis, orientacija, glodumas. Linija yra vizualiai suvokiama kryptinga jungtis tarp dviejų taškų. Linijos kryptimi ji matuojama, taip atskiriant padėtį erdvėje nuo objekto matavimų skaičiaus. Uždara linija nurodo arealą.

Būdingi linijų pavyzdžiai:
- segmentas (tiesi linija, kurios abu galai baigiasi taškais – viršūnėmis);
- laužtė (netiesi linija);
- grandinė (linija, sudaryta iš kelių segmentų);
- orientuota briauna ar grandinė;
- pilna grandinė (gaunama, jei yra nurodyta kairioji ir dešinioji pusės);
- žiedas (uždara linija).

Arealai (angl. *polygon, region, zone*) – žemės paviršiaus objektai, kurių aukštis nenurodomas. Arealai atitinka natūralius (dirvožemiai, ežerai, salos, pastato stogas) ar dirbtinius (statistinius) objektus, pavyzdžiui, rajonai. Tai taip pat gali būti tolydžios erdvės diskretizacijos, pavyzdžiui, klimato zonos. Jų ribos yra linijiniai objektai, kurie ne visada svarbūs arba gali nebūti tiesiogiai stebimi, kintantys. Būdingi arealo atributai – plotas, perimetro ilgis, izoliuotumas ar sąsajos, forma, pavyzdžiui, skylės, enklavo, eksklavo buvimas, kontūro tipas, persidengimas su kitais objektais ir pan.

Būdingi arealų pavyzdžiai:
- vidinė sritis (be ribos). Tai plotinė figūra, apribota mažiausiai trijų briaunų;
- poligonas (su riba – išoriniu žiedu);
- kompleksinis poligonas (su vidiniais žiedais).

Linijos ir arealai gali būti tolydūs arba diskretūs. Taškai visada yra diskretūs.

Blokai (angl. *solid, block, polyhedron*) – tai trimačiai dariniai, tūrinės figūros, turinčios vidų ir išorę, apribotos paviršiaus plokštumų, kurios turi bendras viršūnes ir briaunas. Blokai gali būti taisyklingi ir netaisyklingi. Jie, kaip ir arealai, gali turėti aiškias arba neapibrėžtas ribas, o be to, ribojantį paviršių (angl. *shell*). Būdingi bloko atributai – tūris, paviršiaus plotas, pjūvis ir pan.

Vieno tipo objektai gali būti apjungti į sudėtinį (angl. *compound*) objektą. Vieno tipo erdviniai objektai gali būti transformuojami į kitą tipą, arba turėti alternaryvius didesnio arba mažesnio matavimų skaičiaus objektus (alternatyvos pavyzdys: arealas–kontūras). Jei derinius sudaro skirtingų tipų objektai, jie vadinami sudėtingais (angl. *complex*). Sprendžiant daugelį uždavinių naudojamos diados – dviejų objektų deriniai.

Padėties nustatymo uždaviniai ar erdvinės užklausos, o ypač greitojo reagavimo uždaviniai, reikalauja įvairiausių derinių, pavyzdžiui, reikia atrinkti rajonus, kuriuose yra daug gabių studentų (arealai–taškai). Net paprasčiausių užklausų atveju iš taško, linijos ir arealo galima sudaryti devynias skirtingas poras (II-1 lentelė). Devynias, ne šešias, nes ir du tos pačios rūšies objektai gali būti susieti ryšiu. Jei poromis susiesime visas Lietuvos savivaldybes, tokių ryšių bus 60^2. Kai atsižvelgiama į ryšių pobūdį (metriniai, ranginiai, topologiniai), derinių skaičiai būna labai dideli. Toli gražu ne visos sistemos turi galimybę išsamiai koduoti erdvinius ryšius.

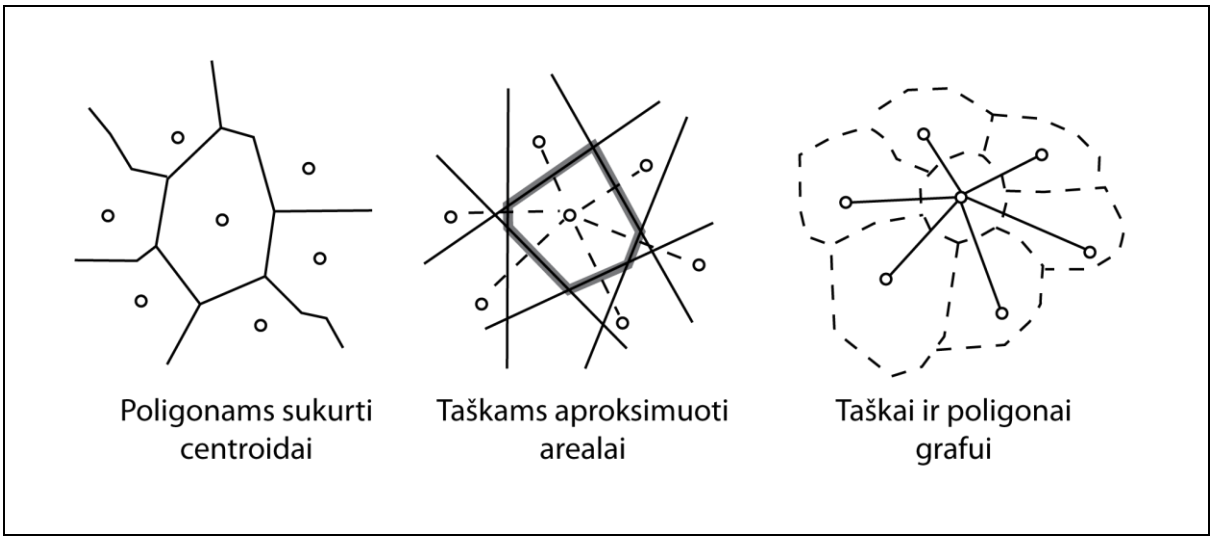
I-3 lentelė. Erdvinių ryšių pavyzdžiai

	TAŠKAS	LINIJA	AREALAS
TAŠKAS	Atstumai, kryptys. *Du miestai yra netoli vienas kito.*	Taško buvimas ant linijos ar kurioje nors jos pusėje. *Miestas yra kairiajame*	Įdėtumas. *Pašto zonoje yra trys mokyklos.*

		upės krante.	
LINIJA	Linijos atstumas nuo taško. *Geležinkelis eina per miestą.*	Sutapimas, sankirta. *Kelias kerta upę.* *Valstybės siena eina upe.*	Sankirta *Gatvė nekerta gamyklos teritorijos.*
AREALAS	Matomumo vietovėje patikrinimas *Iš apžvalgos taško matomas visas miestas.*	Įdėtumas. *Ryšio tinklas yra tik rajono ribose.*	Sąlytis, persidengimas. *Du sklypai turi bendrą ribą.*

Gali būti ne vienas ryšys tarp objektų, o jų tinklas, kuris tuo atveju turi papildomas, tik junginiui būdingas savybes, pavyzdžiui, kelių tarp viršūnių buvimą, jungumą. Taškai gali būti apjungti į gardeles (angl. *lattice*), kurios interpretuojamos kaip tinklas ar reguliari struktūra. Arealai gali sudaryti mozaikas (angl. *tesselation*), pavyzdžiui, politinis žemyno žemėlapis; paviršiai – poliedrus.

Geografinių objektų pakeitimai. Kai kurių tipų objektai gali būti pakeisti kitais, mažesnio ar didesnio matavimų skaičiaus. Taškai ir linijos gali atstoti arealus, nors praktiškai arealo centroidas ar laisvai pasirinkta vieta, reprezentuojanti dvimatį objektą, būna taškas, o tik atskirais atvejais – kontūro linija. Bet atributus visada saugo mažesnio matavimų skaičiaus objektai. Toks požiūris palengvina daugumą matematinių ar kartografavimo operacijų.



I-10 pav. Geografinių objektų pakeitimai ir dualumas

Bet gali būti ir atvirkščiai, kai turint duomenis apie mažesnio matavimų skaičiaus objektus, pavyzdžiui, taškus, galima aproksimuoti linijas arba arealus. Tokia technika naudojama nustatyti statistiniams vienetams, rajonams. Ji naudinga, kai svarbus ne tiek pats arealas, pavyzdžiui, pašto indekso zona, o kelių ar mazgų buvimas jame. Kai kada taškas ir arealas yra dualūs, pavyzdžiui, pašto įstaiga ir jos aptarnaujamas rajonas.

Intensyvūs ir ekstensyvūs duomenys. Intensyvia forma duomenys apie geografinį objektą yra pateikiami tik keletu reprezentacinių parametrų, pavyzdžiui upės vingiai aproksimuojami pagal nedaug taškų, o ne saugant didelį skaičių viršūnių. Taigi, šiuo metodu saugoma palyginti mažai duomenų ir procedūra ar metodas gauti tarpiniams duomenims. Ir atvirkščiai, duomenų bazė,

kurioje saugoma skaitinė ar tekstinė informacija (dažniausiai ne geografinė) paprastai yra ekstensyvi.

Mažai tikėtina, kad visus galimus erdvinius objektus ir jų ryšius būtų galima saugoti duomenų bazėje. Dažniausiai yra saugomos topologinės savybės, tuo tarpu matuojamos savybės, pavyzdžiui, atstumai tarp taškų, yra išvestinės ir jas saugoti nėra prasmės.

Pagrindinis vektorinio modelio trūkumas yra tai, kad jis sudėtingas, be to, su vektoriniais objektais sudėtingiau atlikti daugelį erdvinės analizės operacijų.

Vektorinio modelio privalumai:

- galimos operacijos su atskirais geografiniais objektais ar jų grupėmis;
- duomenys užima kelis ar keliolika kartų mažiau vietos, negu rastras;
- galima efektyviai saugoti labai tikslius, taip pat skirtingo tikslumo duomenis;
- galima saugoti daug ir sudėtingų atributų, pagal juos ieškoti, atlikti analizę.

I.5.3 Mišrūs geografinių duomenų modeliai

Galima pastebėti, kad vektorinis ir rastrinis modeliai pasižymi skirtingais trūkumus ir privalumais vaizuojant geografinius duomenis, t.y., tarsi papildo vienas kitą. Geografinės informacijos specialistai dar ir dabar diskutuoja apie universalaus modelio, turinčio abiejų modelių gerąsias savybes, galimybę. Kol toks modelis nesukurtas, specifiniams tikslams dažnai naudojami nesudėtingi mišrūs modeliai.

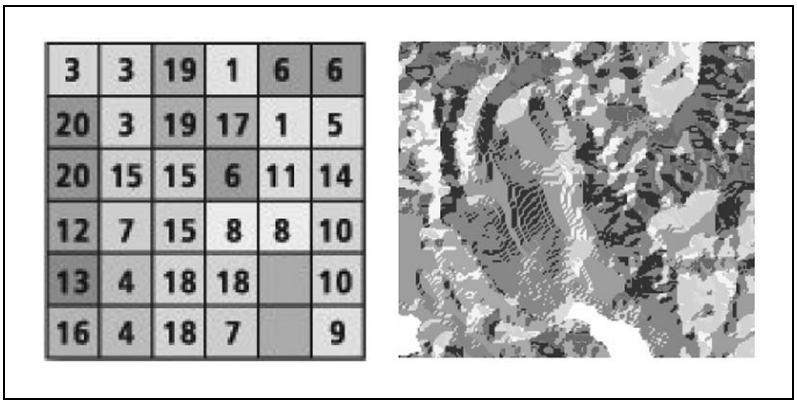
Mozaikos (angl. *tesselation*) – tai rinkiniai diskrečių plotinių objektų, kurie jungiasi tarpusavyje ir tolydžiai dengia erdvės segmentą. Šie objektai gali būti įvairios vienodos ar skirtingos formos, taisyklingi ar netaisyklingi. Juos galima sukurti įvairiais būdais diskretizuojant tolydžią erdvę. Mozaikų pavyzdžiai yra

- žemės sklypai (netaisyklingi),
- topografinio žemėlapio nomenklatūriniai lapai,
- iš anksto sukurti greitai peržiūrai rastrinio žemėlapio lapai,
- šešiakampės gardelės, dar 1933 m. pasiūlytos G. Kristalerio (Gottlieb Christaller) centrinių vietų analizei regionų ir miestų geografijoje ir 1965 m. P. Hageto (Peter Haggett) kaip bendresnis teritorinės analizės metodo pagrindas visuomeninėje geografijoje.

Mozaikų elementai gali būti ir rastriniai vaizdai, tačiau jie taip pat gali turėti susietus atributus, kaip vektorinio modelio plotiniai objektai.

Savitas taisyklingos mozaikos tipas – **geografinis tinklelis** (angl.: *grid*, *lattice*), dažnai naudojamas vaizduoti geografiniams duomenims, yra labai panašus į rastrą modelis, kurio kiekvienas vaizdo elementas yra **gardelė**, susieta su geografinėmis koordinatėmis. Gardelėje taip pat saugoma viena reikšmė, tačiau ne spalva, o kokio nors tolydaus geografinio paviršiaus atributo, pavyzdžiui, reljefo aukščio, žemės dangos tipo, vidutinio metinio kritulių kiekio ar pan., reikšmė (I-11 pav.). Gardelės paprastai būna kvadratinės, jų matmenys priklauso nuo duomenų skiriamosios gebos – centimetrai, metrai, ar kilometrai. Kuo mažesnė gardelė, tuo didesnė skiriamoji geba ir atitinkamai galima pavaizduoti smulkesnius geografinius objektus. Paprastai gardelės dydis parenkamas atsižvelgiant į duomenų matavimo tikslumą, duomenų apimtis, t.y., vietos poreikį duomenų saugykloje, duomenų vaizdavimo ir apdorojimo spartą. Gardelės siejamos su geografinėmis koordinatėmis naudojant geografinio rastro *atskaitos tašką,* kurio geografinės koordinatės žinomos. Žinant, kokį atstumą

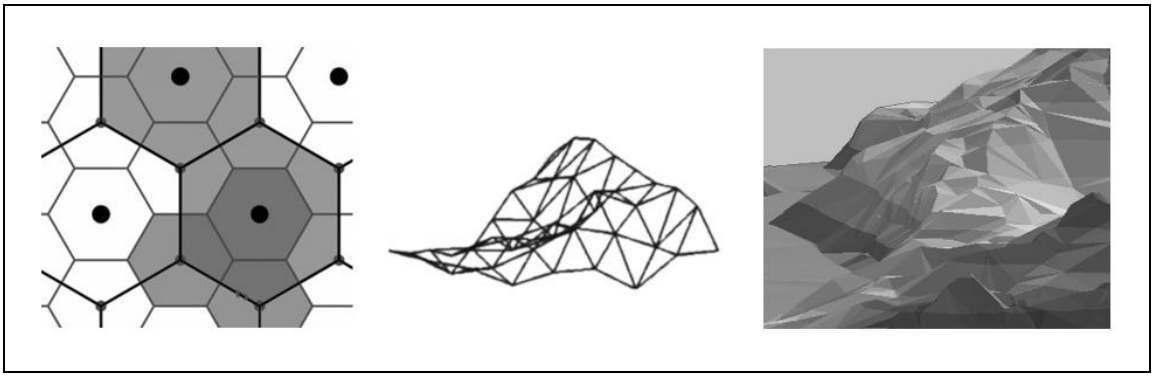
vietovėje atitinka gardelės dydis, ir gardelės padėtį atskaitos taško atžvilgiu, nustatomos bet kurio geografinio rastro elemento koordinatės.



I-11 pav. Geografinio rastro struktūra (kairėje) ir pavyzdys (šlaitų ekspozicijos žemėlapis dešinėje)

Netaisyklingos mozaikos trimatėje erdvėje pavyzdys yra **netaisyklingųjų trikampių tinklas** (angl.: *Triangulated Irregular Network, TIN*). Tai yra nepersidengiančių trikampių, jungiančių nevienodais atstumais išdėstytus taškus, rinkinys, sudarantis tolydų paviršių trimatėje erdvėje. Šis iš esmės vektorinis modelis puikiai tinka modeliuoti reljefui – kiekvienas trikampis atitinka vienodo nuolydžio plotą. Kadangi netaisyklingųjų trikampių tinklu galima vaizduoti netaisyklingai išdėstytus objektus, jais patogu modeliuoti suskaidytus paviršius su dideliais nuolydžio gradientų skirtumais – kalnų viršūnes, skardžius, slėnius, įdubas. Tokiais atvejais šis modelis efektyvesnis ir tikslesnis, negu rastras. Tinklo viršūnės išdėstomos naudojant specialius algoritmus, kurie padeda nustatyti taškus, svarbiausius tiksliam paviršiaus vaizdui sudaryti. Todėl reikia mažiau taškų negu rastrui, kuriame taškai išdėstyti reguliariai.

Sudarant netaisyklingųjų trikampių tinklą naudojama Delonė (Boris Delaunay) trianguliacijos algoritmas, kuris leidžia išvengti labai smailių kampų. Jo esmė ta, kad neturi būti jokių taškų apskritimuose, apibrėžtuose apie trikampius.



I-12 pav. Mišrių geografinių duomenų modelių pavyzdžiai: šešiakampių elementų mozaika (kairėje); netaisyklingųjų trikampių tinklas; tinklo pagrindu sukurtas reljefo modelis (dešinėje)

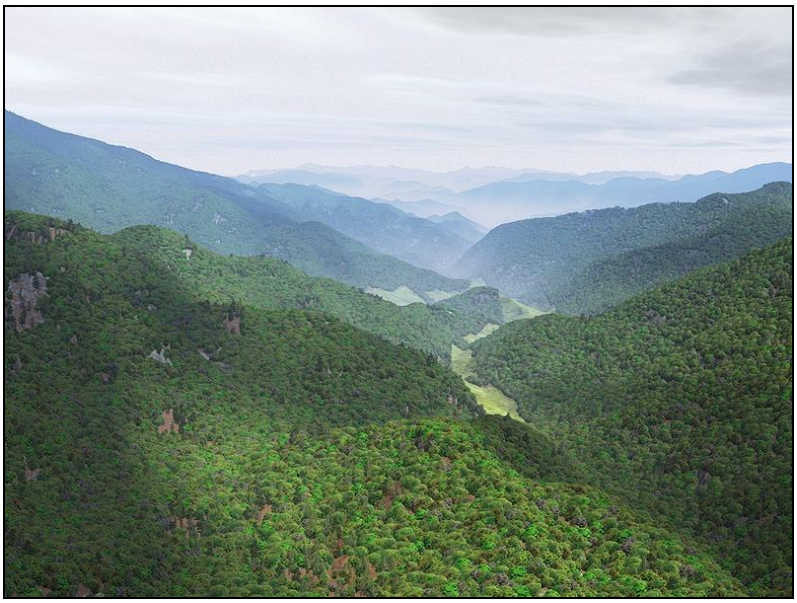
Montažas (angl. *mosaic*) – tai skaitmeninių vaizdų rinkinys ir priemonės, leidžiančios šiuos vaizdus tvarkyti, rasti bei peržiūrėti kaip vieną vaizdą, vientisą ar sudarytą iš fragmentų. Pavyzdžiui, *ArcGIS* montažo duomenų rinkinį sudaro:

- vaizdų katalogas,
- montažo aprėpties riba (vektoriniai duomenys);
- dinaminio montavimo taisyklės,
- operacijų žurnalas,
- kita informacija (atributai reikalingi montažo sudarymui, jei reikia, spalvų korekcijų lentelės, segmentų ribos ir kt.)

Skaitmeniniai paviršiaus modeliai (SPM, angl.: *Digital elevation model*, *DEM*) yra kuriami mozaikų pagrindu kaip geografinis rastras arba *TIN*. Netaisyklingųjų trikampių tinklas laikomas pirminiu paviršiaus modeliu, o geografinio rastro SPM – išvestiniu (apskaičiuotu). Skaitmeniniai paviršiaus modeliai yra pagrindas reljefo žemėlapiams rengti. Jie sudaromi iš nuotolinių tyrimų ar antžeminių matavimų duomenų, taip pat interpoliuojant aukščių izolinijas. Skaitmeninių reljefo modelių kokybei turi įtakos ne tik jų duomenų šaltinis (pradinių aukščio taškų tankis ir pasiskirstymas) ir interpoliavimo algoritmas, bet ir modeliuojamo reljefo sąskaida, rastro gardelės dydis bei vertikali skiriamoji geba.

Skaitmeniniai reljefo modeliai svarbūs GIS vertinant atstumus, analizuojant reljefo morfologinius ypatumus – šlaitų polinkį, vyraujančią orientaciją, modeliuojant vandens tėkmes, potvynius, objektų matomumą ir pan. Jie taip pat svarbūs kartografijai.

Fraktalai (angl. *fractal* – fragmentas, dalis) yra savito tipo geometriniai objektai. Juos pirmą kartą 1975 m. aprašė B. Mandelbrotas (Benoit Mandelbrot), remdamasis daugelio ritmiškai besikartojančių gamtos reiškinių ir objektų stebėjimais ir matematine analize. Fraktalų geometrijos esmė yra objektų fragmentacija ir panašumas į pačius save. Fraktalai – tai skaidomi ir pakartojami geometriniai objektai, pasižymintys panašumu į save tęsiant skaidymą iki begalybės. Fraktalas gali užimti ribotą erdvės plotą, tačiau jo kontūras yra begalinio ilgio, todėl gali būti analizuojamas kiek norima stambiu masteliu. Bet kuri fraktalo dalis yra tokia pat sudėtinga, kiek ir jis pats. Formų panašumu skirtinguose masteliuose pasižymi daugelis geografinių objektų – kranto linijos, kopos, kalnagūbriai, kelių tinklas, gyvenvietės. Keičiant mastelį dažnai išlieka būdingos kontūrų savybės – tai yra struktūros struktūrose. Fraktalų geometrija kur kas geriau, negu Euklido, leidžia pavaizduoti realius objektus, kurių kontūrai netaisyklingi, bet pasikartoja skirtinguose masteliuose ar objekto dalyse, pavyzdžiui, medis, upė. Jas būtų galima modeliuoti ir geometrinėmis figūromis: kūgiais, cilindrais ir kt., tačiau fraktalai leidžia tą padaryti greičiau ir gaunamas vaizdas yra panašesnis į tikrąjį. Fraktalų geometrija naudojama ir kai kurių socialinių reiškinių, ligų plitimui modeliuoti.



I-13 pav. Fraktalais sumodeliuotas kraštovaizdis[1]

Geografinius duomenis galima transformuoti iš vieno duomenų modelio į kitą. Pavyzdžiui, turint vektorinius duomenis – reljefo taškus su aukščio atributu – galima interpoliuoti tarp taškų ir suskaidžius gautą paviršių norimo dydžio gardelėmis sukurti geografinį rastrą. Taip diskretūs duomenys paverčiami tolydaus lauko modeliu. Gautą tolydų lauką galima diskretizuoti kitaip, pavyzdžiui, nubraižant reljefo aukščių izolinijas ar vektorinius objektus, tokius kaip upės ar kalnagūbriai.

I.6 KARTOGRAFINIAI DUOMENYS

Grafinio informacijos vaizdavimo istorija yra ilga, o žemėlapiai – vieni seniausių sudėtingų grafinių vaizdų. Kartografija visą laiką vaidino svarbų vaidmenį kaupiant ir skleidžiant geografinę informaciją apie mūsų planetą, jos gamtą, gyventojus, o vėliau – ir abstrakčią informaciją, kurios svarbus komponentas yra padėtis Žemės, kito dangaus kūno ar virtualiame paviršiuje.

Kartografinio vizualizavimo reikšmė nuolat didėja

- atsirandant naujiems reiškiniams, kuriuos reikia grafiškai pavaizduoti,
- vystantis ir atsirandant naujiems vaizdavimo metodams, tarp jų elektroninėse aplinkose, mobiliuosiuose įrenginiuose,
- dėl šių procesų kuriantis naujai kartografinei semiotikai.

Vizualiai suvokiamas realaus pasaulio duomenų modelis, kuriame užkoduota vaizduojamų reiškinių esmė, yra labai svarbus efektyviam geografinės informacijos valdymui, naudojimui ir sklaidai.

Kartografiniai duomenys – tai geografiniai duomenys, saugomi kartu su išsamia jų vaizdavimo įvairiose aplinkose informacija.

Akivaizdu, kad kartografui, sudarančiam žemėlapį, skirtingai negu geografui, neužtenka duomenų apie objektų realias koordinates ir savybes. Kiekvienam objektui egzistuoja mažiausiai keletas

[1] *© Gary R. Huber, 3D Nature, LLC*

galimų jo vaizdavimo būdų – grafinių objektų, kurie atstovauja realų objektą arba reiškinį skirtingo mastelio ar paskirties žemėlapiuose. Tai grynai kartografiniai duomenys, kurie sudaro didelę duomenų bazės dalį. Kartografinėje duomenų bazėje saugoma geografinė ir visa pagalbinė informacija, kuri panaudojama sudarant žemėlapius. Žemėlapiai komponuojami iš atskirų informacijos sluoksnių, kurių daugelis yra tie patys skirtingiems teminiams žemėlapiams, todėl duomenų bazėje saugomą geografinę informaciją turi būti galima sutvarkyti ir žemėlapių sluoksnių pavidalu. Be to, daug kartų naudojami tie patys sutartiniai ženklai, spalvų skalės ir kita pagalbinė informacija. Taisyklės, kaip atskiri komponentai jungiami ir derinami kuriant galutinį produktą, taip pat yra kartografinių duomenų dalis. Kitas kartografinių duomenų komponentas – kartografinė projekcija, t.y., funkcija, atvaizduojanti geografines koordinates (ilgumą, platumą) į plokštumos taškus. Realius geografinius duomenis visada galima pavaizduoti norima projekcija naudojant atitinkamą parametrų rinkinį. Grįžimas prie geografinių koordinačių, taip pat automatinis perskaičiavimas iš vienos projekcijos į kitą gali būti atliktas GDBVS, su sąlyga, kad sistema sugeba saugoti ir apdoroti tų projekcijų parametrų duomenis. Taip kartografiniai duomenys tampa, ko gero, sudėtingiausia duomenų klase. Šiuolaikinės GIS sistemos turi didesnes ar mažesnes galimybes saugoti kartografinio vaizdavimo informaciją, bent jau žemėlapio maketą, sutartinius ženklus, spalvų schemas. Pagrindiniai dalykai, dėl kurių kartografiniai duomenys ir šiuo metu valdomi neefektyviai, yra susiję su daugkartiniu geografinių duomenų naudojimu skirtingame kontekste ir jų specifiniu vizualizavimu.

Geografinių duomenų vizualizavimas pats savaime yra techniškai sudėtingas, nes trimačiai duomenys vaizduojami plokštumoje stipriai sumažinti. Pradiniai geografiniai duomenys juos vaizduojant yra iškraipomi šiais aspektais.

- Kartografinė projekcija, kurios pagalba trimačio Žemės paviršiaus geografinės koordinatės (ilguma ir platuma) atvaizduojamos plokštumoje, iškreipia atstumus, kryptis, formas ir dydžius.
- Generalizavimas, neišvengiamas vaizduojant duomenis smulkesniu masteliu, negu jie buvo sukaupti. Generalizuojant atsisakoma informacijos, kuri konkrečiu atveju yra perteklinė, be to, neretai kokybiškai pasikeičia objektai, pavyzdžiui, atskiri pastatai turi būti apjungiami į vieną plotą – miesto teritoriją, o dar labiau generalizuojant, plotas keičiamas sutartiniu ženklu. Be abejo, vaizduojant teritoriją, kurios padėties duomenys yra daugelio kontūro taškų koordinačių sąrašas, ženklu, kurio padėtį apibrėžia vienintelis taškas, prarandama labai daug informacijos.
- Kiti sąmoningi iškraipymai, gerinantys žemėlapio skaitomumą, estetinį vaizdą ir pan. Jie taip pat dažnai susiję su generalizavimu, pavyzdžiui, miesto sutartinio ženklo centro patraukimas į vieną ar kitą upės pusę, jei objekto, esančio vienoje upės pusėje, ženklas dengia upės liniją žemėlapyje, mieste esančių turistinių objektų ženklų išdėstymas šalia miesto ženklo, ir pan.

Todėl kiekvienas kartografas susiduria su praktiškai neišsprendžiama problema. Iš vienos pusės, pavaizduoti duomenys saugomi duomenų bazėje ir atspindi realią situaciją, juos galima naudoti tiksliems matavimams, analizės operacijoms ir pan. Iš kitos pusės, žemėlapis niekada nėra tikslus tų duomenų vaizdas. Jei žemėlapis naudojamas atskirai nuo duomenų bazės, jį galima laikyti klasikiniu popieriniu žemėlapiu su visais jo trūkumais. Tačiau šiuolaikinės technologijos sukurtos būtent tam, kad per žemėlapio vaizdą būtų galima pasiekti realius, tikslius ir neiškraipytus duomenis. Tam, kad naudotojas gautų maksimalią naudą iš duomenų produkto, turi būti išsaugotas abipusis ryšys: didelio tikslumo duomenų bazė – gražus, tačiau iškraipytas kartografinis vaizdas. Šis ryšys būtinas ir tada, kai numatomas žemėlapio vaizdo atnaujinimas, pasikeitus duomenims, t.y., turint omenyje, kad vis daugiau žemėlapių teikiama Internete ir atspindi visuomenei aktualią informaciją, beveik visada. Jei

jo nėra, pasikeitus duomenims, kiekvieną kartą reikia atlikti naujų duomenų vizualizavimą, kartu su būtinais jų iškraipymais.

Pirmą kartą abipusės sąsajos galimybė (kartografinė reprezentacija) atsirado tik 2007 metais vienos populiariausių komercinių GIS sistemų *ArcGIS* 9.2 versijoje. Iki tol kartografai buvo priversti arba kurti grafiškai gana primityvius ar dizaino požiūriu ne visai korektiškus žemėlapius su GIS programine įranga, arba perkelti geografinius duomenis į grafinio redagavimo programų aplinkas tam, kad būtų galima pasinaudoti jų teikiamomis galimybėmis, tačiau prarasdami vėlesnio duomenų apdorojimo galimybę. Dabar jau galima teigti, kad GIS sistemos naudojamos kurti kokybiškiems pasirinkto konkretaus mastelio kartografiniams kūriniams. Sudėtingiau yra žemėlapio elektroninių paslaugų atveju, kai mastelis keičiamas naudotojo. Tada kiekvienam masteliui turi būti parengtas skirtingo detalumo vaizdas išlaikant vieningą tokio elektroninio žemėlapio stilių.

Didžiąją dalį kartografijoje naudojamos informacijos sudaro geografinė informacija, tai yra, objektai, turintys apibrėžtą ir išmatuojamą padėtį erdvėje, kaip vieną iš atributų ir vaizduojami žemėlapiuose arba naudojami parengti įvairiems kitiems kartografinių kūrinių komponentams. Tačiau be jos yra naudojami ir kitokie duomenys.

1. Skaitiniai ir tekstiniai duomenys.

- Koordinatės – geodezinių matavimų, GPS ir kiti duomenys, t.y., duomenys apie padėtį erdvėje ir/arba laike. Jie vėliau paprastai transformuojami į geografinę informaciją.
- Skaitinė atributinė informacija – tai įvairūs kiekybiškai išreikšti duomenys apie geografinius objektus. Jos rūšis yra statistiniai duomenys – teritoriškai pasiskirsčiusių objektų apibendrintos charakteristikos.
- Tekstinė atributinė informacija – įvairūs kokybiškai išreikšti duomenys apie geografinius objektus, pavyzdžiui, objektų vardai, kodai.

2. Geografinė informacija.

Skaitmeninė geografinė informacija gali būti gaunama tiesiogiai arba įvairiais metodais panaudojant anksčiau sukauptas duomenų bazes. Tai gali būti vektorinių geografinių duomenų rinkiniai, kosminės ar kitais nuotoliniais metodais gautos nuotraukos, popieriniai žemėlapiai ar brėžiniai, skaidrės, fotogrametrinė informacija. Kartografinius produktus dažnai sudaro skirtingų tipų geografinė informacija, todėl juos apibūdinant struktūriškai yra patogiau operuoti ne duomenų tipais, o komponentais. Pagrindiniai komponentai yra du.

- Žemėlapis yra įvairių geografinių objektų informacijos rinkinys, kuris turi ir bendras, visam rinkiniui būdingas savybes. Žemėlapio pagrindinės savybės yra pavadinimas, tipas (topografinis, bendrasis geografinis, teminis ir pan.), tema, sudėtingumo klasė (schema, apžvalginis žemėlapis, detalus žemėlapis), vaizduojamas teritorinis vienetas (jie kategorizuojami ir indeksuojami įvairiais metodais – kodais, geografinėmis koordinatėmis ir kt.), mastelis.
- Geografinės informacijos sluoksnis. Tai elementarus žemėlapis, vaizduojantis tik vieno tipo geografinius objektus. Sluoksniai transformuojami ir panaudojami sudaryti įvairių mastelių žemėlapiams. Patogu saugoti keletą skirtingu lygiu generalizuotų vienodos informacijos sluoksnių, kuriuos transformavus galima sudaryti įvairių mastelių ir mastelį atitinkančio sudėtingumo žemėlapių. Projektuojant duomenų bazę, reikia iš anksto numatyti, kokio tipo vektoriniai duomenys ir keliais variantais bus saugomi.

3. Sutartiniai ženklai.

Kiekvienam geografiniam objektui tui būti nurodyta, kaip jis vaizduojamas žemėlapyje. Sutartinių ženklų rinkiniuose pateikiami sugrupuoti pagal objektų klases ar sluoksnius sutartinių ženklų etalonai, stiliai, spalvų skalės ir kita kartografiniam vaizdavimui naudojama informacija. Tokie rinkiniai gali būti ir spaudiniai, ir skaitmeniniai. Sukurti sutartiniai ženklai naudojami pakartotinai, pavyzdžiui, skirtingiems žemėlapiams ir atlasams parengti.

4. Kita grafinė informacija.

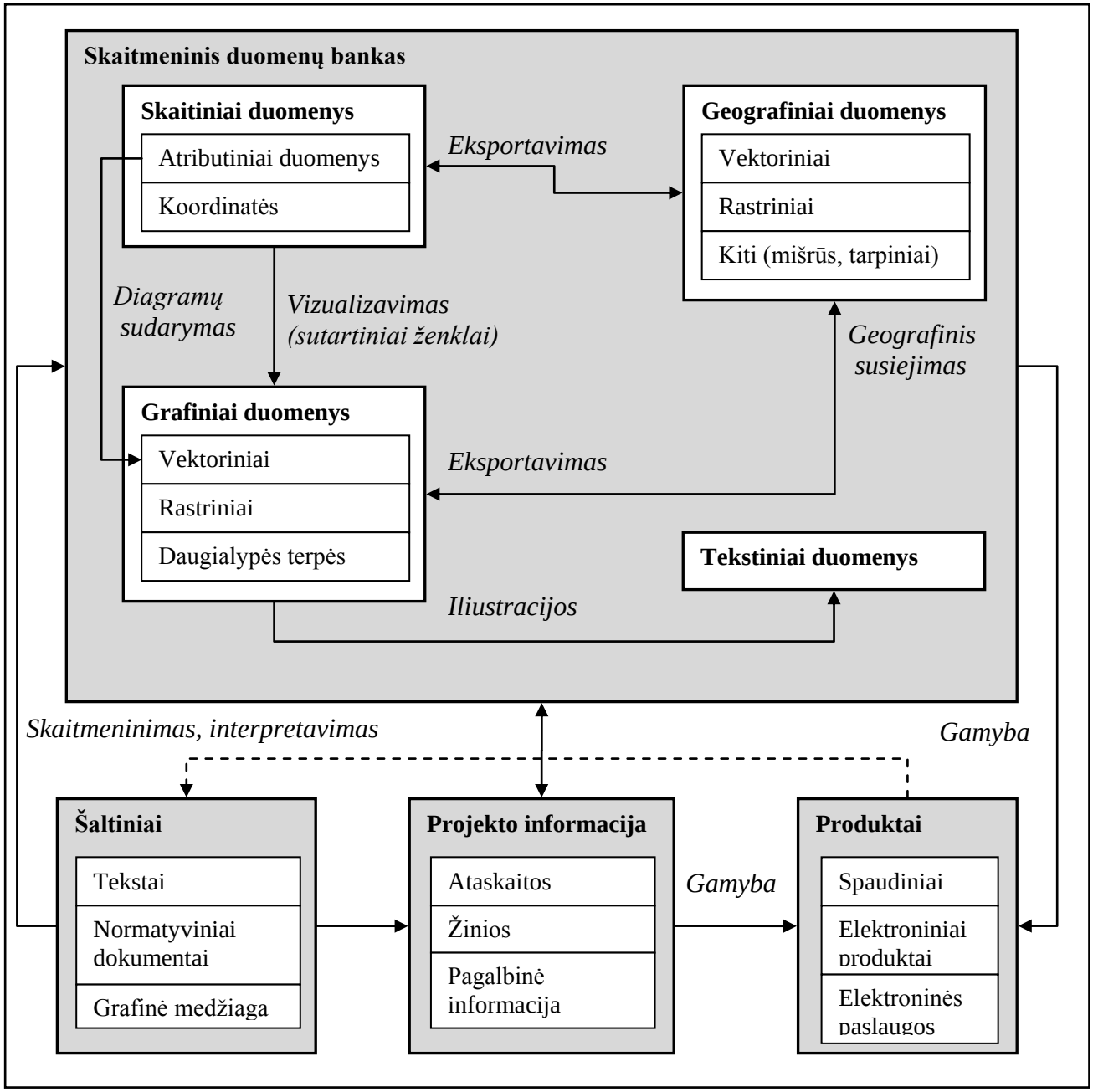
- Iliustracijoms skirtos nuotraukos, piešiniai.
- Diagramos, grafikai ir kiti negeografiniai objektai, paprastai generuojami iš skaitinių duomenų.

5. Tekstai.

Tai žemėlapių aiškinamieji ir aprašomieji tekstai, kurie laikomi struktūriškai vientisais, nors juose gali būti skaitinių duomenų lentelės, iliustracijos ir pan.

6. Daugialypės terpės (angl. *multimedia*) **objektai**.

Ši sąvoka apima priemones pateikti informacijai šiuolaikinių technologijų priemonėmis, kompleksiškai veikiant skirtingus žmogaus pojūčius. Juos galima panaudoti tik skaitmeniniuose variantuose, – tai, pavyzdžiui, garso arba vaizdo įrašai, animuoti objektai ir pan.



I-14 pav. Geografinio atlaso informacijos transformavimo sąryšiai

7. Informacinės sistemos produktai.

- Galutiniai produktai. Tai žemėlapiai ir kiti leidiniai (dažniausiai spaudiniai), kurie publikuojami ir platinami naudotojams, toliau nesirūpinant juose pateiktos informacijos atnaujinimu. Kalbant apie teminės kartografijos projektus ir jiems naudojamus duomenis sudėtingiausias atvejis yra atlasas, kuriame mažų mažiausiai turi būti įvairūs skirtingų autorių sudaryti teminiai žemėlapiai, juos papildantys grafikai, diagramos, aiškinamieji tekstai, iliustracijos, o skaitmeniniame atlaso variante – dar daugiau skirtingų komponentų.
- Nuolat atnaujinami produktai. Tai gali būti elektroninės geografinės informacijos paslaugos, Internete publikuojami žemėlapiai ar kita parengta ir naudotojui pateikiama

(dažniausiai skaitmeniniu pavidalu) geografinė informacija, už kurios aktualumą jos kūrėjas lieka atsakingas.

8. Pagalbinė informacija.
Tai produkto informacija, tiesiogiai nesusijusi su geografinių objektų vaizdavimu, bet reikalinga užtikrinti svarbioms jo palaikymo funkcijoms. Jai priklauso programų moduliai (programos tekstas ir kodas, naudojamas skaitmeninių žemėlapių funkcijoms palaikyti) bei skaitmeninio vaizdo valdymo ir modifikavimo priemonės – skaitmeniniai šriftai, meniu, įrankių paletės, ir kiti panašūs objektai.

Skirtingos šios informacijos klasės gali būti gaunama viena iš kitos ar kitaip susiejama tarpusavyje standartiniais procesais. Kartografinių duomenų, naudojamų teminiam atlasui parengti, klasių sąryšis parodytas paveiksle (I-14 pav.), ties perėjimo ryšiais nurodant procesą, kuris transformuoja duomenis.

I.7 METADUOMENYS

Metaduomenys – tai informacija apie duomenis, atsakanti į klausimus apie duomenų paskirtį, kokybę, ryšius ir kitas savybes. Metaduomenys reikalingi tam, kad būtų galima teisingai interpretuoti duomenų bazės informaciją. Jais naudojantis priimami strateginiai sprendimai, pavyzdžiui, ar apskritai verta įsigyti tam tikrus duomenis, ar jie bus suderinami su jau turimais, ir pan. Taigi, metaduomenys – tai duomenų modelio dalis, kuri yra skirta duomenų naudotojui tam, kad jis galėtų duomenis greitai identifikuoti, spręsti apie jų tinkamumą naudoti skirtinguose kontekstuose ir tinkamai juos pasirinkti neanalizuodamas pačių duomenų rinkinių. Taip pat naudotojas turi gauti informaciją apie duomenų rinkinio kainą ir naudojimo sąlygas. Šiuolaikinės duomenų bazės paprastai yra didelės apimties ir sudėtingos, norint išsamiai su jomis susipažinti, reikia daug laiko. Todėl ypač svarbu, kad būtų galima greitai nuspręsti, ar duomenys yra visiškai tinkami tam tikslui, kuriuo juos norima įsigyti.

Visai vieningo ir išsamaus metaduomenų apibrėžimo nėra, tačiau įvairiuose šaltiniuose pateikiamos apibrėžimo versijos atspindi pagrindines šios informacijos kategorijos savybes:

- metaduomenys aprašo duomenų savybes,
- metaduomenys leidžia duomenis rasti, identifikuoti, interpretuoti, įvertinti ir valdyti,
- metaduomenys yra struktūrizuoti, o vis dažniau ir standartizuoti,
- metaduomenys yra vieši,
- yra minimali metaduomenų aibė, privaloma konkretiems duomenims, ir pakankamai daug neprivalomų metaduomenų.

Pradėjus naudoti skaitmenines technologijas, metaduomenys pradėti aprašyti formaliai ir pateikiami kaip struktūrizuota skaitmeninė informacija. Vis dėlto nėra universalaus sprendimo, kaip jie turi būti saugomi. Šuo metu populiarūs metaduomenų formatai, kuriuos patogu skaityti ir suprasti žmogui, koks, pavyzdžiui, yra XML. Tokius metaduomenis galima redaguoti nenaudojant specialių įrankių. Tačiau tokie formatai nepritaikyti taupyti vietai, todėl užima daugiau kompiuterio atminties ir lėčiau perduodami ryšio tinklais. Todėl kartais pasirenkamas kitoks, mašininis formatas.

Metaduomenys gali būti saugomi kaip neatskiriama duomenų bazės dalis, t.y., tame pačiame faile, kaip ir duomenys (**vidiniai** metaduomenys), arba atskirai nuo duomenų (**išoriniai** metaduomenys). Abu metodai turi privalumų ir trūkumų:

- vidinius metaduomenis patogiau naudoti, nes jie visada pasiekiami kartu su jų aprašomais duomenimis, tačiau nemažai informacijos kartojasi, be to, prireikus bus sunku surinkti visus duomenų bazės metaduomenis į vieną vietą;
- išoriniai metaduomenys saugomi vienoje vietoje, todėl patogesni atliekant paiešką, jie nedubliuojami, tačiau jų sąsaja su aprašomu duomenų šaltiniu silpnesnė. Pavyzdžiui, pakeitus aprašomą duomenų rinkinį kitu, kuris vadinasi taip pat, bet yra kitokios struktūros, yra rizika, kad išoriniai metaduomenys taps klaidingi.

Duomenų bazių terminologijoje vidinių metaduomenų rinkinys vadinamas **katalogu**.

Geografijoje metaduomenys ypač svarbūs tiek fiziniams ir juridiniams asmenims, kurie naudojasi metaduomenų teikiama informacija, tiek ir valstybės sektoriaus organizacijoms ir organizacijoms tiesiogiai dirbančioms su erdvine informacija. Pavyzdžiui, žemėlapio legenda yra metaduomenų pavyzdys, teikiantis informacija apie žemėlapio leidėją ir išleidimo datą, mastelį, tikslumą, referencinį pagrindą ir kitas žemėlapio charakteristikas. Metaduomenys taip pat dažnai naudojami spausdintų žemėlapių versijoms apibūdinti. Tokiu pačiu principu, per metaduomenis, apibūdinami skaitmeniniai geografiniai duomenys, saugomi kaip duomenų rinkinių versijos, pavieniai duomenų rinkiniai, duomenų rinkinių dalys arba pavieniai geografiniai objektai. Pagrindinė erdviniams metaduomenims būdinga savybė yra atsakymas į klausimą “kur?“.

Metaduomenų standartas leidžia ne tik rasti, įvertinti ir panaudoti egzistuojančius duomenis, bet ir nusako erdvinių duomenų platinimo metodą. Spaudinių eroje geografiniai duomenys buvo perduodami tik per žemėlapius. Dabar praktiškai kiekvienas naudotojas gali juos gauti iš skirtingų elektroninių šaltinių. Yra du pagrindiniai duomenų bazių perdavimo būdai.

- „Off-line“ metodas, kai teikėjas perduoda duomenų kopiją ir tampa nebeatsakingu už duomenis. Taip vyksta spaudinio ar kompaktinio disko perdavimas.
- Duomenys teikiami per ryšio priemones (paprastai Internetą) ir naudotojas visada gauna jų naujausią versiją („on-line“ metodas)..

Norint, kad gavėjas teisingai perskaitytų jam skirtus duomenis, reikia konvertuoti juos iš šaltinio formato į gavėjo formatą. Taigi, duomenų perdavimo standartizavimas tampa dar svarbesnis.

Pavyzdys. Geografinių metaduomenų tuometinę sampratą iliustruoja JAV Geologijos tarnybos sukurtas *DLG* (angl. *Digital Line Graph*) modelis, kuriame numatyti tokie geografinių duomenų specifikavimo skyriai:

– objekto vardas,
– atributų ir jų domenų sąrašas,
– kartografinis vaizdavimas (angl. *delineation*),
– vaizdavimo taisyklės,
– derinimo su kitais objektais taisyklės,
– ryšių su kitais objektais tipai,
– panaudojimo taisyklės.

Pirmieji metaduomenų dokumentai, nurodantys, kaip turi būti aprašomi **geografiniai duomenys**, buvo sukurti apie 1993 m. JAV Federalinio geografinių duomenų komiteto. Ilgą laiką pasaulyje buvo lygiagrečiai vystoma daugelis metaduomenų standartų, iš kurių svarbesni yra du.

- Skaitmeninių geoerdvinių metaduomenų turinio standartas (angl. *Content Standard for Digital Geospatial Metadata*, http://www.fgdc.gov/metadata/csdgm), priimtas 1994 metais JAV, taip pat galiojęs Didžiojoje Britanijoje, Kanadoje, Pietų Afrikos Respublikoje, Japonijoje, kai kuriose Lotynų ir Pietų Amerikos šalyse. Tai dabartinis JAV federalinis standartas.
- Europos norminimo komiteto standartas *CEN* (pranc.: *Comité Européen de Normalisation*, http://www.cenorm.be), priimtas 1998 m. ir naudotas daugelyje Europos šalių.

Bet koks metaduomenų standartas ar aprašas nurodo privalomus duomenų specifikacijos skyrius. Apskritai siekiant išspręsti geografinių duomenų nesuderinamumo problemą, duomenų ir metaduomenų specifikavimą siekiama standartizuoti. Standartiniai metaduomenų komponentai yra informacija apie duomenų bazės šaltinį, koordinačių sistemą, būseną (vientisumą, išsamumą, prieinamumą), kokybę, istoriją, perdavimo strategiją, informacija apie organizacinius kontaktus, bei informacija apie pačius metaduomenis, pavyzdžiui, kalba, kuria jie pateikiami. Pageidaujama, kad metaduomenys būtų automatiškai skaitomi ir interpretuojami bei palyginami tarpusavyje, todėl ypač svarbus jų pasaulinis standartizavimas. Šiuo metu ISO tarptautinis standartas http://www.isotc211.org yra pripažįstamas kaip pasaulinis geografinės informacijos standartas. Standarte ISO 19115 „Geografinė informacija – metaduomenys“ yra pateikiamas detalus geografinių duomenų metaduomenų modelis su išsamias metaduomenų elementų rinkiniais.

Metaduomenys gali būti skirtingų abstrakcijos lygmenų.

1. Visos duomenų bazės (arba spaudinių žemėlapių serijos).
2. Produkto (pavyzdžiui, vieno žemėlapio, aeronuotraukos ir pan.).
3. Duomenų grupės (pavyzdžiui, žemėlapio sluoksnio).
4. Objektų kategorijos (pavyzdžiui, kelių arba miestų).
5. Konkretaus objekto (pavyzdžiui, Vilniaus miesto).

Duomenų bazės lygmeniu minimalią pageidaujamą metaduomenų aibę sudaro šios grupės.

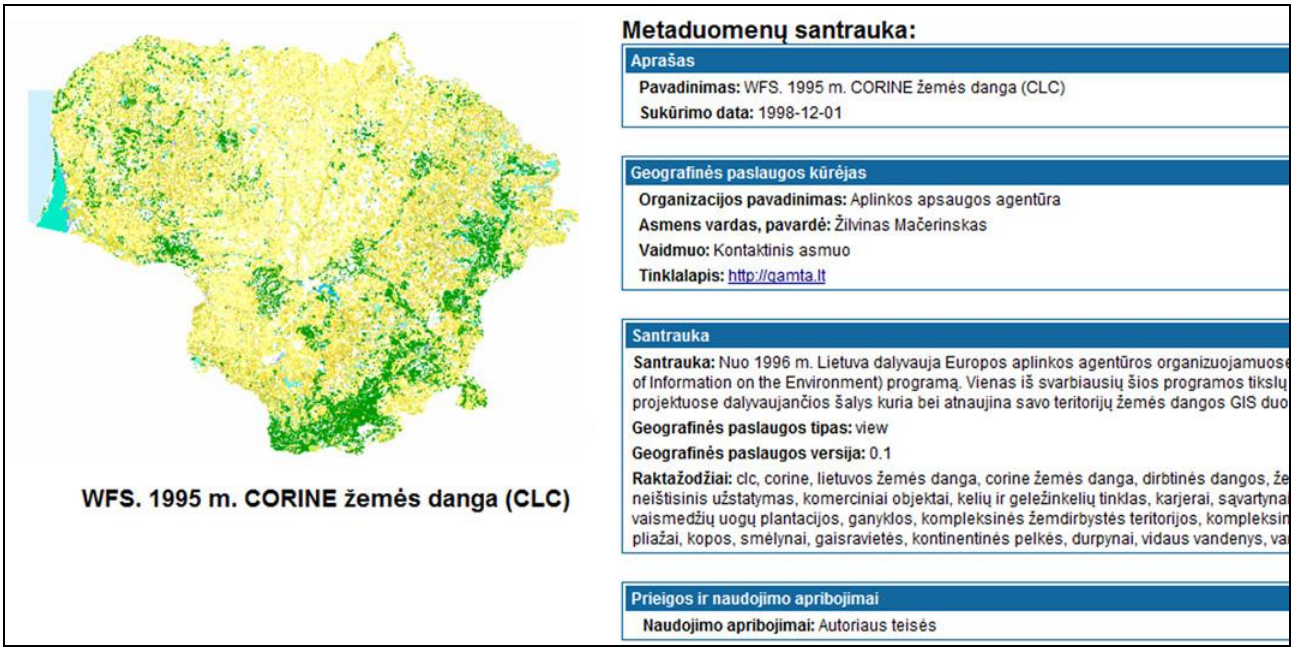
1. Unikalus identifikatorius.
2. Duomenų teikėjas – organizacija, platinanti duomenis būtent ta forma. Tai gali būti leidykla, topografinė tarnyba, savivaldybė ir pan.
3. Duomenų autorius aba organizacija, atsakinga už originalių duomenų pateikimą.
4. Kiti kontributoriai.
5. Nuorodų sistemos (erdvė, vieta, semantiniai apibrėžimai).
6. Duomenų apimtis. Turi būti nusakoma apimama erdvė (pavyzdžiui, objektus apimantis stačiakampis), laikas (pavyzdžiui, galiojimo periodas), ir semantinė apimtis (pavyzdžiui, objektų ar jų tipų sąrašai).
7. Tekstinis aprašymas su pavyzdžiais.
8. Data.
9. Metaduomenų kalba.
10. Duomenų perdavimo formatas ir tvarkymui, peržiūrai ir perdavimui reikalinga programinė įranga.
11. Duomenų kokybė (edvės, laiko ir semantinės kokybės rodikliai, nurodant, kaip jie išmatuoti).
12. Ryšiai su kitais duomenimis.
13. Teisės ir valdymas (autorinės teisės, naudojimo ir platinimo būdai bei apribojimai).

Žemesniais lygmenimis saugoma gali būti, pavyzdžiui, dar ir tokia informacija:

1. Komponentų atributų ir jų reikšmių aibių sąrašas.
2. Panaudojimo taisyklės. Tai informacija apie tai, kokiems tikslams komponentas gali būti korektiškai panaudotas (pvz., informacijos sluoksnis – Atlaso teminiam, sieniniam žemėlapiui, reklaminio leidinio kartografinei iliustracijai ir pan.).
3. Derinimo su kitais komponentais taisyklės. Jos apibrėžia santykius tarp kartu vaizduojamų komponentų (pavyzdžiui, informacijos sluoksniui – virš kurių sluoksnių jis vaizduojamas sudarant žemėlapį).
4. Transformavimo taisyklės. Jos nusako, kaip vykdomas generalizavimas, kiek galima didinti ar mažinti objektus, kaip objekto dydis priklauso nuo žemėlapio mastelio ir kt.

Tam tikri modeliavimo įrankiai, pavyzdžiui, *Sybase PowerDesigner* leidžia organizacijoms sistemiškai kaupti, integruoti ir tvarkyti metaduomenis. Jie apima duomenų modeliavimą, verslo procesų modeliavimą ir UML, leidžia konkrečiame kontekste valdyti ir analizuoti verslo procesų, duomenų, informacijos pokyčių įtaką, taip pat pagal metaduomenis gali automatiškai sugeneruoti kodą, suprantamą rinkoje žinomiems duomenų bazių serveriams, bei taikomųjų programų kūrimo ir vystymo platformoms. Bendri organizacijos metaduomenys saugomi jos duomenų saugykloje ir sieja duomenų bazių valdymo sistemas, duomenų rinkinių kopijas, transformavimo įrankius, paskirstytus duomenis bei taikomąsias programas. Ši informacija labai svarbi verslo procesų analizei ir modeliavimui.

Lietuvos metaduomenų faktinį standartą – **Nacionalinį metaduomenų profilį (NMDP)** sudaro 62 metaduomenų elementai. Šie elementai yra paimti iš standartų ISO 19115 "Geografinė informacija – Metaduomenys", ISO 19119:2005 "Geografinė informacija – Paslaugos", ISO 19139 „Geografinė informacija – Metaduomenys – XML schemos kūrimas (angl. *implementation*)" ir pritaikyti tenkinant Lietuvos nacionalinio metaduomenų standarto poreikius taip, kad ISO 19115 standarte pateikti privalomieji laukai yra privalomieji ir NMDP. Duomenų produktui aprašyti užtenka mažiau metaduomenų elementų – būtinoji aibė yra 24 elementai iš 62 esančių NMDP. Šie 24 elementai yra vadinami **kamieniniais**.



I-15 pav. Metaduomenų peržiūros lango fragmentas LEI portale (www.geoportal.lt)

Siekiant kuo aiškiau pateikti metaduomenų elementais kaupiamos informacijos pobūdį, elementai susiję su ta pačia informacija yra sugrupuoti į aštuonis katalogus ir vieną atskirą skyrių skirtą pagrindinių metaduomenų elementų aprašymui papildančiais metaduomenų elementais:

1) metaduomenų rinkinio informacija;
2) duomenų identifikavimas;
3) duomenų apribojimai;
4) duomenų kokybės informacija;
5) duomenų priežiūros informacija;
6) duomenų erdvinio vaizdavimo informacija;
7) duomenų referencinės sistemos informacija;
8) duomenų platinimo informacija;
9) papildantys metaduomenų elementai.

NMDP sudarantys kamieniniai (privalomieji) metaduomenų elementai struktūrizuoti taip, kad visų pirma padėtų naudotojui rasti duomenis, įvertinti jų turinį ir geografinę aprėptį bei tinkamumą įvairiems tikslams. Kiti NMDP metaduomenų elementai yra parinkti atsižvelgiant į duomenų tipus ir metaduomenų kaupimo tikslus, į užsienio šalių nusistovėjusią praktiką, geriausius pavyzdžius bei esamą GIS duomenų naudojimo situaciją Lietuvoje. NMDP nedraudžia naudoti papildomų ISO 19115 standarto metaduomenų elementų, kurie nepateko į NMDP. Su visu NMDP dokumentu galima susipažinti Lietuvos erdvinės informacijos infrastruktūros portale www.geoportal.lt (Metodinė informacija → Metaduomenų standartai).

Metaduomenų įrašai turėtų būti kuriami tuo metu kai duomenys yra renkami arba dar anksčiau, planavimo etape. Metaduomenų kūrimo procese turėtų dalyvauti specialistai, turintys detalias žinias apie kuriamus duomenis. Rengiant metaduomenų įrašą tam tikram duomenų rinkiniui, žodžių ir minčių formuluotės turi būti kruopščiai apgalvotos. Kiekvieno elemento turinys turi būti aiškus ir glaustas tiek, kad leistų lengvai rasti ieškomą duomenų rinkinį. Tai ypač svarbu pildant tokius elementus kaip “Pavadinimas” ir “Santrauka“. Metaduomenų įrašas turi būti skaitomas ir suprantamas ne tik techninį pasirengimą turintiems, bet ir eiliniams geografinės informacijos naudotojams.

Klausimai diskusijai

Pagalvokite, su kokiomis duomenų bazėmis susiduriate kasdien, kokiu būdu jas naudojate. Kokia tų duomenų bazių dalis yra ar galėtų būti geografinės? Ar naudojate metaduomenis?

Užduotys savarankiškam darbui

Susipažinkite su geografinių duomenų produktais – erdvinių duomenų rinkiniais ir elektroninėmis paslaugomis (geoproduktais) ir jų metaduomenimis, teikiamais Lietuvos geografinės informacijos infrastruktūros portale www.geoportal.lt. Išnagrinėkite Lietuvos nacionalinį metaduomenų profilį aprašantį dokumentą.

Internete raskite Lietuvos skaitmeninį geografinių duomenų arba žemėlapių rinkinį ir pabandykite jį aprašyti pagal pavyzdyje pateiktus metaduomenims keliamus reikalavimus. Susipažinkite su žemėlapių pateikimo Internete būdais ir įvertinkite Interneto žemėlapių rinkinių metaduomenų kokybę.

Užduotys praktikos darbams

Susipažinkite su *MS Access* DBVS aplinka: programos paleidimu, pagalbos funkcija, pagrindiniais komponentais (lentelės, užklausos, formos, ataskaitos). Atlikite duomenų paiešką, rikiavimą, filtravimą. Susipažinkite su elementariais atributų tipais (skaičiai, simbolių eilutės, loginiai, datos, automatinis numeris). Susipažinkite su įvedimo šablonais (angl. *input mask*) ir jų naudojimu, patvirtinimo taisyklės predikatais ir loginėmis jungtimis bei patvirtinimo tekstais.

Išnagrinėkite likusius atributų tipus: pasirinkimo sąrašas (pagal įvestas reikšmes, iš turimos lentelės), hipertekstinė nuoroda (įterpimas, elementai: tekstas, adresas, žymė, sufleruojamas tekstas), OLE objektai: įterpti (angl. *embedded*) ir susieti (angl. *linked*) objektai. Išnagrinėkite likusias atributų savybes: antraštė, numatytoji reikšmė, privalomumas, indeksas.

MS Access aplinkoje sukurkite lentelę „Studentai“, kurioje būtų kaupiami svarbiausi duomenys apie studentus. Nurodykite atributų tipus, sukurkite jų įvedimo šablonus ir patvirtinimo taisykles. Įveskite duomenis, įsitikinkite, kad sukurta lentelės struktūra yra patogi naudoti. Pagalvokite, kokios turėtų būti lentelės, jei norėtumėte kaupti duomenis ir apie studentų kasmet rašomus kursinius darbus, pomėgius, aplankytas šalis.

II. DUOMENŲ BAZĖS, GEOGRAFIJA IR KARTOGRAFIJA

Ateities kompiuteris tikriausiai svers ne daugiau kaip 1.5 tonos.
"Populiarioji mechanika", 1949

II.1 DUOMENYS NESKAITMENINĖSE INFORMACINĖSE SISTEMOSE

Duomenų modeliu vadiname loginę struktūrą, kuria remiantis duomenys kaupiami, saugomi ir naudojami. Pavyzdžiui, teminio žemėlapio duomenis galime įsivaizduoti suskirstytus į sluoksnius pagal vaizduojamus objektus, suskirstytus lapais pagal teritoriją, arba sudėtus į keletą aplankų pagal duomenų šaltinius. Tai ir yra trys skirtingi duomenų modeliai. Principinio skirtumo tarp duomenų modelių skirtingose informacinėse sistemose, gyvavusiose iki masinės kompiuterizacijos, prasidėjusios paskutiniais 20 amžiaus dešimtmečiais, nebuvo. Skyrėsi tik duomenų saugojimo ir panaudojimo efektyvumas, kuris priklausė nuo konkrečios organizacijos priimtos strategijos.

Duomenys patekdavo į organizacijas dviem pavidalais:

- Tekstinė ir skaitinė informacija (geodeziniai, statistiniai duomenys, tekstai, dokumentai ir pan.);
- grafinė informacija – ant popieriaus; ji vėliau turėjo būti kopijuojama, atrenkama ir panaudojama (žemėlapiai, planai, aerofotonuotraukos ir pan.).

Pagrindiniai produktai, sukuriami teminės kartografijos procesų, buvo išskiriami tokie:

- žemėlapio eskizas (grafiškai išreikšta idėja),
- žemėlapio maketas (grafinis vaizdas, kurį dar reikia tvarkyti, atrinkti informaciją ir pan.),
- autorinis originalas (užbaigtas žemėlapis paruoštas tiražavimui).

Kaip papildomas produktas būdavo sukuriamas žemėlapio projektas – dokumentas, kuriame aprašytas žemėlapio turinys, vaizduojami objektai ir reiškiniai, pateiktas duomenų šaltinių sąrašas ir kita informacija. Atrodo, kad be įvairių ataskaitų ir patvirtinimų, šis dokumentas buvo vienintelė visur priimta kartografinių kūrinių dokumentavimo forma. Žinios, kurias sudarė darbo patirtis ir projektų statistika, buvo labai neapibrėžta duomenų kategorija. Duomenų apdorojimas praktiškai nebuvo kompiuterizuotas, skaičiavimo technika buvo apdorojama tik dalis statistinių duomenų. Skaitmeninės duomenų saugojimo formos atsiradimas ankstyvojoje stadijoje (praktiškai iki GIS sukūrimo) pats savaime dar nepakeitė egzistavusio modelio.

Galima pasakyti, kad iki masinio organizacijų kompiuterizavimo jose buvo kaupiami įvairios struktūros duomenų rinkiniai, kurie nebuvo efektyvūs dėl kelių priežasčių. Jie nebuvo aiškiai struktūrizuoti, buvo saugoma daug viena kitą dubliuojančios informacijos, kai tuo tarpu dalies logiškai susijusių duomenų galėjo trūkti. Nebuvo numatyta jokios strategijos duomenų vientisumui palaikyti, kaip ir neapibrėžta pati duomenų vientisumo sąvoka. Dalis saugomos informacijos būdavo praktiškai nepanaudojama, nes nebuvo efektyvių paieškos, o kartais net ir inventorizavimo mechanizmų arba trūko dalies susijusių duomenų. Galėjo būti saugomi prieštaringi duomenys (o SSRS dar ir naudojami sąmoningai klaidinantys iškraipyti geodezinių pagrindų masteliai), kurie galėjo būti panaudoti nepatikrinus duomenų korektiškumo, nes toks tikrinimas nebuvo numatytas. Apskritai plačiam naudotojų ratui skirta geografinė informacija tuo metu nebuvo labai vertinama, reikiamai nesirūpinama jos saugumo palaikymu, turbūt todėl, kad ji buvo arba lengvai prieinama arba visai neprieinama. Žinioms, metodikoms, vidiniams standartams, projektiniams sprendimams

apskritai nebuvo teikiamas kaupiamų duomenų statusas. Be abejo, nebuvo ir personalo, atsakingo už duomenų bazių palaikymą dabartine prasme. Tiesa, galima numanyti, kad visai kitokia situacija buvo susidariusi uždarose organizacijose (pavyzdžiui, karinėse), kurių veikloje informacijos kokybė bei panaudojimo efektyvumas visą laiką buvo kritinis veiksnys.

Statistiniai duomenys buvo saugomi archyvuose, jei numatomas poreikis juos naudoti dar kartą; naikinami, jei ateities poreikis nebuvo akivaizdus, taip taupant vietą archyvuose, kurios popieriniai duomenys užimdavo daug. Apskritai duomenys būdavo saugomi tam tikrą laiką bet kuriuo atveju, kad, esant reikalui, būtų galima įsitikinti kad jie panaudoti korektiškai. Tačiau saugojimo strategija liko formaliai neapibrėžta, nors jos sukūrimas ir buvo nagrinėjamas kaip teminės kartografijos problema. Tuo tarpu nemaža informacijos dalis buvo naudojama kopijuojant ją daug kartų (pavyzdžiui, geodezinė informacija, topografiniai duomenys kaip pagrindas įvairaus pobūdžio ir mastelio teminiams žemėlapiams). Kildavo daug problemų tokią informaciją transformuojant, pavyzdžiui, keičiant mastelius. Praktiškai tą sudėtingą darbą reikėdavo atlikti kiekvieną kartą, panaudojant tą pačią informaciją skirtingiems produktams sukurti.

Su kiekvienu nauju kartografiniu kūriniu, ypač didesnės apimties ir sudėtingesnės struktūros, be abejo būdavo įgyjama nauja organizacijos patirtis. Tai reiškia, kad formuodavosi naujos žinios, kurios paprastai nebuvo kaupiamos ir analizuojamos pagal kokį nors bendrą modelį, o tik panaudojamos intuityviai, siekiant išvengti klaidų vėlesniuose projektuose. Galima sakyti, kad žinios egzistavo daugiausia kaip organizacijų ar projektų vadovų asmeninė patirtis, organizacijos duomenų valdymo strategija, kuri sudarė pagrindą formuotis naujomis technologijomis pagrįstai strategijai ir, be abejo, įgavusi konkurencinę vertę, tapo atskirų komercinių kompanijų intelektine nuosavybe.

Neskaitmeninių sistemų ir duomenų valdymo strategijos problemas gerai atskleidžia keletas pavyzdžių.

Rengiant Lietuvos SSR atlasą (Lietuvos TSR atlasas, 1981), buvo panaudota daugybė statistikos ir grafinių duomenų iš įvairių sričių, atrenkant tik tą dalį, kuri galėjo būti tiesiogiai panaudota atlaso informacijai. Didelė dalis duomenų, kuriuos ir dabar būtų galima naudoti tolesniems darbams, yra išsaugota, bet tokiu pavidalu, kad jų naudojimas jau nebūtų efektyvus. Taigi, daug potencialios informacijos prarasta vien dėl to, kad iš anksto nebuvo suformuota duomenų rinkinio struktūra ir neapibrėžti rinkinio kaupimo principai. Dėl aplaidaus vadovų požiūrio buvo prarastos spaudinės statistinių duomenų lentelės ir dabar istorinius statistinius duomenis tenka atkurti iš diagramų žemėlapiuose. Galima teigti, kad darbai buvo orientuoti į vieną tikslą, kurį pasiekus, sukaupta informacija pasidarydavo formaliai nebereikalinga, t.y., organizacija neturėjo ilgalaikių tikslų, todėl ir negalėjo išlikti rinkos sąlygomis.

1986 metais Vilniaus universitete buvo pradėtas visai Lietuvos kartografijai svarbus darbas – žemėlapių (įskaitant seniausius), apimančių Lietuvos teritoriją ar jos dalį, inventorizavimas ir aprašymas. Teoriškai buvo užsibrėžtas tikslas pagal vieną schemą aprašyti archyvuose saugomą kartografinę medžiagą, išskiriant ir apibendrinant esminę informaciją apie juos. Tačiau pradedant aprašymus, tokia bendra schema, t.y., aprašo forma, turinys ir klasifikacija, neegzistavo, o buvo kuriama ir keičiama inventorizacijos metu. Todėl po kiekvieno struktūrinio pakeitimo darbas buvo praktiškai pradedamas iš naujo ir galų gale visai sustojo. Darbo vykdytojams išėjus iš darbo, pasirodė, kad jų darbo rezultatai negali būti efektyviai panaudoti, taigi, ir jų sukaupta patirtis organizacijai yra prarasta.

II.2 SKAITMENINIŲ DUOMENŲ BAZIŲ IR VALDYMO SISTEMŲ REIKŠMĖ KARTOGRAFIJAI

Skaitmeninių duomenų bazių valdymo sistemos (DBVS) – tai sistemos, leidžiančios saugoti struktūrizuotus duomenis, įvesti naujus bei redaguoti esamus duomenis, atlikti greitą paiešką pagal norimus kriterijus. Jos pradėtos kurti apie 1960-uosius metus. Tokių sistemų būdingos savybės yra integralumas (galimybė į vieną duomenų bazę sujungti įvairius duomenis) ir galimybė skirtingiems naudotojams nepriklausomai vienam nuo kito naudotis reikiamomis duomenų bazės dalimis. Centralizuotos organizacijos duomenų bazės ir valdymo priemonių įdiegimas, palyginus su tradiciniu saugojimo „ant popieriaus“ būdu, ne tik leido bet kuriuo metu greitai gauti išsamią ir tikslią informaciją, kompaktiškai ją saugoti, bet ir suteikė galimybę automatizuotai spręsti duomenų pertekliaus, neprieštaringumo, vientisumo, standartizavimo, saugumo problemas. Be abejo, centralizuotoms skaitmeninėms sistemoms įdiegti reikalingos didelės sąnaudos: aparatūra, personalas, palaikymo sąnaudos. Be to, organizacijos veikla tampa labai priklausoma nuo jos DBVS teisingo ir nepertraukiamo funkcionavimo. Tačiau šie DBVS trūkumai yra nereikšmingi palyginus su jų teikiama nauda.

Su DBVS susiję įvairūs teoriniai duomenų modeliai, kuriuos aptarsime kituose skyriuose, kartu su atitinkama programine įranga būdavo greitai pritaikomi įvairiems duomenims tvarkyti. Teorinių duomenų modeliavimo tyrimų rezultatas buvo sukurti metodai bei apibrėžti principai, vėliau pritaikyti ir geografinių duomenų modeliavimui kartografijoje. Būtent iš duomenų modeliavimo vėliau į kartografiją atėjo tokios esminės modeliavimo sąvokos kaip esybė, atributas, ryšys, klasė, metaklasė ir kt.

Iš duomenų bazių valdymo sistemų ir dirbtinio intelekto tyrimų išsivystė koncepcinis modeliavimas, kurį greitai perėmė taikomieji mokslai, tarp jų ir kartografija. Vėliau buvo išskirti modeliavimo lygiai: realybė – koncepcinis loginis modelis (identifikuotos esybės ir jų atributai) – ryšių loginis modelis (identifikuoti esybių ryšiai) – duomenų saugojimo modelis (konkrečios duomenų struktūros). Apibrėžti duomenų vientisumo, neprieštaringumo, išsamumo bendrieji reikalavimai suteikė galimybę įgyvendinti geografinės informacijos tolydumo principą kartografijoje.

Duomenų tvarkymo būdai, nepriklausomai nuo pačių duomenų prasmės ar naudojimo, buvo tiriami nuo pat skaitmeninių duomenų bazių atsiradimo pradžios. Iš pradžių įvairūs geografiniai duomenys tiesiog buvo saugomi paprasčiausiu skaitmeniniu pavidalu. Sukaupus pakankamai daug ir išsamių duomenų, juos pradėta struktūrizuoti. Vėlesnėse geografinių duomenų bazėse, buvo saugomos ne tik realių, bet ir įvairių statistinių arealų ribos, nors nesaugant pačių statistinių duomenų. Dabartinės geografinių duomenų bazės yra labai išsamios.

Įdiegus kompiuterines technologijas, kitaip pradėti klasifikuoti duomenys. Klasifikuoti darėsi sunku, nes buvo atrandami vis nauji duomenų šaltiniai ir pavidalai. Skirtingiems duomenims įvesti ir apdoroti buvo reikalingos skirtingos darbo vietos, daugėjo geografinių duomenų šaltinių, tokių kaip palydovinės padėties nustatymo sistemos, bei elektroninių jų kaupimo ir tvarkymo metodų. Paskutiniame 20 a. dešimtmetyje geografinių duomenų klasifikacijos problemomis domėjosi ir GIS kūrėjai. 1992 metais Dž. Langranas (Gail Langran) iškėlė laiko ir geografinių duomenų sąsajos problemą, pasiūlydamas du galimus sprendimus: „laiko sluoksnius“ (angl. *temporal layer*), skirtus iš esmės saugoti rastriniams vaizdams ir „laiko“ atributų grupę skaitmeninėse duomenų bazėse. Tokių atributų reikšmės rodo, kada objektas atsirado, išnyko, pasikeitė jo kitų atributų reikšmės ir pan. Tačiau ir iki šiol vieninga strategija, numatanti, kaip organizacijoje kaupti ir saugoti geografinius duomenis, susijusius su pokyčiais laike, nėra įprastas dalykas.

Geografinių duomenų bazių vystymosi pavyzdys.

1965 metais buvo sukurta JAV duomenų bazė *DIMECO,* kurioje skaitmeniniu pavidalu saugota kranto linija ir administracinės ribos.

1966 m. – CŽV sukurtas pasaulio duomenų bankas *World Data Bank I,* kuriame buvo saugoma pasaulio kranto linija ir valstybių ribos.

1977 m. – naujas išplėstas pasaulio duomenų bankas *World Data Bank II,* kuriame teminiai duomenys (įvairios ribos, hidrografija, transporto tinklas) jau buvo struktūrizuoti.

1982 m. – šio duomenų banko pagrindu JAV Geologijos tarnyba baigė projektą sukurti JAV nacionaliniam atlasui. Duomenys apie administracines ribas, transportą, hidrografiją, žemėnaudą, gyventojus sudarė skaitmeninį duomenų banką su apie 7 mln. įrašų. Sluoksniams buvo nurodyti reikšmingumo lygiai.

1998 m. – „Skaitmeninės Žemės“ (angl. *Digital Earth*) iniciatyva, iškelta buvusio JAV prezidento A. Goro (Al Gore) – planetos virtualus koordinuotas vaizdas, susietas su įvairiais kitais duomenų bankais, skirtas žmonėms gauti įvairią informaciją apie pasaulio gamtą ir kultūrą. Tai buvo pirmoji globalaus geografinių duomenų tinklo vizija.

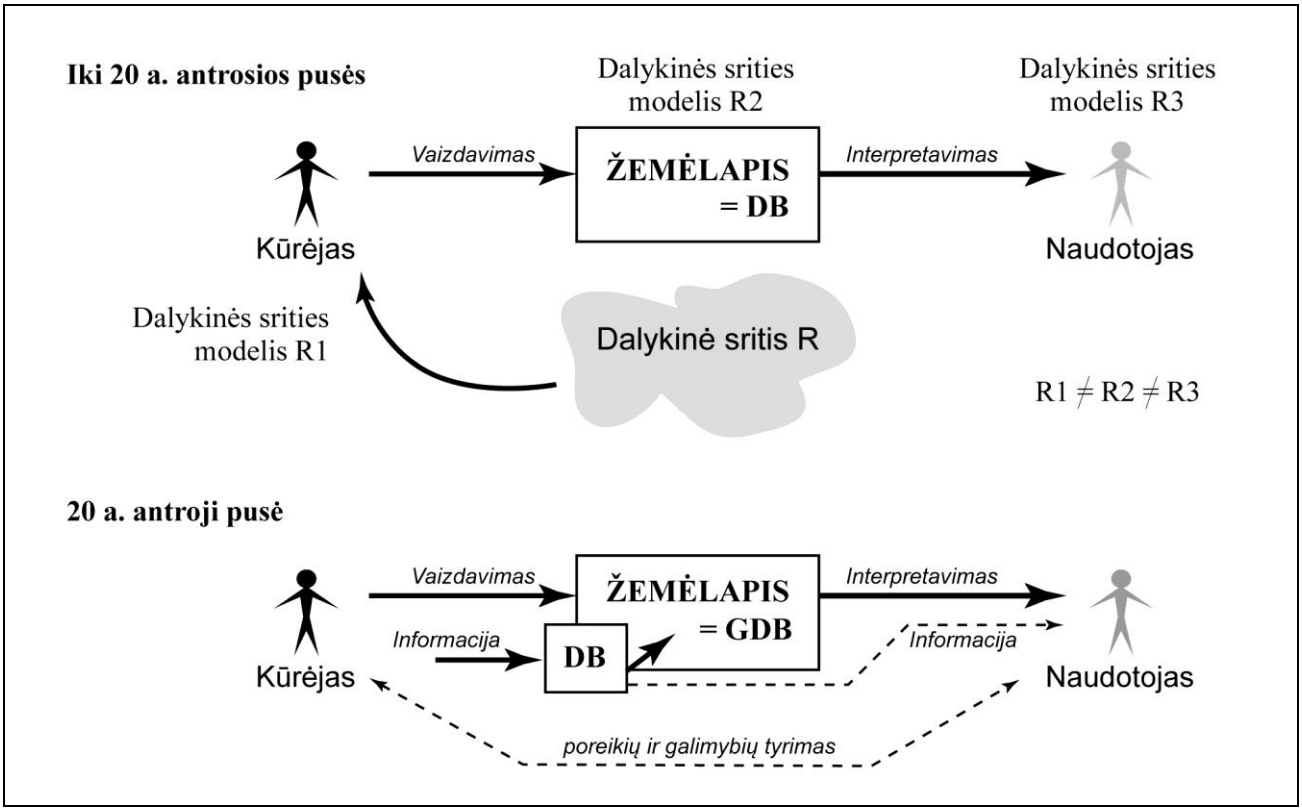
2000 m. – skaitmeninis nacionalinis JAV atlasas jau apėmė virš 100 temų ir pateikė naudotojui pagrindines duomenų analizės priemones. Atskirai egzistuoja nacionalinės 1:100 000 pagrindinio mastelio reljefo ir hidrografijos duomenų bazės, pagrįstos Digital Line Graph (DLG) modeliu.

2011 m. – „*Google* Žemė“ (angl. *Google Earth*) jau leidžia virtualiai keliauti bet kuriose žemės vietose, apžiūrint palydovo vaizdus, žemėlapius, reljefą, trimačius pastatus, apžiūrėti gausų geografinį turinį, išsaugoti aplankytas vietas ir pasidalyti jomis su kitais.

Pagrindinis skirtumas tarp paprastų skaitmeninio pavidalo duomenų ir duomenų, susietų į bet kokį modelį atitinkančią formalią duomenų bazės struktūrą yra galimybė antruoju atveju duomenis tvarkyti naudojant standartizuotas automatines priemones. Tokiu būdu iš dalies išsprendus didelių duomenų kiekių saugojimo, kopijavimo, atkūrimo ir kai kurias kitas anksčiau esminėmis buvusias problemas, pradėta galvoti apie efektyvesnį pačių duomenų panaudojimą. Tam reikėjo duomenis struktūrizuoti, tai yra, sukurti teorinį modelį, atsižvelgiant į galimus duomenų interpretavimo būdus, kurie yra susiję su jų realia prasme. Taip pat buvo svarbu, kad modelyje būtų numatyta galimybė apjungti įvairius duomenis į vieningą sistemą.

Yra žinoma keletas klasikiniais tapusių duomenų modelių, pavadintų pagal naudojamų duomenų struktūrų tipus: hierarchinis, tinklinis, reliacinis, objektinis – kurių pagrindu kuriami nauji (dedukcinis, semantinis) modeliai. Nuo maždaug 1970-ųjų metų beveik visos komercinės duomenų bazės yra pagrįstos *reliaciniu duomenų modeliu*, kuris yra virš dvidešimties metų mokslinių tyrimų rezultatas. Jo pagrindinis principas yra tas, kad duomenys naudotojui pateikiami lentelių pavidalu suteikiant galimybę atlikti įvairias operacijas ir automatiškai atnaujinti lenteles.

Įvairius teorinius duomenų modelius galima derinti ir optimaliai pritaikyti kartografinei informacijai struktūrizuoti. Nebūtinai naudotinas kuris nors „grynas“ modelis. Pavyzdžiui, GIS vektoriniai duomenys dažnai būna iš dalies sutvarkyti pagal objektinį modelį: kartu su reliacinėmis atributų lentelėmis egzistuoja vektorinių objektų klasės, tokios kaip taškiniai objektai, linijos, poligonai ir kt., nors apskritai GIS technologija paremta reliaciniu modeliu, kuris naudojamas ir ryšiams tarp geografinių objektų aprašyti.



II-1 pav. Žemėlapio funkcijos iki GIS

Vis dėlto, nors skaitmeninės technologijos ir DBVS atnešė didžiulę naudą kartografijai, palengvino ir pagreitino žemėlapių duomenų gavimą, kaupimą ir žemėlapių gamybą, jos iš esmės nepakeitė klasikinio informacijos perdavimo modelio tarp žemėlapio, jo kūrėjo ir naudotojo. Šis modelis, sėkmingai naudotas nuo tada, kai sudaryti pirmieji žemėlapiai[1], iki paskutinių 20 a. dešimtmečių, kol jo daugumoje žemėlapių taikymo sričių nepakeitė GIS modelis, aptariamas kitame skyriuje. Reikia pastebėti, kad klasikinis modelis tinka ir dabar, kai kalbama apie spausdintus žemėlapius, platinamus kaip atskiri produktai (II-1 pav.).

Pagal šį modelį žemėlapio kūrėjų ir naudotojų santykis yra paprastas: sukurtas žemėlapis naudojamas kaip geografinių duomenų saugojimo forma, teikianti naudotojams minimalias

[1] Nesutariama, net dėl apytikrės datos. Oficialiose chronologijose dažnai minimas Babilono Imago Mundi (originalas datuojamas apie 700 pr. m. e.), bet yra daug senesnių grafinių vaizdų, turinčių akivaizdžius žemėlapio požymius. Pavyzdžiui, Mežericuose, Ukrainoje aptiktas raižinys ant mamuto ilties, datuojamas 11–12 tūkst. m. pr. m. e., vaizduoja gyvenvietes palei upę. Saragosos universiteto mokslininkai, tyrinėjantys 14 tūkst. m. pr. m. e. Raižinį ant akmens, tiki, kad jame vaizduojamas reljefas, upės ir medžioklės plotai. Gali būti, kad poreikis vaizduoti informaciją erdvėje žmogui yra toks pat natūralus, kaip poreikis kalbėti.

duomenų analizės galimybes (laikotarpis iki 20 a. vidurio). Žemėlapis atlieka dvi funkcijas, susijusias su geografinių duomenų perdavimu naudotojams:

a) pateikti tinkamai suformuotą grafinį vaizdą, kurį naudotojas vertina ir analizuoja;
b) saugoti pačius duomenis – vaizduojamų objektų vietą ir kitas ypatybes, perteikiamas žemėlapio sutartiniais ženklais.

Žemėlapio kūrėjas (kartografas) analizuoja sukauptus duomenis apie pasirinktą dalykinę sritį, juos atrenka, supaprastina ir apibendrina, taip suformuodamas dalykinės srities modelį R1. Šį modelį jis vaizduoja kartografinėmis priemonėmis, kurios nėra tobulos, be to, vaizdavimui daro įtaką kartografo gebėjimai, o kartais ir noras dalį informacijos pabrėžti ar nuslėpti. Todėl žemėlapyje dalykinės srities modelis R2 nesutampa su R1. Naudotojas žiūri į žemėlapį, taip susidarydamas bendrą įspūdį apie dalykinę sritį, o norėdamas atkurti jame saugomus duomenis – jį „skaito“ ir atlieka matavimus. Duomenų atkūrimo tikslumas ir jų interpretacija priklauso ir nuo žemėlapio kokybės, ir nuo naudotojo pasirengimo, gebėjimų ir skiriamo laiko. Dalis informacijos neišvengiamai prarandama. Todėl naudotojo dalykinės srities modelis R3, suformuotas naudojantis žemėlapiu, skiriasi ir nuo R1, ir nuo R2.

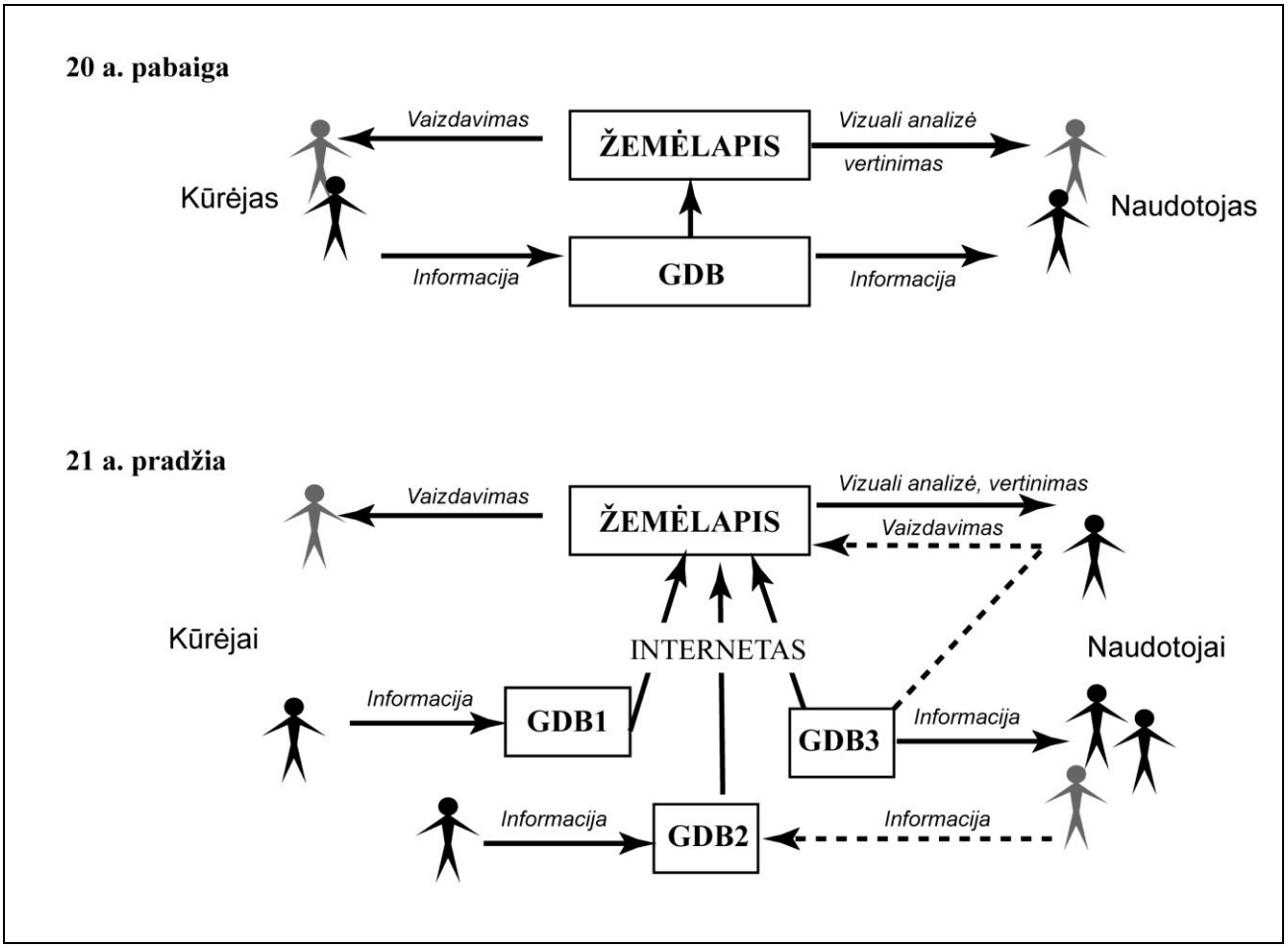
Žemėlapio, kaip geografinių duomenų saugyklos ir perdavimo priemonės, funkcija buvo labai svarbi tol, kol neatsirado geresnių tam skirtų priemonių. Skaitmeninės DBVS leido kaupti labai didelius kiekius duomenų bei juos naudoti neatliekant pakartotinių matavimų ir neprarandant pradinio duomenų tikslumo. Tokie duomenys jau 20 a. antroje pusėje buvo naudojami žemėlapiams sudaryti, o taip pat dar ir saugomi atskirose duomenų bazėse (II-1 pav. antra dalis). 20 a. pabaigoje žemėlapiai matavimams praktiškai nebenaudoti. Išryškėjus pragmatikos elementui kartografijoje, pradėta tirti žemėlapių naudotojų poreikius ir stengtis juos patenkinti. Taip buvo sukurtas, nors ir dar griežtai neapibrėžtas, grįžtamasis ryšys tarp žemėlapio kūrėjo ir jo naudotojo. Taip prasidėjo dviejų žemėlapio funkcijų atskyrimas, privedęs prie esminio žemėlapio naudojimo modelio pasikeitimo.

II.3 GIS IR JŲ ĮTAKA KARTOGRAFIJOS RAIDAI

Teorinės, technologinės ir organizacinės prielaidos geografinėms informacijos sistemoms (GIS) atsirasti pradėjo formuotis atsiradus DBVS, tačiau pagrindinis GIS vystymosi etapas buvo du paskutinieji 20 amžiaus dešimtmečiai. Tuo metu ir paplito terminas „geografinės informacijos sistema“ dažniausiai apibūdinantis įvairiams kompiuterines technologijas, skirtas darbui su geografiniais duomenimis. Šia prasme GIS naudojamas ir dabar (pavyzdžiui, išsireiškime „atviro kodo GIS“), bet vis dažniau ta pati santrumpa reiškia informacinę sistemą plačiąja prasme, t.y., geografinės informacijos sistemą, kurią sudaro ne tik techninė ir programinė įranga, bet ir žmonės, metodinis ir teisinis pagrindas, organizacinė aplinka. Vystantis GIS, informacinės sistemos apibrėžimas buvo išplėstas nurodant dar ir duomenų analizės funkcijas, bei apibrėžiant informacinės sistemos tikslus. Išskiriami trys GIS vystymosi lygmenys: sistemas skirta geografiniams duomenims inventorizuoti, vėliau – jiems analizuoti, o galiausiai – valdyti ir sprendimams priimti. Tokių sistemų pavyzdžiai yra nekilnojamo turto kadastro GIS, saugomų teritorijų GIS, savivaldybės GIS ir pan.

GIS technologijos vystėsi greitai. Dešimtajame 20-ojo amžiaus dešimtmetyje GIS naudojimas labai priklausė nuo šalies išsivystymo lygio, pavyzdžiui, Europoje 1995 metais apie 1/3 GIS rinkos teko Vokietijai, dar 1/3 – Didžiajai Britanijai ir Italijai. Šiuo metu GIS sprendimai sėkmingai kuriami ir naudojami visose šalyse. Geografinės informacijos mokslas, kaip ir kartografija, yra susijęs su įvairiais mokslais ir tam tikra prasme perima ir vienija jų idėjas. Kartografinius GIS aspektus pabrėžė įvairūs autoriai (McHarg, 1969, Berry, 1987), įvardindami GIS kaip „žemėlapių kūrimo

sistemą“, „žemėlapiu“ laikydami tam tikrą geografinių duomenų aibę (sluoksnį, temą). Nepaisant to, daugiau kaip dešimtmetį GIS sistemų kartografinės galimybės buvo itin kuklios ir jų plėtrai neskirta daug dėmesio, koncentruojantis į „naudingesnį“ duomenų valdymo ir analizės funkcionalumą. Tik 21 a. antrajame dešimtmetyje vėl prisiminta taisyklingo ir efektyvaus kartografinio vaizdavimo svarba – jis ypač aktualus kuriant skirtingais masteliais peržiūrimus Interneto žemėlapius.



II-2 pav. Žemėlapio funkcijų kaita

Dabar geografiniai duomenys naudojami praktiškai vien skaitmeniniu pavidalu, o GIS tapo pagrindiniu kartografo darbo instrumentu. Pagrindinis GIS vystymosi keliamas reikalavimas informacijai yra didelis analizei reikalingų tikslių duomenų kiekis. Todėl duomenų rinkimas ir apdorojimas tampa vienu svarbiausių ir „brangiausių“ procesų. Be to, didėjant išlaidoms duomenims gauti ir tvarkyti, svarbu, kad tuos pačius duomenis būtų galima naudoti daug kartų. Todėl kaupiami duomenys turi atitikti iš anksto apibrėžtus reikalavimus, t.y., reikia turėti duomenų naudojimo strategiją bent jau organizacijos mastu. Tai reiškia, kad organizacijoje turi būti priimtas bendras veiklos modelis bei naudojama viena bendra duomenų bazė.

Paskutiniame 20-ojo amžiaus dešimtmetyje jau buvo iš esmės pasikeitusios žemėlapio funkcijos: plačiai naudojant skaitmenines geografinių duomenų bazes ir greitai keičiantis informacijos poreikiams, žemėlapis tapo akivaizdžiai neefektyvus kaip duomenų saugojimo forma. Nors kiekviena GIS naudoja unikalų duomenų modelį, visos jos yra pagrįstos savitu bendruoju geografinių duomenų modeliu, kuriame duomenų struktūros lygmenyje atskirti geografiniai

objektai (informacija apie padėtį erdvėje) nuo jų atributinės informacijos (informacijos apie su padėtimi nesusijusias objekto savybes). Be to, duomenys visada yra atskirti nuo vaizdo, kuris gali būti automatiškai generuojamas bet kuriuo momentu. Tipiška GIS sujungė duomenų modelį su jų vaizdavimo būdu, perimdama DBVS ir grafinių projektavimo sistemų jų savybes bei principus.

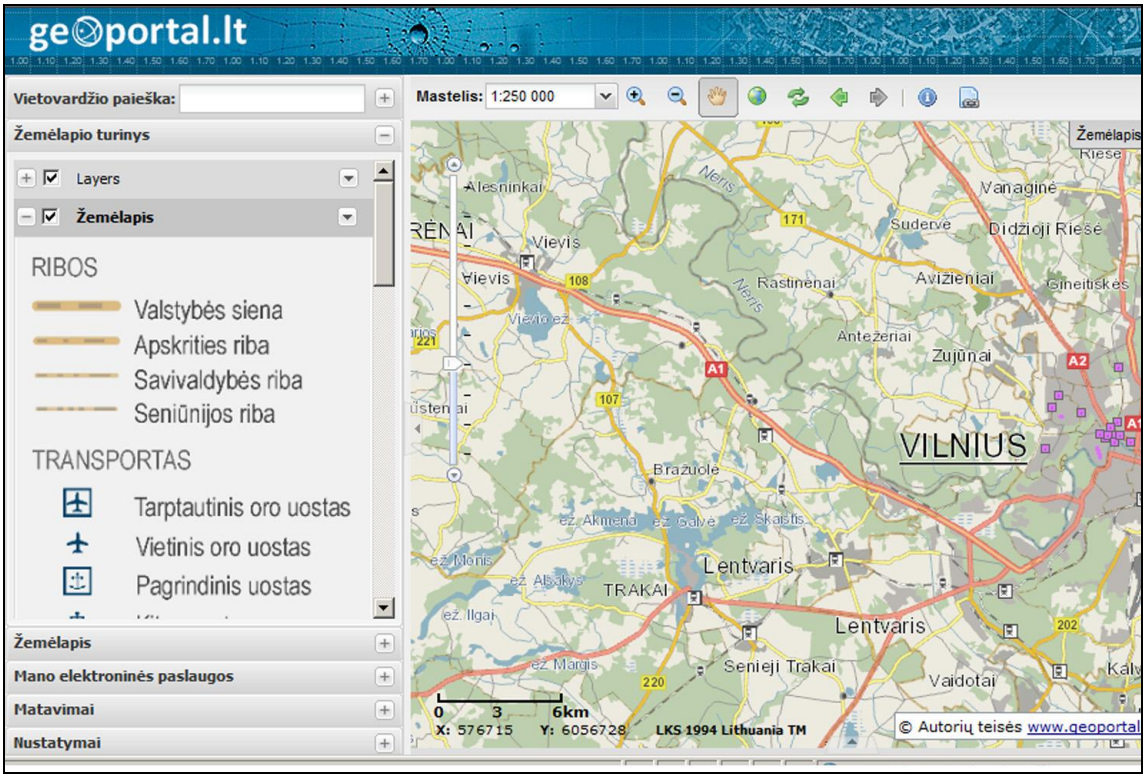
GIS duomenų valdymo technologijos reiškė esminius pokyčius **žemėlapio informacijos perdavimo modelyje.** Atsiradus efektyvioms priemonėms skaitmeniniu pavidalu kaupti ir perduoti geografinius duomenis tokios apimties ir tikslumo, kokiu jie buvo sukaupti, žemėlapis praktiškai prarado duomenų saugyklos ir perdavimo priemonės funkciją. Paveiksle (II-2 pav.) parodytas naujas modelis, kuriame žemėlapio funkcijos atskirtos: duomenų saugykla ir pagrindinis tikslios informacijos šaltinis yra geografinių duomenų bazė, o kartografo sukurto žemėlapio tikslas – tinkamai pavaizduoti duomenis pasirinktu masteliu, kuo geriau išnaudojant vizualios analizės galimybes. Žinoma, geografiniai duomenys gali būti peržiūrimi ekrane ar spausdinami, tačiau tokia duomenų vizualizacija dar nėra „tikras“ žemėlapis ir negali jo visiškai pakeisti. Taip pat spausdintame žemėlapyje galima, kaip ir anksčiau, atlikti matavimus ar iš sutartinių ženklų nustatyti objektų savybes, tačiau tai nebus taip efektyvu, kaip naudojant duomenų bazę. Tai reiškia, kad šiuolaikiniai žemėlapiai geriausiai panaudojami tada, kai jie yra teikiami naudotojams geografinių duomenų baze, panaudota jiems sukurti.

Ryšys tarp kartografinės produkcijos kūrėjų ir jos naudotojų tapo stabilus. GIS technologija suteikė naudotojams ne tik dideles žemėlapių analizės galimybes, bet ir galimybę pasirinkti ir derinti norimus geografinės informacijos sluoksnius, pasirinkti norimus matyti objektus, tai yra, formuoti patį žemėlapį iš duomenų komponentų. Dėl to riba tarp kartografo-žemėlapio sudarytojo ir žemėlapio naudotojo tapo nebe tokia aiški.

Atitinkamai keitėsi ir profesionalaus kartografo vaidmuo. Jei anksčiau jo pagrindinis tikslas buvo parengti kartografinius kūrinius, skirtus naudoti ateityje, šiuo metu darosi svarbiau pateikti geografinę informaciją taip, kad ji būtų nuolat tinkama ir pasiekiama jos naudotojams. Tai reiškia, kad reikia nuolat prižiūrėti geografinių ir kartografinių duomenų bazę, ją laiku atnaujinti; jei duomenų bazė teikiama Internetu – dar ir užtikrinti, kad ji bus naudotojui pasiekiama bet kuriuo metu. Be to, kartografas turi numatyti derinimo ir atrankos taisykles, kurios padėtų naudotojui iš duomenų pačiam susikurti norimą žemėlapio vaizdą ar jį pritaikyti savo poreikiams. Taigi, šiuolaikinis kartografas visų pirma turi būti gerai susipažinęs su duomenų valdymo principais ir technologijomis.

Be neįsivaizduojamai išaugusių geografinės informacijos apdorojimo galimybių, naujas perspektyvas teminei kartografijai dar atvėrė naujos ryšio, informacijos apsikeitimo ir efektyvios paieškos priemonės. Pasaulinis kompiuterių tinklas (Internetas) buvo pradėtas kurti 1969-aisiais metais ir per labai trumpą laiką masiškai paplito. Atsivėrusi galimybė naudotojui prieiti prie bet kurioje vietoje saugomos informacijos iškėlė naują uždavinį kartografams – kurti optimalias struktūras, sudarytas iš kartografinio vaizdo bei įvairios jį papildančios informacijos, siekiant efektyvaus šios informacijos perdavimo (komunikacijos). Programavimo įvairiomis kalbomis galimybė leidžia kurti interaktyvius, t.y., galinčius „reaguoti“ į naudotojų veiksmus žemėlapius. Naudojant įvairias šiuolaikines technologijas galima kompiuterio ekrane efektyviai pavaizduoti reiškinių dinamiką, suteikti naudotojui galimybę kurti norimą vaizdą pagal pasirinktus parametrus. Dirbtinio intelekto sistemos gali padėti optimizuoti žemėlapio generalizavimo procesą, transformacijas, sukauptas žinias integruoti į vieningą duomenų bazę.

Kaip matyti paveikslo (II-2 pav.) antrojoje dalyje, 21-ajame amžiuje paprastas „geografinių duomenų bazės – žemėlapio“ modelis išsivystė iki mums jau įprasto Interneto žemėlapio modelio, t.y., žemėlapio, kuris gali būti sudaromas dinamiškai iš skirtingų geografinių (ir ne tik geografinių) duomenų bazių. Naudotojams toks žemėlapis pasiekiamas naudojant Internetu ir peržiūrimas naudojant ne specialią GIS įrangą, o tiesiog Interneto naršyklę. Interneto žemėlapiai dažniausiai yra sąveikūs – naudodamas grafinės sąsajos priemones, naudotojas gali, pavyzdžiui, pamatuoti atstumus, plotus, sužinoti tikslias reikšmes, kurios saugomos jam tiesiogiai nepasiekiamoje geografinių duomenų bazėje. Neretai naudotojai gali keisti žemėlapio vaizdą – įjungti ir išjungti sluosnius, pakeisti sluoksnių permatomumą, spalvas ir kitas savybes, taip pritaikydami žemėlapį savo poreikiams.



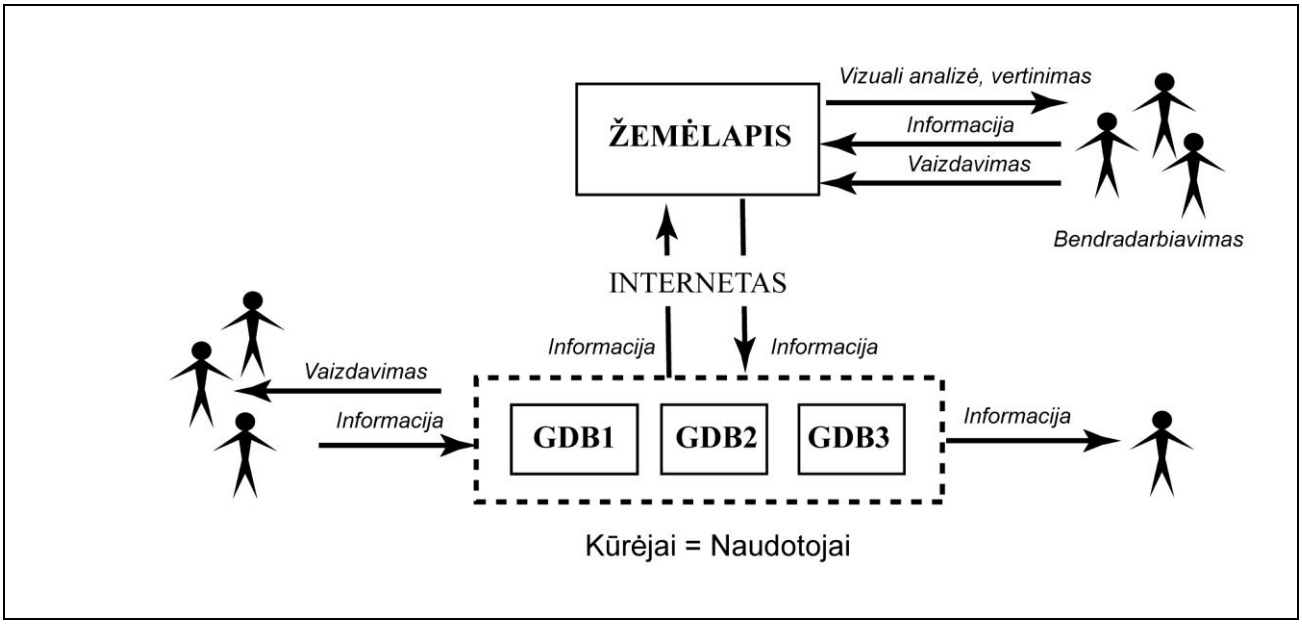
II-3 pav. GIS technologija pagrįstos žemėlapių naršyklės lango fragmentas LEI portale (www.geoportal.lt/map)

GIS technologijos yra išsivysčiusios iš kartografijos ir neabejotinai ją populiarina. Naudotojui be specialaus pasirengimo jos suteikia galimybę lengvai operuoti geografiniais duomenimis, peržiūrėti ir pakeisti pagal savo poreikius žemėlapį, įvairiai naudoti žemėlapį kaip geografinės informacijos vaizdavimo, teritorinio pasiskirstymo analizės ir modeliavimo priemonę. Būtent GIS sistemų pagrindu kuriamos kartografinių duomenų valdymo sistemos, kuriose numatyta galimybė automatiškai manipuliuoti kartografiniais duomenimis, kontroliuoti vaizdavimo priemonių korektiškumą, bei, valdyti pagrindinius žemėlapių sudarymo procesus. Iš kitos pusės, kartografijos teorija labai svarbi GIS sistemose. Ja remtis būtina kuriant taisyklingus, lengvai skaitomus ir gerai suvokiamus žemėlapius ar teikiant naudotojams priemones tokiems žemėlapiams kurti patiems iš jiems pasiekiamų geografinių duomenų, kurių vis daugėja.

II.4 GEOGRAFINIAI DUOMENYS: ATEITIES PERSPEKTYVOS

Kompiuterinėms technologijoms vystantis su dabartiniu pagreičiu, galima prognozuoti, kad ateityje dar labiau didės skaitmeninių žemėlapių naudojimas asmeninėms reikmėms. Standartizuojant skaitmenines duomenų bazes, jau kuriami tarptautiniai ir pasauliniai informacijos bankai, kurių geografinė vieta nebėra svarbi, nes jie tapo pasiekiami per pasaulinį kompiuterių tinklą. Naudotojui atsiveria galimybės bet kuriuo metu gauti norimą geografinę informaciją iš tokio banko ir pačiam kurti skaitmeninius žemėlapius. Svarbia skaitmeninio žemėlapio funkcija taps naudotojo sąveikos su informacijos bankais palaikymas, taigi jis tampa ne tik konkrečių geografinių duomenų vaizdavimo forma, bet ir **sąsajos priemone**, „protingu“ objektu, per kurį naudotojas gauna jam reikalingą informaciją. Dar daugiau, naudodami žemėlapį kaip sąsają su geografinėmis duomenų bazėmis, naudotojai gali bendardarbiauti jas kurdami ir skleisdami informaciją (taip pat žemėlapio pavidalu) kitiems naudotojams (II-4 pav.).

Niekas negali pakeisti žemėlapio, kaip pagrindinės geografinės informacijos perdavimo priemonės, funkcijos. Todėl kartografams tenka dar didesnė atsakomybė už šios priemonės kokybę ir galimybę tenkinti nuolat besikeičiančius žemėlapių naudotojų poreikius. Tai, ką anksčiau įvardindavome kaip žemėlapių ir planų sudarymą bei geodeziją, dabar vis dažniau bendrai vadinama **geografinės informacijos technologijomis ir paslaugomis**.



II-4 pav. Žemėlapis kaip sąsaja ir bendradarbiavimo priemonė

Geografinės informacijos valdymo technologija nuolat tobulėja, sudarydama sąlygas palaikyti duomenų tęstinumą erdvėje ir labai didelės apimties duomenų bazes. Jau rasti sprendimai, kaip valdyti sudėtingas transakcijas, optimaliai paskirstyti duomenų bazę didelėje teritorijoje. Bandoma vienoje duomenų bazėje palaikyti skirtingus objektų vaizdus – objektų „atstovus“ skirtingo mastelio žemėlapiuose. Ateityje galima būtų tikėtis, kad visiems pasaulio kartografams užteks vienintelės visa apimančios geografinių duomenų bazės, tačiau tokia perspektyva labai tolima dėl įvairių technologinių, teisinių ir kitų praktinių priežasčių. Vis dėlto su GIS susijusi **vieningo geografinio-kartografinio projekto** samprata: susieti tarpusavyje erdvėje registruoti informacijos sluoksniai (vertikali struktūra) ir sritys (horizontali struktūra), taip pat informacijos derinimo ir atrankos

galimybės, kai galima atskirai nagrinėti norimą teritoriją ir pasirinktą teminę informaciją (sluoksnių aibę). Tokiu būdu, kiekvienas GIS projektas potencialiai yra globalaus vieningo projekto, apimančio visą Žemės rutulį ir visas geografines sferas, dalis. Tai verčia organizacijas standartizuoti duomenis.

Vis dar svarbi ir iki galo neišspręsta geografinių duomenų bazių problema yra automatinis **generalizavimas**. Duomenų bazės, kuriose saugomi keli kartu naudojami geografinio objekto vaizdai (angl. *multiple representation databases*) gali naudoti alternatyvias geometrijas skirtinguose kontekstuose, pavyzdžiui, skirtinguose detalumo lygmenyse arba objekto „specialius egzempliorius“ skirtingiems žemėlapiams. Generalizavimo algoritmai paprastai apdoroja vieną sar keletą vieno tipo geografinių objektu vienu metu, tačiau kalbant apie geografinius duomenis ir jų vaizdavimą žemėlapyje, negalima atsieti objektų nuo jų konteksto. Pavyzdžiui, supaprastinus upę vaizduojančią liniją, atitinkamai turėtų būti pakeistos palei upę esančių teritorijų ribos, tiltų geometrinė informacija, kuri, savu ruožtu, turi atitikti kelių geometriją, ir pan. Kaip reikia elgtis su kiekvienu konkrečiu objektu, kad kartografinis rezultatas skirtingais masteliais būtų tinkamas, priklauso nuo daugelio aplinkybių. Aktyvių objektų metodas leidžia sukurti duomenų bazės objektus ne tik kaip statines duomenų struktūras, bet aktyvius agentus, kurie gali naudoti taip vadinamus *metodus* suprogramuotoms funkcijoms atlikti. Objektų metodai susiejami su tam tikrais duomenų bazės įvykiais, taip suteikiant šiems objektams sugebėjimą reaguoti į įvykius, „žinojimą, kaip elgtis“ įvairiais atvejais. Agentų metodas yra naujas daug žadantis požiūris šioje srityje. Agentais – sugebančiais siekti tikslo programiniais komponentais – pagal šią koncepciją tampa patys geografiniai objektai, sugebantys prisiderinti prie savo konteksto.

Duomenų **apimtys ir naudojimo intensyvumas** yra dar vienas iššūkis geografinių duomenų bazėms. Naujajai kartografijai labai svarbu greitai gauti reikalingas palydovines nuotraukas. Naujos kartos palydovai leidžia gauti milžiniškus kiekius duomenų, kurie saugomi dideliuose archyvuose, dažniausiai prieinamuose Internetu. Tokiuose archyvuose galima ieškoti pagal geografinę padėtį, duomenų gavimo laiką ar sensoriaus tipą. Kuriamos intelektualios palydovinės informacijos valdymo sistemos, ypač svarbios geografinei žvalgybai (angl. *Geospatial Intelligence*).

Svarbūs organizaciniai pokyčiai vyksta **geografinių duomenų infrastruktūrose**. Nuo nacionalinių kartografijos organizacijų pereinama prie erdvinių duomenų vadybininkų ir teikėjų sistemos; nuo atskirų duomenų saugyklų – prie debesų kompiuterijos (angl. *cloud computing*). Naujoje informacinėje visuomenėje organizacijos, kuriančios geografinę informaciją, taigi, ir žemėlapius, turi sugebėti valdyti pagrindinę skirtingiems tikslams naudojamą topografinę informaciją. Tai reiškia, kad jos turi kurti savo strategiją, apimančią duomenų gavimo, duomenų bazės valdymo, kokybės valdymo, duomenų teikimo ir verslo procesus. Su padėtimi susietos, GPNS naudojančios paslaugos ir mobilios technologijos reikalauja vis naujų duomenų ir geresnės jų kokybės. **Standartizavimas** atveria kelius duomenų tiekėjams ir naudotojams į pasaulinę rinką. Platesnis ir atviresnis geografinės informacijos naudojimas susijęs su **teisiniais ir etiniais** klausimais, kuriems skiriama vis daugiau dėmesio kuriant geografinių duomenų produktus ir paslaugas.

Apibendrinant galima išvardinti keletą tendencijų, susijusių su duomenų bazių kartografijoje naudojimu:

- didelės geografinės duomenų bazės, duomenų saugyklos, didesnės investicijos į duomenų tvarkymą, negu į jų kaupimą, duomenų standartizavimas;
- lengvai kuriami dinamiški, keičiamo mastelio ir realaus laiko žemėlapiai Internete ir mobiliuose įrenginiuose;
- paprastesnis, skaidresnis ir intensyvesnis geografinių duomenų naudojimas Interneto technologijų ir geografinės informacijos infrastruktūrų dėka;

- nauji produktai ir paslaugos, analizės ir sintezės įrankiai, sukuriantys pridėtinę vertę geografinių duomenų naudotojui;
- vis didėjantis bendradarbiavimas kaupiant geografinius duomenis – nuo nedidelių visuomeninių projektų iki valstybių bendradarbiavimo.

II.5 SVARBIAUSIOS LIETUVOS GEOGRAFINIŲ DUOMENŲ BAZĖS

Lietuvoje yra sukaupti pakankamai dideli kiekiai įvairių geografinių duomenų. Visų duomenų rinkinių apžvelgti beveik neįmanoma, tačiau kai kurie jų yra ypač svarbūs visai valstybei.

II.5.1 Georeferencinės duomenų bazės ir georeferencinio pagrindo kadastras.

Georeferenciniai duomenys – tai tam tikros teritorijos (valstybės ar administracinio vieneto) pagrindiniai geografiniai duomenys bazė, kuri apima geodezinio pagrindo, inžinerinių tinklų bei topografinio žemėlapio duomenis. Ji yra kaupiama ir tvarkoma pagal atsakingų valstybės ar savivaldos institucijų patvirtintas specifikacijas. Pagrindiniai georeferencinių duomenų gavimo ir tikslinimo šaltiniai yra:

a) ortofotografiniai žemėlapiai, sudaryti iš aeronuotraukų;
b) geografiniai duomenys, gaunami iš pirminių šaltinių (pavyzdžiui, matavimų vietovėje, inžinerines komunikacijas tvarkančių įmonių teikiami duomenys, kurių tikslumas dažniausiai atitinka stambesnį mastelį);
c) palyginti neseniai paplitusiu vietovės lazerinio skenavimo (LIDAR[1]) metodu gauti taškiniai duomenys.

II-5 pav. Lietuvos georeferencinių duomenų pavyzdžiai: ortofotografinio žemėlapio ORT10LT, sudaryto 2005 metų spalvotos aeronuotraukos pagrindu, fragmentas (kairėje) ir Lietuvos Respublikos M 1:10 000 georeferencinio pagrindo duomenų bazės GDB10LT (dešinėje). Šaltinis www.geoportal.lt.

[1] Plačiau apie LIDAR metodą galima paskaityti anglų kalba http://en.wikipedia.org/wiki/LIDAR.

Lietuvoje georeferencinių duomenų bazes valdo Nacionalinė žemės tarnyba prie Žemės ūkio ministerijos (www.nzt.lt). Ši įstaiga organizuoja, koordinuoja ir kontroliuoja georeferencinių duomenų tvarkymo, tikslinimo ir atnaujinimo darbus, rengia ir priima reikalingus teisės aktus, užtikrina tinkamą duomenų bazių funkcionavimą ir naudojimą. Lietuvos goreferencinių duomenų bazes tvarko ir platina valstybės įmonė Distancinių tyrimų ir geoinformatikos centras „GIS-Centras“ (www.gis-centras.lt). Šios įmonės specialistai tikslina, atnaujina, apdoroja, sistemina duomenis, užtikrina, kad jie nebūtų įrašyti klaidingi ir neišsamūs, užtikrina techninės ir programinės įrangos veikimą, techninę priežiūrą ir tobulinimą, surinktų duomenų konfidencialumą, apsaugą nuo nesankcionuoto naudojimo, pakeitimo ar sunaikinimo, teikia duomenis naudotojams.

Lietuvoje 1:10000 mastelio georeferencinio pagrindo duomenų bazė, apimanti visą šalies teritoriją, buvo sukurta 2001–2003 metais ortofotografinių žemėlapių (ORT10LT) pagrindu. Tokie ortofotografiniai žemėlapiai, taigi, ir georeferencinių duomenų bazė pagal Lietuvos teisės aktus turi būti atnaujinami ne rečiau kaip kas penkeri metai.

Geografinių duomenų rinkiniai mūsų šalyje ilgą laiką buvo sudaromi ir atnaujinami naudojant skirtingus duomenų šaltinius ir nesuderintus tarpusavyje procesus. Buvo kuriami atskiri panašios paskirties to paties mastelio geografinių duomenų rinkiniai, tokie kaip kartografinių duomenų bazė KDB10LT ir GDB10LT bei keleto didžiųjų miestų vektorinės duomenų bazės VDB10. Pagrindinio 1:10000 mastelio duomenys kartu su kitais šaltiniais buvo naudojami išvestiniams smulkesnio mastelio georeferencinio pagrindo produktams kurti. Iš jų verta paminėti pirmąją 1994–1996 m. sudarytą Lietuvos kosminio vaizdo žemėlapio M 1:50000 skaitmeninių duomenų bazę LTDBK50000 (http://www.gis-centras.lt/gisweb/index.php?pageid=213). Ši skaitmeninė duomenų bazė apima visą Lietuvos teritoriją. Ją sudaro vektorinių duomenų rinkinys (LTDBK50000-V), kur pateikiama visos Lietuvos reljefo, hidrografijos, žemėnaudos, infrastruktūros bei kita topografinė informacija, ir spalvoto rastro duomenų rinkinys (LTDBK50000-SR). Pateikiami žemėnaudų, vandens tėkmių, kelių ir geležinkelių, infrastruktūros, reljefo, administracinių ir saugomų teritorijų ribų duomenys. Lietuvos kosminio vaizdo žemėlapio M 1:50000 skaitmeninių duomenų bazę sudaro 135 nomenklatūriniai lapai.

2011 m. platintos šios vektorinės georeferencinės duomenų bazės:

- GDB5LT (sudaryta 2006–2007 m., mastelis 1:5000),
- GDB10LT (sudaryta 2008 m., mastelis 1:10000),
- GDB50LT (sudaryta 2008 m., mastelis 1:50000),
- GDB250LT (sudaryta 2007 m., mastelis 1:250000).

Lietuvos Respublikos georeferencinis pagrindas GDB10LT – georeferencinių duomenų bazės sudedamoji dalis, kurią sudaro visos Lietuvos teritorijos geodezinio pagrindo ir topografinių duomenų bazių svarbiausių objektų duomenų rinkinys. GDB10LT apima visą Lietuvos teritoriją – 2782 lapus. GDB10LT sudaro atskiri geografinių duomenų sluoksniai: Lietuvos teritorijos suskirstymas lapais, geodezinio pagrindo punktai, vietovardžiai, valstybės siena, upės, upeliai ir kanalai, keliai, geležinkeliai, hidrografijos plotiniai objektai. GDB10LT gali būti naudojamas kadastrų ir registrų tvarkymui, savivaldybių bendriesiems ir specialiesiems planams rengti, transporto organizavimo, logistikos, teisėsaugos, karybos, saugos ir krizių prevencijos uždaviniams spręsti.

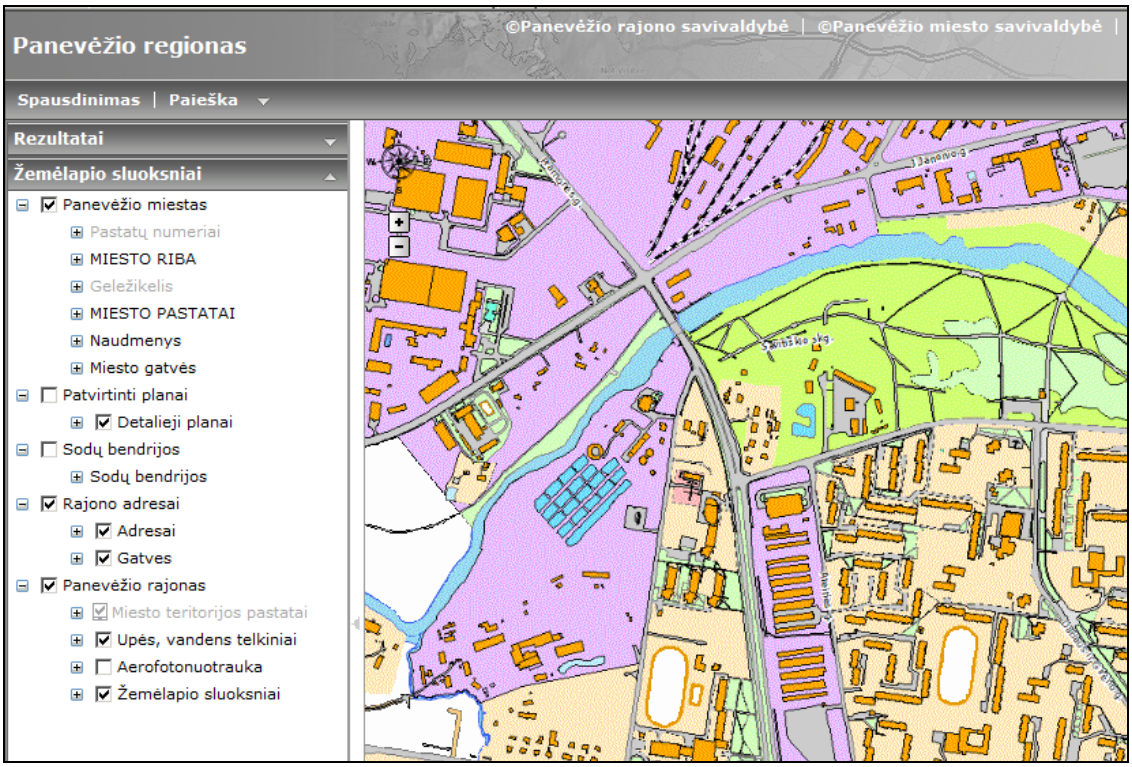
2010-05-11 priimtame Lietuvos Respublikos Geodezijos ir kartografijos įstatymo pakeitime įteisintas **Georeferencinio pagrindo kadastras**, kurio paskirtis – registruoti svarbiausius geografinius objektus, rinkti, kaupti, apdoroti, sisteminti, saugoti ir teikti fiziniams ir juridiniams

asmenims kadastro duomenis ir dokumentus, atlikti kitus kadastro duomenų tvarkymo veiksmus. Georeferencinio pagrindo kadastro duomenys teikiami naudotojams neatlygintinai, išskyrus atvejus, kai šie duomenys naudojami pridėtinės vertės paslaugoms ar produktams sukurti (tuo atveju atlyginimo už georeferencinio pagrindo kadastro duomenų teikimą dydį nustato Vyriausybė). Vadovaujančioji georeferencinio pagrindo kadastro įstaiga yra Nacionalinė žemės tarnyba, o kadastro tvarkymo įstaiga – VĮ „GIS-Centras“.

Įstatymas nustato, kad kadastras turi apimti šiuos objektus:

1) kelių ašinės linijos;
2) gatvių ašinės linijos;
3) geležinkelių ašinės linijos;
4) upių, upelių ašinės linijos;
5) kanalų ir melioracijos griovių ašinės linijos;
6) ežerų ir tvenkinių ribos;
7) pastatų ribos;
8) miškų naudmenų ribos;
9) žemės ūkio naudmenų ribos;
10) geodezinio pagrindo punktai;
11) žemės paviršiaus (aukščio) taškai.

2010 m. pradėti rengti Georeferencinio pagrindo kadastro nuostatai ir kiti kadastrui funkcionuoti būtini poįstatyminiai teisės aktai. Tai pirmas žingsnis link vieninga metodika paremto valstybės georeferencinių duomenų tvarkymo.



II-6 pav. Panevėžio rajono savivaldybės georeferenciniai duomenys Interneto žemėlapyje (fragmentas, šaltinis: Panevėžio rajono savivaldybė, http://www.panrs.lt)

Savivaldybių duomenų bazėse saugoma daug georeferencinių duomenų, kurie būtini teritorijų planavimui, projektavimo darbams, išteklių apskaitai, aplinkos vertinimui, modeliavimui, analizei ar kontrolės tikslams (II-1 lentelė). Tokia informacija savivaldybės reikmėms kaupiama stambiu, t.y., 1:5000 ir stambesniu masteliu, todėl galėtų būti puikiu pradinių duomenų šaltiniu valstybės georeferenciniam pagrindui. Savivaldybės, tvarkančios savivaldybių erdvinių duomenų rinkinius, yra įstatymo įpareigotos teikti juos Georeferencinio pagrindo kadastrui atnaujinti. Deja, daugiau ar mažiau išsamius ir tvarkingus geografinius duomenis kaupia tik nedaugelis savivaldybių (pavyzdžiui, Vilniaus, Klaipėdos, Šiaulių miestų, Panevėžio, Utenos, Alytaus rajonų savivaldybės, kurių geografiniai duomenys naudotojams teikiami Internete). Kitose savivaldybėse, dėl įvairių priežasčių (gerai nesuvokto poreikio, finansinių išteklių ar gebėjimų stokos) duomenys arba nėra kaupiami, arba jie kaupiami chaotiškai, nestandartizuoti, todėl sudėtinga juos panaudoti savivaldybės uždaviniams spręsti ar valstybės duomenų rinkiniams atnaujinti.

II-1 lentelė. Savivaldybių pagrindinių geografinių duomenų bazių turinys

GDB tipas	**GDB informacija**
Topografinė	Reljefas, hidrografija, augalija, transporto infrastruktūra, urbanistiniai objektai.
Inžinerinių komunikacijų	Elektros perdavimo komunikacijos, vandentiekio komunikacijos, dujotiekio komunikacijos, naftotiekio komunikacijos šilumotiekio komunikacijos, nuotekų komunikacijos, kitos komunikacijos.
Adresų	Gatvių pavadinimai, pastatų numeriai ir kodai.
Administracinio suskirstymo	Administracinės ribos, seniūnijos, planuojami rajonai.
Želdinių	Duomenys apie želdynus, jų užimamą plotą, rūšį.
Detaliųjų planų	Naujai suformuoti sklypai, jų ribos, reglamentas, plėtros ribos, servitutai, gatvių ribos.
Techninių projektų	Statybų leidimus gavę projektuojami pastatai, gatvės, inžinerinės komunikacijos.

II.5.2 Registrai ir kadastrai

Valstybės **registras** – tai oficiali valstybės duomenų bazė ir visuma susijusių teisinių, organizacinių, technologinių priemonių, skirta registruoti įstatymų nustatytus objektus, rinkti, kaupti, apdoroti, sisteminti, saugoti bei teikti fiziniams ir juridiniams asmenims registruojamų objektų kiekybinius, kokybinius ir kitus duomenis bei dokumentus. Registras, kuriame kaupiama geografiniai duomenys, vadinamas **kadastru**. Registrų kūrimą Lietuvoje reglamentuoja Valstybės registrų įstatymas[1]. Registrų duomenys ir jų informacinės sistemos nuosavybės teise priklauso valstybei.

Valstybės registrų sąrašas skelbiamas Informacinės visuomenės plėtros komiteto prie Susisiekimo ministerijos tinklapyje adresu http://www.ivpk.lt/registrai/. Kai kurių registrų fiziškai dar nėra, jie kuriami, bet yra priimti jų teisės aktai. Eksploatuojamų registrų

Įgyvendinant 2003 m. lapkričio 17 d. Europos Parlamento ir Tarybos direktyvą 2003/98/EB „Dėl viešojo sektoriaus informacijos pakartotinio naudojimo“ siekiama efektyvios visų registrų sąsajos, kurioje gali dalyvauti ir kitos duomenų bazės (pavyzdžiui, georeferencinių duomenų bazė, naudojama susieti kurio nors registro objektams su žemės paviršiaus objektais). Sąsajoms ypač svarbūs **pagrindiniai registrai**, kurių duomenys naudojami susijusiuose registruose.

[1] http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=238881

Pagrindiniai valstybės registrai yra šeši:

1) juridinius asmenis registruojantis registras;
2) gyventojus registruojantis registras;
3) nekilnojamąjį turtą ir teises į jį registruojantis registras;
4) objektų, kurių geografinė padėtis nesikeičia, adresus registruojantys registrai;
5) teisės aktus registruojantys registrai;
6) daiktų ir turtinių teisių įkeitimą registruojantis registras.

Trys iš jų yra glaudžiai susiję su geografiniais duomenimis – adresų geografine vieta, gyventojų registracijos adresais, taigi, ir geografine vieta bei žemės sklypų padėtimi. Specifinis registras yra ankstesniame skyrelyje aprašytas Georeferencinio pagrindo kadastras, kurio duomenys priklauso nuo mastelio ir atitinka realius žemės paviršiaus objektus (kurių ribos nebūtinai sutampa su juridinėmis ribomis). Todėl šis kadastras, nors nėra pagrindinių registrų sąraše, labai svarbus teisingai geografinei sąsajai sukurti tarp II-2 lentelėje išvardintų registrų.

II-2 lentelė. Svarbiausi registrai, siejami su geografine informacija[1]

Eil. Nr.	**Registro pavadinimas**	**Registro tvarkytojas**
1.	Adresų registras	Valstybės įmonė Registrų centras
2.	Juridinių asmenų registras	Valstybės įmonė Registrų centras
3.	Kultūros vertybių registras	Kultūros paveldo departamentas prie Kultūros ministerijos
4.	Lietuvos Respublikos gyventojų registras	Gyventojų registro tarnyba prie Lietuvos Respublikos Vidaus reikalų ministerijos
5.	Lietuvos Respublikos miškų valstybės kadastras	Valstybinė miškotvarkos tarnyba
6.	Lietuvos Respublikos saugomų teritorijų valstybės kadastras	Valstybinė saugomų teritorijų tarnyba prie Aplinkos ministerijos
7.	Nekilnojamojo turto kadastras[2]	Valstybės įmonė Registrų centras
8.	Nusikalstamų veikų žinybinis registras	Finansinių nusikaltimų tyrimo tarnyba prie Vidaus reikalų ministerijos Informatikos ir ryšių departamentas prie Lietuvos Respublikos vidaus reikalų ministerijos Policijos departamentas prie Lietuvos Respublikos vidaus reikalų ministerijos Priešgaisrinės apsaugos ir gelbėjimo departamentas prie Vidaus reikalų ministerijos Valstybės sienos apsaugos tarnyba prie Lietuvos Respublikos vidaus reikalų ministerijos
9.	Švietimo ir mokslo institucijų registras	Švietimo informacinių technologijų centras
10.	Ūkininkų ūkių registras	Valstybės įmonė Žemės ūkio informacijos ir kaimo verslo centras
11.	Žemės gelmių registras	Lietuvos geologijos tarnyba prie Aplinkos ministerijos

[1] 2011 m. spalio mėn. duomenys, šaltinis: Informacinės visuomenės plėtros komitetas, http://www.ivpk.lt)
[2] 2011 m. gruodį dar neeksploatuojamas, duomenų bazė ir IS kuriama.

12.	Lietuvos Respublikos upių, ežerų ir tvenkinių valstybės kadastras[1]	Lietuvos Respublikos aplinkos ministerija
13.	Lietuvos Respublikos teritorijų planavimo dokumentų registras[1]	Lietuvos Respublikos aplinkos ministerija
14.	Lietuvos Respublikos geležinkelių infrastruktūros registras[1]	Lietuvos Respublikos susisiekimo ministerija
15.	Valstybinės reikšmės ir pavojingų objektų registras[1]	Priešgaisrinės apsaugos ir gelbėjimo departamentas prie Vidaus reikalų ministerijos

Adresų registro duomenų bazėje saugoma informacija apie objektų unikalius adresus. Ji pildoma ir tikslinama nuolat, remiantis kitų pagrindinių valstybės registrų duomenimis bei dokumentais, kuriais suteikiami gatvių pavadinimai, suteikiami, patikslinami ar panaikinami adresai, bei kitais dokumentais. Adresų suteikimo objektams ir jų registravimo tikslas ir esmė – užtikrinti adreso unikalumą, kurį sąlygoja gatvės pavadinimo unikalumas gyvenamojoje vietovėje, pastato numerio unikalumas gatvėje, patalpos unikalumas pastate, adreso unikalus ir nekintantis kodas bei adreso vietos (koordinačių) unikalumas. Taip galima turėti tikslią informaciją, būtiną fizinių asmenų gyvenamajai vietai, juridinių asmenų buveinėms deklaruoti, nekilnojamajam turtui registruoti, planuoti logistikai, operatyvinių ir avarinių tarnybų darbui, teikti pašto, kurjerių paslaugoms. Adresas siejamas su vieninteliu tašku Žemės paviršiuje, todėl galima kartografiškai pavaizduoti bet kokią informaciją, kurios dalis yra adresas. Tokia informacija sugoma Juridinių asmenų, Kultūros vertybių, Gyventojų, Nusikalstamų veikų, Švietimo ir mokslo institucijų bei kituose registruose.

Nekilnojamojo turto registre ir kadastre kaupiami duomenys apie nekilnojamojo turto objektus. Integruoti, išsamūs ir teisinį statusą turintys nekilnojamojo turto kadastro ir registro duomenys yra būtini vykdant nekilnojamojo turto restituciją, valstybinio turto privatizavimą, naujas statybas bei esamų statinių rekonstrukciją. Būtina efektyvaus nekilnojamojo turto administravimo sąlyga – tiksli geografinė informacija apie nekilnojamojo turto objekto geografinę padėtį, konfigūraciją, plotą ar ilgį. Tai kadastro vietovių ir blokų ribos, žemės sklypų ribos, pastatų centro taškai ir kontūrai, inžinerinių statinių ašinės linijos ir kontūrai, nekilnojamojo turto verčių zonos, administracinių vienetų ir gyvenamųjų vietovių ribos, gatvių ašinės linijos, adresų taškai. Kadastro GIS sistemoje kaip kartografinis pagrindas naudojamas Georeferencinio pagrindo kadastras, skaitmeninis ortofotografinis žemėlapis ir skaitmeninės kartografinės duomenų bazės.

Lietuvos Respublikos upių, ežerų ir tvenkinių valstybės kadastre, Saugomų teritorijų, Miškų kadastruose, Žemės gelmių ir panašiuose registruose kaupiama geografinė informacija, kuri turi būti susieta su Georeferencinio pagrindo ir Nekilnojamo turto kadastru ne per adresus, o naudojant geografinių objektų (pavyzdžiui, saugomų teritorijų ar miško žemės ribų, gręžinių vietų) padėties informaciją. Tokią sąsają įgyvendinti labai sudėtinga, nes skiriasi registrų geografinių duomenų tikslumas, generalizavimo metodas, be to, objektai kinta laike.

II.5.3 Statistinių duomenų bazės

Statistiniai duomenys valstybėje kaupiami **ataskaitų** (atskaitomybės statistika) ir specialiai **organizuotų statistinių stebėjimų** pagrindu. Pirmuoju atveju duomenis atsakingoms institucijoms ataskaitų pavidalu teikia įmonės, įstaigos, organizacijos ir ūkiai, remdamiesi ūkinės veiklos apskaitos dokumentais. Ataskaitos rengiamos pagal iš anksto patvirtintas formas bei teikimo terminus, kuriuos Lietuvoje tvirtina Statistikos departamentas. Už ataskaitų teikimą laiku ir jų

teisingumą atsako įmonių, įstaigų, organizacijų ir ūkių vadovybė. Ataskaitos gali būti teikiamos metais ar trumpesniais laikotarpiais. Atskaitomybė gali būti valstybinė ir žinybinė. Valstybinė atskaitomybė privaloma visoms žinyboms bei naudojama šalies valdymo reikmėms. Žinybinė atskaitomybė vedama ministerijose, žinybose ir naudojama jų uždaviniams spręsti.

Oficialioji statistika – tai valstybės ir savivaldos institucijų valstybės reikmėms skirtų statistinių duomenų apie ekonominius, demografinius procesus, socialinius veiksnius ir visuomeninius bei aplinkos pokyčius rinkimo, tvarkymo ir statistinės informacijos skelbimo pagal Oficialiosios statistikos darbų programą sistema, kurios pagrindiniai uždaviniai yra:

a) nustatyti vienodą statistinės informacijos apie ekonominius, demografinius procesus, socialinius veiksnius ir visuomeninius bei aplinkos pokyčius rinkimo ir tvarkymo sistemą, vadovaujantis nacionalinėmis reikmėmis ir tarptautinių organizacijų taikoma metodologija;
b) apdoroti, apibendrinti ir analizuoti statistinius duomenis, rengti statistinę informaciją;
c) skelbti parengtą informaciją ir užtikrinti, kad ji būtų pasiekiama visuomenei;
d) teikti statistinę informaciją valstybės valdžios ir valdymo bei vietos savivaldos institucijoms, tarptautinėms organizacijoms.

Kitą valstybės statistiką Lietuvos Respublikoje gali tvarkyti politinės partijos, politinės ir visuomeninės organizacijos, profesinės sąjungos, religinės bendruomenės bei privačios statistikos įstaigos[1].

Oficialiosios statistikos organizavimo bendruosius principus, fizinių ir juridinių asmenų teises ir pareigas teikiant duomenis statistikos reikmėms, valstybės ir savivaldybių institucijų bei įstaigų teises ir pareigas tvarkant bei naudojant statistinius duomenis reglamentuoja Lietuvos Respublikos statistikos įstatymas[2]. Oficialiosios statistikos duomenų šaltiniai yra:

a) fizinių asmenų, juridinių asmenų bei juridinio asmens teisių neturinčių įmonių apskaitos duomenys, statistiniai stebėjimai ir surašymai;
b) mokesčių, muitinių, švietimo, sveikatos, darbo biržos, socialinės apsaugos ir komunalinio ūkio, Lietuvos banko informacinės sistemos, valstybės registrai, taip pat savivaldybių institucijų ir įstaigų sukaupti administracinių tvarkomųjų bei asmens dokumentų duomenys.

Oficialius statistinius duomenis tvarko Lietuvos statistikos departamentas (http://www.stat.gov.lt), jo teritorinės įstaigos; ministerijos, kitos valstybės institucijos, kurių veiklą reglamentuojančiuose teisės aktuose numatyta statistikos tvarkymo funkcija, Lietuvos bankas. Lietuvos statistikos departamentas yra Vyriausybės įstaiga kuri dalyvauja formuojant valstybės politiką statistikos valdymo srityje ir ją įgyvendinant. Lietuvos statistikos departamentas renka, apdoroja, analizuoja ir skelbia oficialiąją statistiką apie šalies ekonominius, socialinius, demografinius ir fizinės aplinkos pokyčius, rengia regioninius ir administracinius teritorinius statistinius rodiklius, koordinuoja ministerijų ir kitų institucijų veiklą oficialiosios statistikos srityje. Statistinė informacija Lietuvoje yra prieinama visiems. Statistikos departamento Interneto svetainėje ji teikiama nemokamai.

Statistikos informacinės sistemos duomenų bazėse yra kaupiama ši informacija.

- Statistiniai rodikliai, 2011 m. apimantys 10 temų: gyventojai, socialinė statistika, žemės ūkis, medžioklė, miškininkystė ir žuvininkystė, verslas, makroekonomika, surašymai, regioninė statistika.

[1] Plačiau apie Lietuvos statistikos sistemą ir institucijų teikiamus duomenis – http://www.stat.gov.lt/lt/pages/view/?id=1159
[2] http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=371719

- Informacija apie institucinius sektorius ir subsektorius, pagal kuriuos klasifikuojami ūkio subjektai.
- Statistiniai klasifikatoriai. Tai 29 tarptautiniai ir nacionaliniai klasifikatoriai.

Rodiklių duomenų bazėje dauguma informacijos pateikiama pagal savivaldybes[1]. Regioninė statistika rengiama ir seniūnijų lygmeniu.

Klausimai diskusijai

Kaip įsivaizduojate kartografijos, informatikos ir geografinės informacijos mokslo santykį dabar ir ateityje? Kaip šios disciplinos susijusios su geografija?

Užduotys savarankiškam darbui

Internete surinkite informaciją apie Lietuvoje esančius valstybinės reikšmės duomenų rinkinius, apimančius nurodytą ar pasirinktą temą (pavyzdžiui, pastatus, saugomas teritorijas, gyventojų informaciją) ir ją susisteminkite. Įvertinkite, kaip duomenų rinkinių įvairovė, patikimumas ir apimtis atitinka temos svarbą ir apimtį.

Išnagrinėkite Lietuvos Respublikos Geodezijos ir kartografijos įstatymo bei Lietuvos Respublikos Valstybės informacinių išteklių valdymo įstatymo nuostatas, kurios liečia georeferencinius duomenis. Pasirenkite jas pakomentuoti.

Užduotys praktikos darbams

Susipažinkite su reliacinių duomenų bazių pavyzdžiais. Pasižiūrėkite, kaip lentelėse saugoma informacija apie įvairius objektus. Turimoje studentų duomenų bazėje panaudokite įvairius *MS Access* duomenų bazės komponentus: lentelę, užklausą, formą, ataskaitą, makro komandą, modulį.

Sukurkite lenteles „Dėstytojai“, “Paskaitos” ir „Egzaminai“, kuriose būtų saugoma informacija apie lankytas paskaitas, laikytus egzaminus ir dėstytojus. Nustatykite visų laukų tipus ir apribojimus. Įveskite duomenis, įsitikinkite, kad sukurtos struktūros yra tinkamos ir patogios naudoti.

Sukurkite lentelėms paprastas jų peržiūros formas, naudodamiesi vedliu (angl. *wizard*).

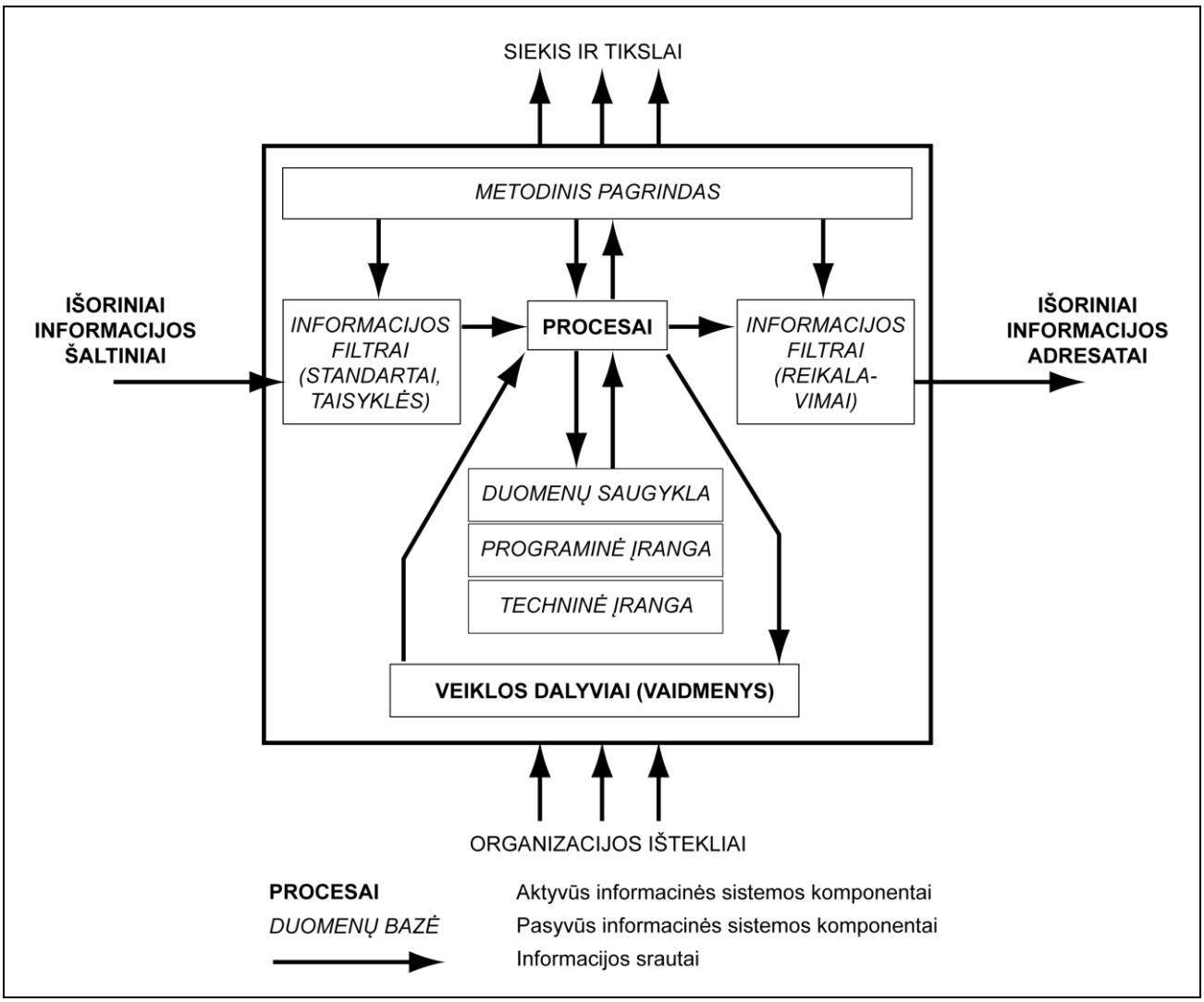
[1] Kol nebuvo panaikintos apskritys – ir pagal apskritis.

III. DUOMENYS ORGANIZACIJOS VEIKLOJE

Dauguma organizacijų įsivaizduoja, kad gali sukaupti duomenis, kurių kaina būtų maža, o kokybė puiki. Tačiau iš tiesų jos paprastai turi brangiai kainavusius blogos kokybės duomenis.
Evan Miller

III.1 DUOMENŲ BAZĖ ORGANIZACIJOS INFORMACINĖJE SISTEMOJE

Informacinė sistema (IS) informatikoje apibrėžiama kaip duomenų apdorojimo sistemos ir organizacijos išteklių (duomenų, žmonių, techninių priemonių, finansų ir pan.) visuma, skirta informacijai apdoroti, kurti ir skleisti (siųsti ir gauti), siekiant apibrėžtų ir įgyvendinamų tikslų. Kitaip tariant, tai struktūrizuotas priemonių, procesų ir procedūrų rinkinys, kuriame yra kaupiami duomenys, tvarkomi ir perduodami naudotojui.



III-1 pav. Informacinės sistemos bendroji struktūra

Valstybės informacinė sistema – valstybės institucijai teisės aktų nustatytoms funkcijoms, išskyrus vidaus administravimą, atlikti reikiamos informacijos apdorojimo procesus (duomenų ir dokumentų tvarkymo, skaičiavimo, bendravimo nuotoliniu būdu ir t. t.) vykdanti sistema, kuri veikia informacinių technologijų pagrindu. Valstybės informacinių sistemų kūrimą Lietuvoje reglamentuoja valstybės informacinių sistemų kūrimo metodika ir reikalavimai valstybės informacinių sistemų specifikacijoms, patvirtinti Informacinės visuomenės plėtros komiteto prie Susisiekimo ministerijos.[1] Minėti reikalavimai nėra privalomi ne valstybės institucijoms, jie taip pat netaikomi IS, kuriose tvarkomi duomenys yra valstybės ar tarnybos paslaptis.

Informacinė sistema nagrinėjama šiais aspektais:

- motyvacijos (kokie tikslai, vertinimo kriterijai, kas, kam, kaip ir kiek naudinga);
- informaciniu (kokia informacija renkama ir kokia formuojama);
- algoritminiu (kas su informacija daroma, kas ir kur formuoja, laiko ir apdoroja informaciją);
- naudotojo (kas ir kokiuose padaliniuose formuoja, laiko, apdoroja ir naudojasi informacija savo veikloje);
- struktūros (kas sudaro IS: techninė įranga, programinė įranga, duomenų bazė, susijusių valstybės ir žinybinių registrų duomenų bazės, tinklas, procedūros, darbuotojai, tikslai, socialinis kontekstas);
- laiko (kada turi būti gaunama, apdorojama ir perduodama vartotojams IS informacija).

IS modeliai gali būti įvairių tipų (valdymo, sprendimų priėmimo, ir kitokios IS) ir skirtingo detalumo. Paveiksle (III-1 pav.) pateikta bendra IS schema, kurioje pabrėžiama proceso (veiklos), kaip pagrindinio siejančio IS komponento, svarba. Duomenų saugyklą naudoja įvairūs procesai, kurie kuria jos turinį, naudoja duomenis bei juos transformuoja. Jokia informacinė sistema nėra uždara. Į ją patenka duomenys iš skirtingų išorės šaltinių. Išoriniai duomenys turi tenkinti nustatytus reikalavimus, susijusius su IS paskirtimi ir tikslais. Išorinės informacijos filtrai reikalingi tam, kad neatitinkantys reikalavimų duomenys nepatektų į IS. Analogiškai, IS sukuriami duomenys ar kiti produktai, skirti išoriniams adresatams, taip pat turi tenkinti apibrėžtus reikalavimus. Reikalavimų, taisyklių, principų, žinių visuma sudaro IS funkcionavimo metodinį pagrindą. Juo vadovaujantis planuojami ir vykdomi procesai. Procesų sąsaja su IS naudotojais yra sukuriama per vaidmenis, tokiu būdu tiesiogiai nesiejant procesų su konkrečiais darbuotojais.

Duomenų saugykla, kurios dalis yra organizacijos duomenų bazė, yra labai svarbi bet kokios informacinės sistemos dalis, bet ji turi būti tinkamai susieta su kitais informacinės sistemos komponentais. Techniškai duomenų saugykla yra visuma technologijų, naudojamų informacijai saugoti – duomenų laikmenos, diskų sistemos su visomis jų funkcijomis ir tinklai, jungiantys saugyklų infrastruktūrą.

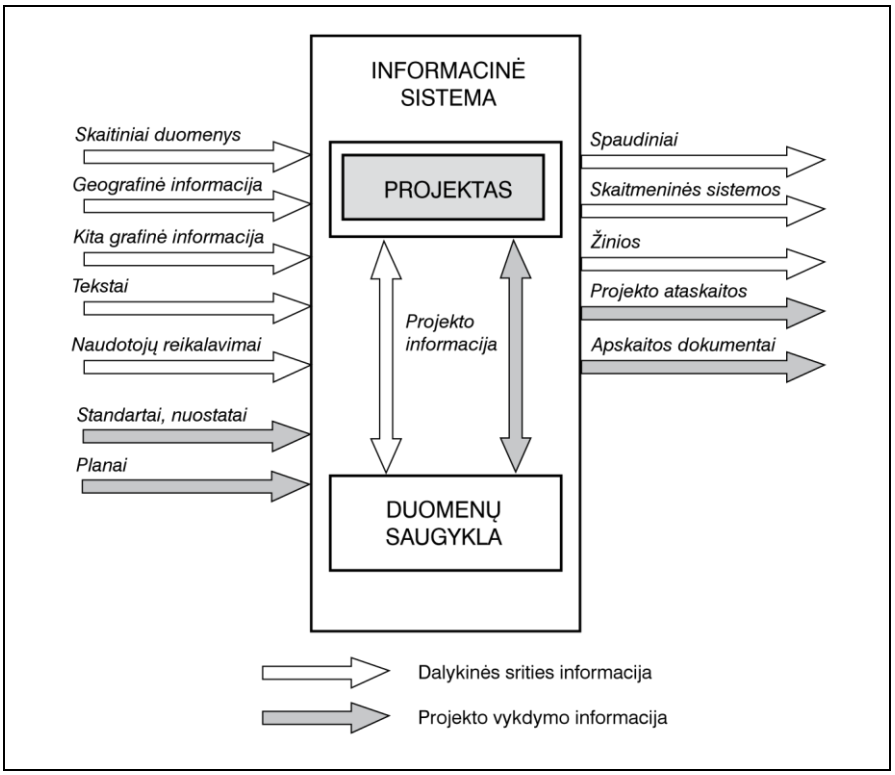
IS duomenys – tai ne tik projektų dalykinių sričių ir organizacijos apskaitos duomenys, bet ir projektų informacija – tai informacija, tiesiogiai nenaudojama produktams (pavyzdžiui, žemėlapiui, atlasui ar geografinių duomenų rinkiniui) sukurti, tačiau svarbi visos informacinės sistemos egzistavimui. Projekto informaciją sudaro daugiausiai įvairūs dokumentai, kuriuose registruojamos projekto vykdymo metu sukauptos žinios: sutartys, planai, darbų grafikai, naudotojų apklausų duomenys, specifikuoti reikalavimai, informacinės sistemos veikimo statistika, ataskaitos, naudoti reglamentuojantys dokumentai, duomenų interpretavimo taisyklės, metodikos ir pan..

[1] Įsakymo tekstas http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=243580&p_query=&p_tr2=

III.2 DUOMENŲ SRAUTŲ MODELIAVIMAS

Duomenų srautas – tai esybių, jų ryšių, atributų ar kitokios (nebūtinai formaliai aprašytos) informacijos rinkinys[1], perduodamas arba tarp veiklos procesų, arba tarp proceso ir duomenų saugyklos. Kiekvienam IS procesui baigus darbą, sukuriami duomenys, kurie arba patenka į informacinės sistemos duomenų saugyklą ir ten saugomi, kol jų vėl prireikia, arba perduodami kitam procesui, kuris gali juos sunaikinti, transformuoti arba perduoti toliau. Duomenų srautai gali būti skirstomi į pagrindinius, kuriuos veiklos procesai naudoja tiesiogiai informacinės sistemos tiklams pasiekti ir sistemos palaikymo srautus (tokie, pavyzdžiui, yra duomenų srautai, skirti sukurti atsarginėms duomenų kopijoms, saugoti ir esant poreikiui atkurti arba naikinti duomenis, patikrinti duomenų vientisumui). Palaikymo srautų nenagrinėsime, nes jie dažniausiai nėra susiję su dalykinės srities specifika.

Duomenų srautų modeliai sudaromi ir informacinėms sistemoms, ir atskiriems projektams tam, kad būtų gerai įsivaizduojama, kokia informacija patenka į sistemą iš skirtingų šaltinių, o kokia teikiama IS ar projekto adresatams, kaip sistemos veiklos rezultatas. Valstybės IS privaloma atskirai aprašyti išorinius ir vidinius informacijos srautus.



III-2 pav. Bendriausias duomenų srautų modelis

Aprašant kiekvieną IS išorinį ar vidinį informacijos srautą, nurodoma: šaltinis (siuntėjas), adresatas (gavėjas), srauto vardas (identifikatorius), srauto tipas (dokumentų srautas, duomenų srautas ir kt.), srauto perdavimo būdas (paštu, kabeliais, šviesolaidžiais, radijo bangomis, kompiuterinėmis ryšio priemonėmis ir kt.), srauto apimtis (per pasirinktą laiko vienetą), srauto perdavimo greitis, srauto perdavimo periodiškumas arba sąlygos, kurioms susidarius perduodamas srautas, duomenų

[1] Nepainioti su nestruktūrizuotu duomenų srautu (angl.: *traffic*)

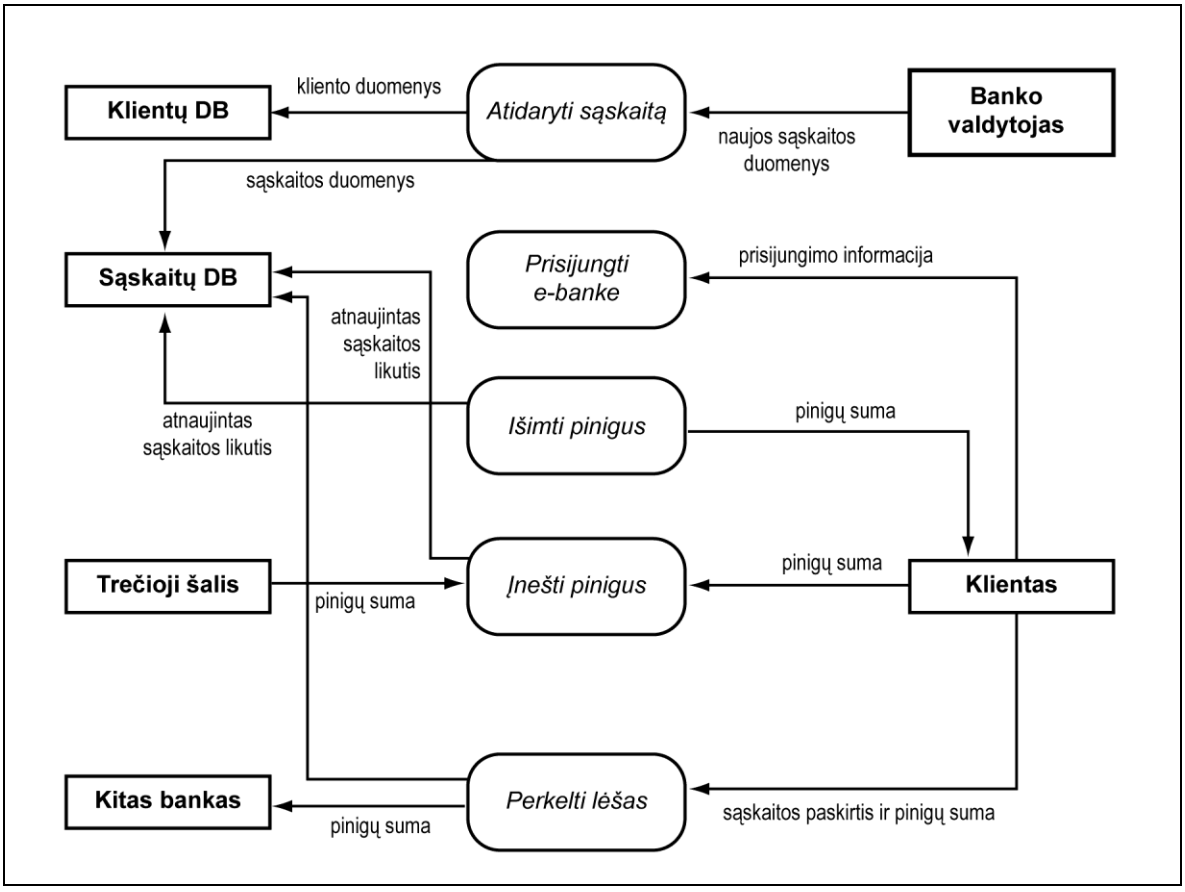
pavidalas (popieriniai dokumentai, skaitmeniniai duomenys ir kt.), intensyvumas (dokumentai, simboliai ir kt. per laiko vienetą). IS srautus rekomenduojama pateikti grafinių schemų pavidalu.

Apibūdinant išorinius informacijos srautus, aprašomi IS ryšiai su kitų objektų IS, duomenų bazėmis, registrais ir kitais su organizacija susietais objektais. Apibūdinant IS vidinius informacijos srautus, aprašoma informacijos judėjimo tvarka tarp IS komponentų. Jei informaciniame modelyje procesai vienareikšmiškai susieti su vykdytojais, duomenų srautai gali būti rodomi ir tarp vykdytojų. Rekomenduojama visų pirma aprašyti informacijos srautus tarp objekto padalinių, o po to – srautus objekto padalinių viduje.

Informaciniai modeliai, siejantys duomenų srautus ir organizacijos strukūrines dalis bei jų vykdomus procesus, sudaromi trimis abstrakcijos lygmenimis.

1. Modelis, atspindintis organizacijos ir išorinių esybių ryšius ir duomenų srautus.
2. Modelis, atspindintis ryšius ir duomenų srautus tarp organizacijos struktūrinių vienetų (skyrių).
3. Detalus modelis, vaizduojantis ryšius ir duomenų srautus skyrių viduje.

Taigi, duomenų srautų modeliai gali būti labai bendri, vaizduojantys tik išorinius srautus, arba didesnio detalumo lygmens – apimantys šaltinius, vidinius duomenų srautus ir jų siejamus IS komponentus. Keičiant modelio detalumo lygmenį duomenų srautai atitinkamai skaidomi ar jungiami pagal klasifikavimo taisykles, aprašytas XII.1 skyriuje.



III-3 pav. Detalus duomenų srautų modelis

Sudarant bendrą duomenų srautų modelį duomenų saugyklos fizinis pavidalas nėra itin svarbus – ją gali sudaryti duomenų bazės, failai, dokumentų segtuvai ar kitos laikmenos. Nedetalizuojant

paveiksle (III-2 pav.) parodyta, kokie gali būti duomenų srautai, ateinantys į organizaciją (pavyzdžiui, vykančią teminio kartografavimo projektus) ir kokie srautai, generuojami informacinės sistemos gyvavimo metu, patenka už jos ribų. Į informacinę sistemą patenkantys dokumentai ar spaudiniai turi būti inventorizuojami, kita informacija įskaitmeninama ir išsaugoma, kad vėliau būtų galima ją naudoti numatytiems rezultatams gauti. Detalaus duomenų srautų modelio pavyzdys pateiktas III-3 pav.

Reikalavimai, apribojantys galimus duomenų srautus modeliuose:

1. Kiekvienas žemiausio lygio procesas turi turėti bent vieną įeinantį duomenų srautą.
2. Kiekvienas žemiausio lygio procesas turi turėti bent vieną išeinantį duomenų srautą.
3. Duomenys, apibrėžti išeinančiu duomenų srautu, turi būti gauti transformuojant duomenis iš įeinančių srautų.
4. Visi įeinančių srautų duomenys turi būti būtini išeinančiu srautų duomenims gauti.
5. Kiekvienas išeinantis duomenų srautas turi susidėti iš duomenų, kurie naudojami kitų procesų, arba patenka į sistemos išore (ja laikoma ir informacinės sistemos duomenų saugykla).
6. Duomenų saugykla turi turėti bent vieną įeinantį srautą (tai reiškia, kad turi būti srautas, sukuriantis jos turinį).
7. Duomenų saugykla turi turėti bent vieną išeinantį srautą (tai reiškia, kad jos turinys yra naudojamas).
8. Sudėtinių procesų ar objektų įeinantys ir išeinantys duomenų srautai turi būti jų komponentų atitinkamai įeinančių ir išeinančių srautų sumos. Šis reikalavimas taip pat reiškia, kad modeliuoti turi būti pradedama nuo žemiausio lygmens komponentų.

III.3 DUOMENŲ SAUGOJIMO TRUKMĖ

Kad organizacijos duomenų bazė būtų optimaliai panaudota, labai svarbu ne tik kaupti visus reikiamus duomenis, bet ir laiku juos atnaujinti, be to, saugoti kuo mažiau perteklinės, nereikalingos informacijos. Todėl nuolat tenka spręsti, kurie sukurti duomenų objektai turi tolesnę vertę ir turi būti išsaugoti po to, kai baigiasi projektas, pavyzdžiui, išleistas atlaso tiražas; kokiu būdu jie turi būti išsaugoti informacinėje sistemoje; kiek laiko saugomi ir kas kiek laiko atnaujinami. III-1 lentelėje pateikta duomenų rinkinių klasifikacija pagal jų saugojimo laiką. Iš jos matyti, kad neužtenka kiekvienu konkrečiu atveju nuspręsti, kiek laiko bus saugomi duomenys, bet reikia numatyti ir galimo kiekvieno tipo duomenų atnaujinimo strategiją.

Nuolatinė informacija yra saugoma be pakeitimų visą duomenų rinkinio egzistavimo laiką. Tai yra nekeičiami, bet naudojami daug kartų įvairiems tikslams duomenys – standartizuoti, istoriniai, baigtų produktų ir projektų duomenys. Informacijos apimtis bet kokioje organizacijoje nuolat didėja, o ji saugoma vis ilgiau, kartais dėl to, kad nėra numatyti metodai išrinkti ir naikinti nereikalingą informaciją, tačiau dažniausiai todėl, kad jos gali prireikti. Kartais saugoti informaciją ilgą, ar net neribotą laiką reikalauja galiojantys teisės aktai, įvairūs kiti reikalavimai. Sukaupus duomenų daugiau negu leidžia techninės galimybės, reikia juos archyvuoti.

Archyviniai duomenys yra nuolatinių duomenų rūšis. Tai duomenys, kurių gali prireikti ateityje ir jie iš principo svarbūs organizacijai, bet yra sudėtingi, užima daug vietos ir praktiškai naudojami retai. Jiems priskiriami ir tokie objektai, kaip gatavi žemėlapiai, kuriuos sudėtinga sukurti iš turimų geografinių duomenų dėl kurios nors priežasties (pavyzdžiui, nestandartiniai kartografuojami objektai). Archyvuose saugomi ir neskaitmeniniai objektai. Skaitmeninis archyvavimas gali būti

nagrinėjamas įvairiais aspektais – nuo struktūrizuoto archyvų specialisto požiūrio iki duomenų saugyklos specialisto techninių vertinimų, tačiau visais atvejais jo tikslas yra nustatytu laikotarpiu saugoti, apsaugoti ir padaryti prieinamą pasirinktą informaciją, t.y.,

- užtikrinti, jog saugoma informacija nepakis laikui bėgant,
- naudotojai galės perskaityti ir suprasti archyvuotą informaciją,
- naudotojai galės gauti duomenis greitai ir patogiai, ieškoti jų įprastais metodais.

Archyvavimas leidžia sumažinti duomenų saugojimo išlaidas, geriau tvarkyti kaupiamą informaciją. Tam, kad archyvavimas būtų efektyvus, reikia specifinių techninių ir programinių sprendimų.

Atnaujinama informacija gali būti dviejų rūšių:

a) Neversijuojama (duomenys pasensta arba tampa neteisingi pasikeitus poreikiams, todėl buvusi šių duomenų versija vertės nebeturi ir yra tiesiog pašalinama iš duomenų bazės).
b) Versijuojama (išlieka reikalingos ir yra saugomos skirtingos duomenų versijos skirtingiems laiko momentams; versijos numeris ir data yra būtini tokio duomenų rinkinio atributas). Duomenų versijos paprastai kuriamos reguliariais laiko intervalais. Akivaizdu, kad duomenų rinkinio skirtingoms versijoms saugoti reikalinga papildoma vieta ir turi būti numatytos versijų tvarkymo procedūros.

Duomenų rinkiniai gali būti atnaujinami reguliariai apibrėžtais laiko intervalais, arba kiekvieną kartą esant poreikiui.

Reguliariai atnaujinami duomenys yra labai įvairūs, kaip ir jų atnaujinimo intervalai. Tai grupei priklauso ir ortofotografiniai vaizdai, atnaujinami kas penkeri metai, ir meteorologinė informacija, atnaujinama kas kelios valandos. Svarbu tai, kad vieno duomenų rinkinio atnaujinimo periodas yra visada tas pats ir iš anksto žinomas.

Pagal poreikį atnaujinami duomenys gali būti pakeisti bet kuriuo metu. Jie paprastai yra vienos iš dviejų rūšių.

1. Duomenų rinkiniai, kuriuose pakeitimai daromi retai, dažniausiai taisant jų klaidas, kartais papildant. Retai kada saugomos jų senos versijos. Tokie yra, pavyzdžiui, hidrografinio tinklo, geomorfologinių objektų ar kitų retai besikeičiančių realių objektų duomenys.
2. Nuolat tvarkomi ir keičiami duomenų rinkiniai, kuriuos naudotojas turi gauti aktualius bet kuriuo metu. Tokio rinkinio geras pavyzdys yra duomenys, Internetu naudotojams teikiami elektroninės paslaugos pavidalu, pavyzdžiui, pajamų deklaravimo duomenys, žemės sklypų ribos. Šie duomenų rinkiniai atnaujinami kiekvieną kartą gavus naują informaciją apie gautas pajamas ar registruotą sklypą. Tvarkant oficialius (juridinius) duomenis dažnai svarbu neprarasti informacijos apie visus anksčiau įvykusius pakeitimus, t.y., tenka saugoti jau ne tik viso duomenų rinkinio, bet kiekvieno duomenų bazės objekto versijas. Tai reikalauja sudėtingo objektų gyvavimo ciklo sekimo ir pokyčių registravimo procedūrų duomenų bazėje.Kai kurie nuolat keičiami duomenys atnaujinami labai dažnai (pavyzdžiui eismo duomenys – kas kelios minutės).

Istoriniai duomenys, dažnai naudojami žemėlapiams sudaryti, gali būti saugomi kaip nuolatiniai arba kaip skirtingų datuotų versijų rinkinys, jei sena informacija nėra nekorektiška ir turi vertę. Sprendimą, kokius duomenis saugoti šiuo būdu, kokiais laiko intervalais atnaujinti, kada laikyti pasenusiais, reikia priimti kiekvienu konkrečiu atveju, dažnai tą gali padaryti tik atitinkamos srities specialistai. Tačiau už atnaujinimo procesą yra atsakingi organizacijos informacinės sistemos tvarkytojai.

III-1 lentelė. Duomenų rūšys pagal jų saugojimo laiką

Duomenų rūšis	**Pavyzdžiai**
Nuolat saugomi	
Paprasti nuolatiniai	Iliustracijos, standartiniai sutartiniai ženklai, istorinių administracinių ribų duomenys
Archyviniai	Dokumentų archyvas, spaudiniai
Atnaujinami	
Reguliariai atnaujinami neversijuojami	Kelių tinklo duomenys, meteorologiniai žemėlapiai
Atnaujinami esant poreikiui, neversijuojami	Reljefo, teritorijų planavimo duomenys
Reguliariai atnaujinami versijuojami	Ortofotografiniai vaizdai, įvairūs statistiniai, pavyzdžiui, gyventojų surašymų, duomenys, meteorologinių stebėjimų duomenys
Atnaujinami esant poreikiui, versijuojami	Nekilnojamo turto kadastro, adresų registro duomenys
Laikinai saugomi	
Tarpiniai skaičiavimų rezultatai	Grafikai, diagramos, buferinės zonos
Lengvai gaunami iš kitų šaltinių	Žemėlapiams sudaryti naudojami statistinių duomenų rinkiniai
Tiesiogiai nesusiję su organizacijos tikslais	Konkrečiam projektui įvykdyti panaudoti autoriaus pateikti duomenys
Apskaičiuoti	
Sudėtingai gauti	Skaitmeninis reljefo modelis, generalizuota geografinė informacija
Saugomi ilgiau už jų šaltinius	Pagal užsakymą naudojant įvairius duomenis parengti teritorijų planavimo sprendiniai

Laikini duomenys – tai duomenys, kurie naudojami konkretiems tikslams pasiekti ir planuotai saugomi kur kas trumpiau, negu informacinės sistemos gyvavimo ciklas. Tokie duomenys yra dviejų rūšių:

a) tarpiniai įvairių skaičiavimų rezultatai;
b) duomenys, kuriuos geriau apsimoka prireikus gauti iš kitų šaltinių, negu saugoti organizacios duomenų bazėje;
c) tiesiogiai nesusiję su organizacijos tikslais, specifiški kuriam nors trumpalaikiam tikslui ar vykdomam projektui, dažnai pateikiami užsakovo. Tai gali būti ir duomenys, kurių licencija (teisė naudoti) organizacijai suteikiama tik ribotam laikui.

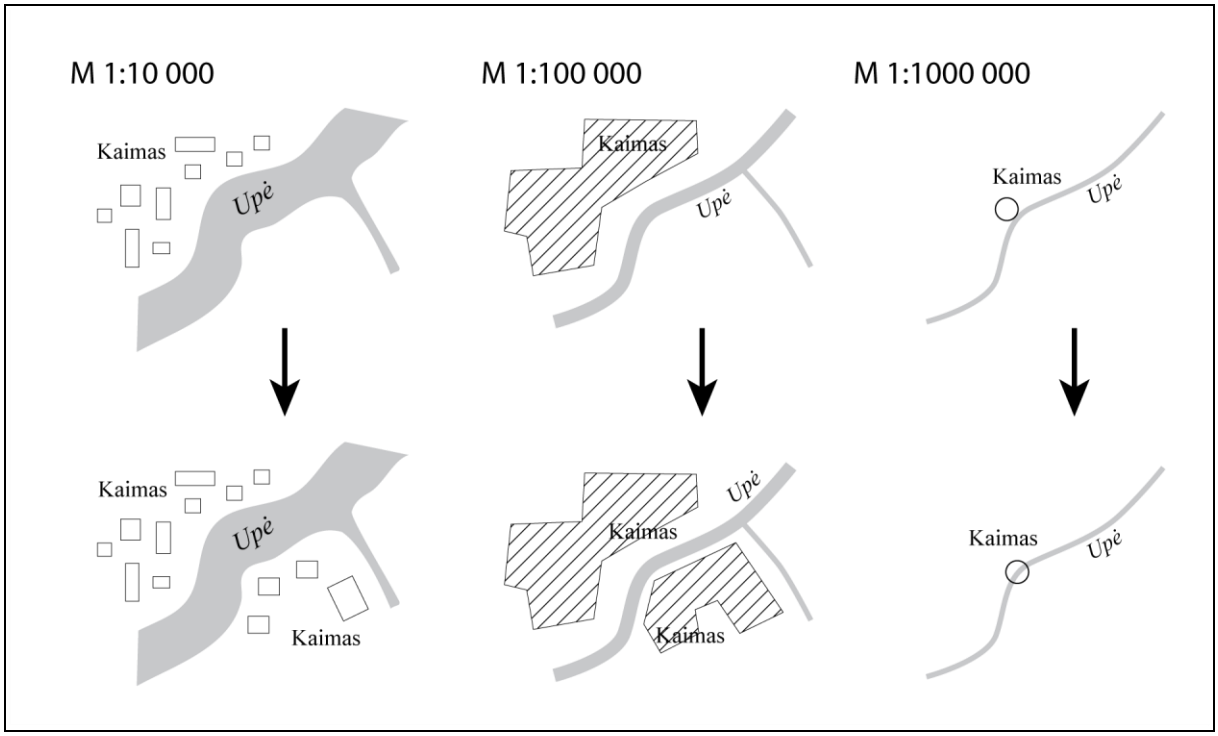
Apskaičiuoti (išvestiniai) duomenys paprastai yra sukuriami iš duomenų bazėje saugomų pagrindinių duomenų esant poreikiui įvykdžius užklausas, užprogramuotas užduotis, panaudojant žemėlapio projektą ir kitais būdais. Projektuojant duomenų bazę svarbu atskirti, kurie duomenys yra apskaičiuojami, o kurie pagrindiniai. Pavyzdžiui, žmonių gimimo datos yra pagrindiniai, nekintantys ir nuolat saugomi duomenys, o žmonių amžius – lengvai apskaičiuojami duomenys, todėl amžiaus duomenų saugoti nėra prasmės. Vis dėlto, yra du atvejai, kai duomenys, kurie gali būti apskaičiuoti, išsaugomi duomenų bazėje:

a) kai skaičiavimai trunka ilgai, rezultatas naudojamas dažnai, o jo pagrindiniai duomenys atnaujinami retai;

b) kai numatoma apskaičiuotus duomenis saugoti ilgiau, negu kurį nors jų apskaičiavimui reikalingą komponentą (pavyzdžiui, tuo atveju, kai skaičiavimams panaudojami laikini duomenys).

Apskritai sprendimas, kokiais atvejais reikia saugoti apskaičiuojamus duomenis, priklauso nuo apskaičiuojamų duomenų naudojimo dažnumo ir resursų sąnaudų, reikalingų jiems sukurti iš turimų duomenų. Apskaičiuoti duomenys gali būti saugomi kaip nuolatiniai arba laikini, bet dažniau jie yra atnaujinami vienu iš aukščiau aprašytų metodų.

Tas pats geografinės informacijos sluoksnis (pavyzdžiui, reljefas, hidrografija, gyvenvietės) gali būti dažnai naudojamas kuriant įvairius žemėlapius. Problema šiuo atveju yra ta, kad, naudojant skirtingo pagrindinio mastelio sluoksnius (nors peržiūrint skaitmeninių žemėlapių mastelis lengvai keičiamas), kartografijos standartai reikalauja skirtingo generalizavimo lygmens (III-4 pav.).



III-4 pav. Geografinės informacijos atnaujinimo skirtingais masteliais problema

Priklausomai nuo naudojamos programinės įrangos, kuri nulemia generalizavimo galimybę ir jo sudėtingumą, galima pasirinkti vieną iš geografinės informacijos sluoksnio saugojimo variantų:

a) saugoti vieną sluoksnį, kurio generalizavimo lygmenį nulemia šaltinio mastelis (jis turėtų būti pakankamai stambus) ir kiekvieną kartą prireikus, jį generalizuoti. Šis metodas tinka, kai geografinė informacija yra tokios struktūros, kuri leidžia ją generalizuoti greitai ir vienareikšmiškai, pavyzdžiui, gyvenvietės, atrenkamos pagal gyventojų skaičių ir/arba administracinę reikšmę;
b) saugoti keletą to paties sluoksnio skirtingo (bet standartinio ir iš anksto numatyto) mastelio variantų. Tai geras būdas, kai naudojama programinė įranga neleidžia generalizuoti automatiškai, arba kai generalizavimo procesas yra ilgas ir sudėtingas, o pagrindiniai duomenys nėra dažnai atnaujinami;
c) saugoti keletą to paties sluoksnio skirtingo mastelio variantų bei naudoti automatizuotas objektų atnaujinimo bei generalizavimo procedūras tarp skirtingų mastelių. Šis metodas

yra pats sudėtingiausias, bet jis vienintelis tinka tada, kai ir pirminiai, ir generalizuoti duomenys yra dažnai keičiami ir naudojami. Pavyzdys gali būti mišraus detalumo georeferencinių duomenų bazė, nuolat papildoma savivaldybių stambaus mastelio duomenimis bei jos pagrindu sukurta smulkesnio mastelio duomenų bazė, naudojama teminiams žemėlapiams sudaryti.

Su atnaujinimais susijusi pagrindinė problema geografinėse duomenų bazėse ir kartografijoje yra ta, kad pakeitus duomenis, reikia tuo pačiu metu juos atnaujinti visuose išvestiniuose produktuose. Todėl idealiu atveju, kai numatoma duomenis atnaujinti, turi būti aprašytas atitinkamas procesas, kuris leistų lengvai surinkti nuorodas į objektus, kuriems sukurti buvo panaudoti keičiami duomenys ir pakeisti juos automatiškai. Tokio tipo duomenys yra, pavyzdžiui, geografinių pavadinimų sąrašai. Pasikeitus vieno pavadinimo rašybai, reikia ją pakeisti visose išvestinėse duomenų bazėse ir žemėlapiuose, kur tas vietovardis buvo panaudotas. Vis dėlto, pavadinimą ar kita paprastą atributo reikšmę rasti ir pakeisti yra daug paprasčiau, negu aptikti objekto geometrijos pasikeitimus ir pakeisti visus susijusius objektus. Tai tikrai sudėtinga problema, nes sekti absoliučiai visų susijusių pakeitimų praktiškai neįmanoma, ypač turint omenyje kad tas pats geografinis objektas skirtingose duomenų bazėse gali būti interpretuojamas ir vaizduojamas skirtingai. Pavyzdžiui, stambaus mastelio duomenų rinkinyje upė saugoma kaip plotinis objektas, ribojamas kairiojo ir dešiniojo kranto, smulkesniu masteliu ta pati upė vaizduojama vienodo pločio plotiniu objektu, kurio krantai atkartoja upės vidurio linijos geometriją, o dar smulkesniu masteliu upė vaizduojama tik kaip jos vidurio linija. Teorinis pakeitimų sekimo ir automatinio atnaujinimo sprendimas geografinėms duomenų bazėms yra paremtas unikalių kiekvieno geografinio objekto identifikatorių visose duomenų bazėse sistema. Taip pat turi būti griežtai apibrėžtos sąsajos tarp skirtingų duomenų bazių bei tarp stambesniu ir smulkesniu masteliu vaizduojamų objektų. Tada atnaujinus objekto duomenis vienu (prasminga – stambiausiu) masteliu, galima atrasti visus jo vaizdus kitose duomenų bazėse ir automatiškai generalizuoti.

III.4 DUOMENŲ TIRIAMOJI ANALIZĖ IR DUOMENŲ GAVYBA

Duomenų tiriamoji analizė (angl.: *exploratory data analysis*) yra specifinis duomenų analizės metodas, kurio tikslas – stebint didelius kiekius duomenis formuluoti naujas hipotezes. Jis naudojamas papildant statistinės analizės hipotezių tikrinimo (angl. *confirmatory data analysis*) metodą. Duomenų tiriamosios analizės metodai plačiai taikomi **duomenų gavybos** (angl. *data mining*) disciplinoje, kuri nagrinėja, kaip dideliuose kiekiuose dažniausiai sudėtingos struktūros duomenų pastebėti iš anksto nežinomas ar tarp daug faktų sunkiai įžvelgiamas struktūras, t.y., kurti naują informaciją. Tiriami organizacijų ar duomenų saugyklų duomenys, siekiant aptikti paslėptus dėsningumus bei sąryšius.

Nors kaip disciplina duomenų tiriamoji analizė susiformavo 20 a. paskutiniaisiais dešimtmečiais, jos pradžia galima laikyti 20 a. vidurį, kai kompiuteriai pirmą kartą buvo panaudoti didelių eksperimentinių duomenų bazių analizei naudojant statistinius ir dirbtinio intelekto metodus. Keletas veiksnių skatina greitą duomenų tyrimo įrankių kūrimą ir diegimą šiuolaikinėse DBVS bei visas verslo, pramonės, mokslo ir valdymo sritis apimančios duomenų gavybos (žinių valdymo) filosofijos ir metodologijos formavimą.

- Paplitusios DBVS technologijos ir vis didėjančios nuolat kaupiamų duomenų apimtys visuose sektoriuose. Tokiuose kiekiuose duomenų tik naudojant automatines priemones

apskritai įmanoma pastebėti naują svarbią informaciją ir ją pateikti naudotojams patogiu būdu.

- Organizacijose didėja supratimas, kad duomenų bazes galima sėkmingai panaudoti žinioms kurti ir geriau pagrįstiems sprendimams priimti.
- Tradiciniai statistinės analizės ir SQL metodai negali efektyviai aptikti ir išskirti naujų žinių duomenų bazėse, kurioms būdinga labai didelė duomenų apimtis (daug saugomų faktų) ir daug atributų.
- Naujos kompiuterių techninės galimybės ir lygiagretūs skaičiavimai, leidžiantys padidinti duomenų apdorojimo greitį.

Duomenų gavyba yra šiuolaikinių DBVS technologijų dalis, vaidinanti svarbų vaidmenį DBVS evoliucijoje – nuo duomenų tvarkymo iki sprendimų priėmimo palaikymo. Kitaip nei kiek anksčiau išvystyta operatyvaus analitinio apdorojimo tinkle technologija (angl. *on-line analytical processing, OLAP*), duomenų gavyba naudoja ne tik tiesoginės istorinių duomenų analizės metodus, bet ir neuroninius tinklus, genetinius algoritmus, evoliucinius statistinius skaičiavimus, semantinių struktūrų aptikimo (angl. *pattern discovery*) ir kitus metodus. Jai būdingos kelios savybės:

a) duomenų gavybos technologijos sukurtos specialiai naudoti labai didelėse duomenų bazėse ar duomenų saugyklose, su milijonais įrašų ir šimtais, gal net tūkstančiais atributų, t.y., tokiose sudėtingose, kad yra neefektyvūs įprasti analizės metodai;
b) svarbiausia yra netiesioginė duomenų analizė, t.y., tokia, kuria siekiama gauti duomenų bazėje slypinčią anksčiau nežinomą informaciją – struktūras, sąsajas, tendencijas, kurių neįmanoma aptikti kitais būdais dėl duomenų sudėtingumo;
c) taikoma induktyvioji analizės strategija, kai žinios kaupiamos palaipsniui be jokių išankstinių hipotezių ar nuostatų, kaip turi atrodyti rezultatas;
d) siekiama nustatyti sąsajas ir savybes, bet ne įrodyti jų priežasties–pasekmės ryšį, t.y., laikomasi tiriamojo nedeterminuoto požiūrio, o ne aiškinamojo determinuoto, kaip įprastoje analizėje.

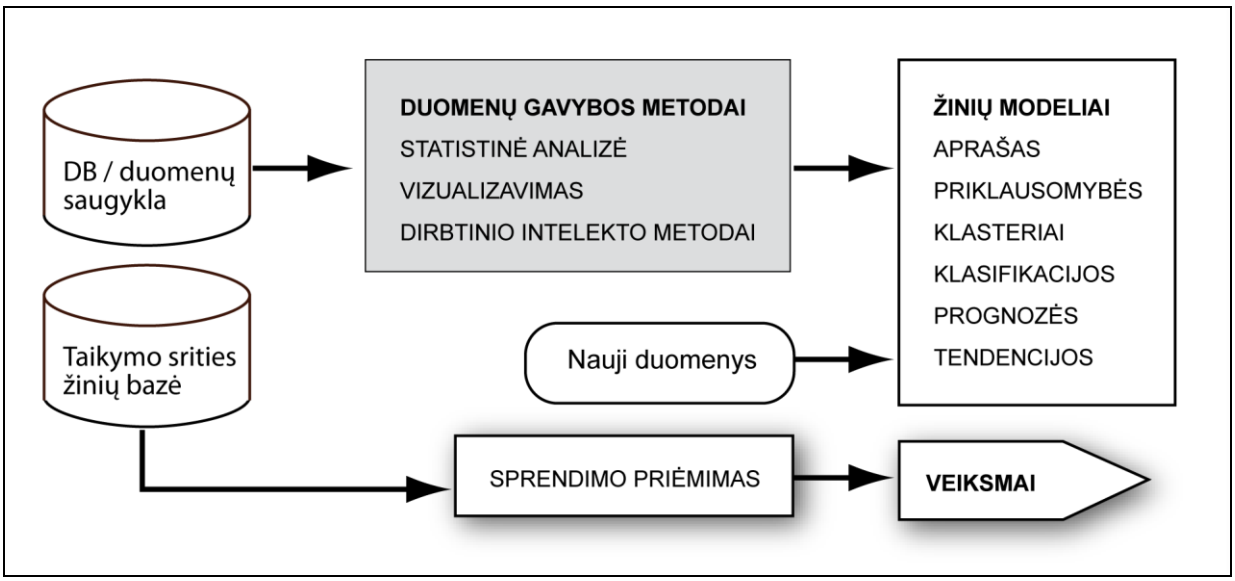
> **Pavyzdys.** 20 a. devintojo dešimtmečio viduryje kanadietis psichologas Rodžeris Bernslis (Roger Barnsley) atsitiktinai pastebėjo, kad dauguma ledo ritulio rinktinės žaidėjų yra gimę sausio, vasario ir kovo mėnesiais. Vėliau toks dėsningumas nustatytas praktiškai visuose geriausių jaunųjų Kanados ledo ritulio žaidėjų sąrašuose – apie 40 procentų žaidėjų buvo gimę tarp sausio ir kovo, 30 procentų – tarp balandžio ir birželio, 20 procentų – tarp liepos ir rugsėjo ir vo 10 procentų – tarp spalio ir gruodžio. Taip pirmą kartą buvo įvardintas „santykinio amžiaus“ fenomenas, kuriam aptikti neprireikė statistinės duomenų analizės.
>
> Šio reiškinio paaiškinimas paprastas. Kanadoje teisės žaisti tam tikroje amžiaus kategorijoje galutinė riba yra sausio 1-oji. Berniukas, kuriam sausio 2 dieną sukanka dešimt metų, gali žaisti kartu su tais, kurie dešimties sulauks metų pabaigoje – tokiame amžiuje dvylikos mėnesių skirtumas reiškia labai didelį fizinės brandos skirtumą. Todėl metų gale gimę sportininkai turi mažesnes galimybes. Mažas pradinis pranašumas, kurį metų pradžioje gimęs vaikas turi prieš vaiką, gimusį metų gale, išlieka.

Duomenų gavybos technologijos naudojamos kuriant elgesio modelius, istoriniais duomenimis pagrįstus modelius ir ateities prognozes, kuriant ir tikrinant įvairias strategijas. Ir versle, ir viešajame sektoriuje ypač svarbi šios technologijos teikiama galimybė suprasti ir susieti ankstesnius scenarijus

su galimu sistemos vystymusi ateityje. Dar platesnė sąvoka, apimanti duomenų gavybą yra **žinių gavyba** (angl. *knowledge discovery*). Žinių gavybai yra būdingi šie etapai:

- duomenų iš skirtingų šaltinių sujungimas ir pradinis tvarkymas (išsprendžiamos klaidingų, trūkstamų, nevientisų duomenų problemos);
- duomenų atranka ir transformavimas, kuri metu su sprendžiamu uždaviniu ar sritimi susiję duomenys atrenkami ir transformuojami (pavyzdžiui, perklasifikuojami, agreguojami ir pan.);
- duomenų gavyba, kai taikomi dirbtinio intelekto, statistiniai ir vizualizavimo metodai tam, kad būtų aptikta paslėpta informacija;
- žinių kūrimas, kuris apima gautų duomenų vertinimą ir interpretavimą bei naujos informacijos įtraukimą į žinių bazę, dokumentavimą ir pateikimą ataskaitų pavidalu;
- taikymas, t.y., praktinis rezultatų panaudojimas pagrįsti sprendimams skirtingose taikymų srityse.

Žinių gavyba yra ne tik iteratyvus, bet ir interaktyvus procesas. Interaktyvumas reiškia, kad proceso metu analitikai gali keisti įvedimo parametrus taip modeliuodami skirtingus scenarijus, susieti tarpusavyje skirtingais metodais gautas žinias. Žmonės nusprendžia, kokie duomenys yra tinkami tiriamajai analizei, kokius duomenų gavybos scenarijus tikslinga naudoti, kada reikia baigti pakartotinį metodų taikymą, kaip pateikti gautas žinias. Ne visa informacija, kurią galima aptikti duomenų gavybos metodais, yra vienodai svarbi. Tikimasi, kad ji bus patikima, nauja (nežinoma ir netikėta), vertinga kuriai nors taikymo sričiai bei lengvai suvokiama. Informacijos svarbą taip pat nustato analitikai, vadovaudamiesi savo žiniomis ir patirtimi.



III-5 pav. Duomenų gavybos pagrindinės sąvokos ir metodai

Paveiksle (III-5 pav.) parodytos pagrindiniai duomenų gavybos metodai. Dirbtinio intelekto metodai remiasi algoritmais, kurių pagalba kompiuterinė sistema mokosi, o mokymo duomenų pagrindu kuria ir vysto modelius, kuriuos taiko naujiems duomenims. Mokymas gali vykti įsikišant žmogui arba be jo. Dirbtinio intelekto metodai apima temporalinę (su laiku susijusių savybių) analizę, sąvokų (klasių su bendromis savybėmis) aprašymą, priklausomybių analizę, klasterių išskyrimą, išimčių ir nuokrypių analizę, klasifikavimą bei prognozavimą.

Labai svarbi yra mokslinė vizualizacija – grafinė analizuojamų duomenų pateiktis. Vizualizavimas aptariamas trimis aspektais:

a) skaičiavimas ir braižymas, t.y., duomenų vertimas grafiniais, dažnai reikalaujantis žmogaus įsikišimo;
b) grafinis dizainas taikant kuriamiems vaizdams grafinės komunikacijos principus;
c) vizualinis pažinimas – žmogaus gebėjimų įžvelgti grafiniame vaizde iš anksto nežinomą informaciją tikslingas panaudojimas.

Teoriškai vizualizavimas yra ideali duomenų gavybos priemonė, nes vaizdus žmonės analizuoja ir daro išvadas intuityviai. Praktiškai, grafinių vaizdų interpretavimas ir vertinimas gali būti labai sudėtingas uždavinys, reikalaujantis specifinių gebėjimų ir patirties. Žemėlapiai yra vieni sudėtingiausių grafinių vaizdų, naudojami **erdvinių duomenų gavybai**. Erdvinių duomenų gavyba dėl geometrinių ir topologinių savybių, didelio erdvinių ryšių skaičiaus, erdvinių savybių priklausomybės nuo objektų aplinkos bei tolydžių reiškinių savaime yra daug sudėtingesnė už skaitinių ar tekstinių duomenų gavybą. Galima sakyti, kad erdvinių duomenų gavyba nagrinėja tolydžią dvimatę ar trimatę geografinę (o kartais ir ne geografinę ir net ne Euklido) erdvę, kai tuo tarpu įprasta duomenų gavyba apsiriboja diskrečia objektų erdve. Įprasti duomenų gavybos algoritmai geografiniams duomenims dažnai netinka. Geometriniams skaičiavimams ir operacijoms reikia specialios programinės įrangos, taigi geografinių duomenų gavybai būdingas ne tik duomenų, bet ir technologinis sudėtingumas. Be to, iki šiol nėra žinomų universalių geografinių duomenų modelių, skirtingose dalykinėse srityse naudojamos skirtingos geografinės erdvės ir geografinių žinių sampratos bei sąvokos. Todėl atliekant geografinių duomenų gavybą ypač svarbu gerai pažinti nagrinėjamą teritoriją ir dalykinę sritį bei suvokti specifinius gavybos tikslus ir jų ryšį su sąvokų hierarchija (pavyzdžiui, bendrasis planavimas ir lokalaus verslo vystymas reikalauja skirtingų duomenų gavybos strategijų).

Geografinių duomenų gavybos tikslas yra atskleisti dar ir erdvines struktūras ir sąsajas, kurios nebuvo nustatytos apibrėžiant duomenis. Didžiulių duomenų bazėse kaupiamus duomenų kiekių fone gali būti labai sunku pastebėti retai pasitaikančias ir nedideles, bet svarbias struktūras. Pavyzdžiui, globalaus klimato šiltėjimo ir cikliško metinio oro temperatūrų kitimo fone lokalūs svyravimai atrodo nereikšmingi. Geografiniai duomenys dažnai kaupiami apibendrinti teritorijoms ar laikotarpiams, dėl ko taip pat gali likti nepastebėta svarbi neakivaizdi informacija. Todėl geografinių duomenų gavybos tikslai dažniausiai apribojami kokia nors teritorija (tuo tarpu įprasta duomenų gavyba orientuota į globalias naujas žinias).

Geografinių duomenų sąvokų hierarchijos konstruojamos remiantis ne vien atributų reikšmėmis. Taip atsiranda naujos klasifikavimo ašys:

- geografinė–atributinė ašis (kai elementarūs objektai yra geografiniai, pavyzdžiui, nusikaltimų vietas atitinkantys taškai, bet juos agreguojant gaunami atributiniai duomenys, pavyzdžiui, nusikaltimų skaičius mieste);
- geografinė–laiko ašis (kai elementarūs objektai yra geografiniai, bet apibendrinti laike jie tampa atributiniais, pavyzdžiui, nusikaltimų skaičiaus variacijos paros metu);
- geografinė ašis (kai ir elementarūs, ir apibendrinimo būdu gauti objektai yra geografiniai, pavyzdžiui, reljefo aukščių taškai ir iš jų gautos izohipsės);

Apskritai, geografinių duomenų gavyba naudoja tuos pačius bendruosius metodus kaip į atributus orientuota duomenų gavyba, skiriasi tik specialiai geografiniams duomenims pritaikyti klasifikavimo, prognozavimo, priklausomybių analizės, klasterių išskyrino ir kiti naudojami

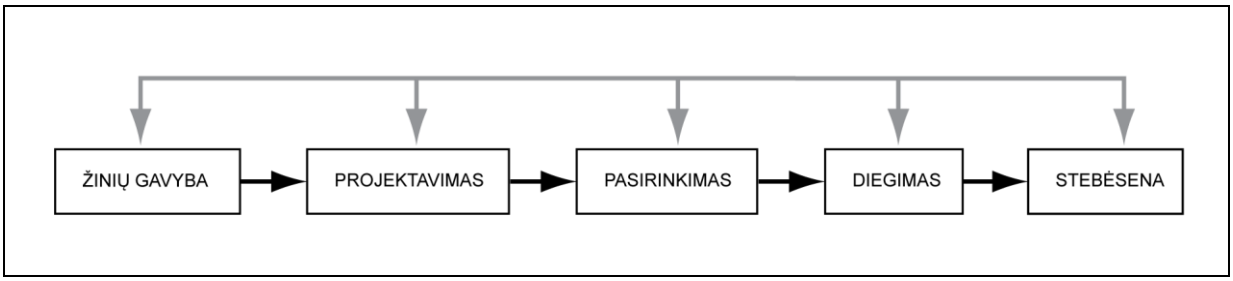
algoritmai. Pavyzdžiui, priklausomybės ryšiai erdvėje apima kolokaciją (objektų buvimą toje pačioje ar gretimose vietose), erdvinę autokoreliaciją (objektų panašumą priklausomai nuo nuotolio); išimčių analizė turi atsižvelgti į išskirtinį objektų dydį ar formą, su laiku susiję duomenys analizuojami ir erdvėje, ir pan.

Vizualizacijos (žemėlapiai) nuolat naudojami geografinių duomenų gavybai. Galima išskirti du požiūrius į jų vaidmenį:

1) dominuoja geografija, t.y., iš pradžių vertinama vizualizacija tam, kad būtų pastebėta kokia nors geografiškai interpretuojama informacija, po to jos patikimumas patvirtinamas matematiniais metodais;
2) dominuoja matematika , t.y., iš pradžių įprastais metodais ieškoma potencialių žinių apie geografinius duomenis, tik po to naudojama vizualizacija joms patikrinti.

Duomenų, ypač geografinių, gavyba yra glaudžiai susijusi su **sprendimų priėmimo palaikymo sistemomis**. Tai interaktyvios kompiuterinės sistemos, kurių pagalba sprendimus priimantys asmenys gali efektyviai panaudoti duomenis ir modelius ir spręsti neaiškios struktūros uždavinius, t.y., uždavinius, kurių negalima aprašyti algoritmais, kurie net neturi vienareikšmio sprendimo.

Pagrindinės sprendimų priėmimo proceso fazės parodytos III-6 pav. Žinių gavybos fazėje identifikuojama ir aprašoma problema, jos svarba ir specifika, nustatomas duomenų poreikis, įvertinamos galimybės. Projektavimo fazėje sukuriami alternatyvūs problemos sprendimo modeliai, iš kurių vienas pasirenkamas įgyvendinti. Įdiegto sprendimo stebėsena būtina jo tinkamumui įvertinti.



III-6 pav. Sprendimų priėmimo proceso fazės

Duomenų bazių valdymo komponentas sprendimų priėmimo palaikymo sistemose be įprastų, atlieka šias papildomas funkcijas:

- leidžia apjungti skirtingų formatų ir skirtingų šaltinių duomenis;
- leidžia kurti modelius ir įrankius duomenų analizei palengvinti.

Sprendimų priėmimo palaikymo sistemose taikomi keturių tipų modeliai:

- strateginiai, naudojami ilgalaikiams sprendimams, tokiems kaip teritorijų planavimas, regionų plėtros politikos formavimas, organizacijos pertvarka;
- taktiniai, takomi konkretiems uždaviniams, tokiems kaip išteklių paskirstymas organizacijoje, logistikos planavimas;
- operatyviniai, naudojami kasdieniniams sprendimams, pavyzdžiui, maršruto sudarymui;
- analitiniai, apimantys visų trijų aukščiau išvardintų modelių aspektus.

Uždaviniai, kuriems būtini geografinius duomenis naudojantys sprendimai dažnai pasižymi išskirtiniu kompleksiškumu, neapibrėžtumu, dideliu įtrauktų naudotojų ir galimų alternatyvų skaičiumi, priklauso nuo konkrečios teritorijos.

🕮 Klausimai diskusijai

Kokį vaidmenį organizacijos IS modelyje vaidina jos darbuotojai? Ar galima jų atsisakyti IS modelyje, viską vaizduojant tik per procesus?

Sugalvokite pavyzdžių, kaip geografiniai duomenys gali būti panaudoti priimant sprendimus apie įmonės plėtrą. Kokio tai tipo uždavinys – strateginis, taktinis, operatyvinis ar analitinis?

✍ Užduotys savarankiškam darbui

Sudarykite geografinių ir projekto duomenų srautų diagramas įsivaizduojamam Europos gyventojų geografijos atlaso ar kito teminio atlaso, skirto mokymui, projektui. Nurodykite srautų šaltinius, adresatus ir tipus. Numatykite, kokie duomenys bus nuolatiniai, kurie – atnaujinami ir nurodykite, kokiu metodu atnaujinami.

💻 Užduotys praktikos darbams (4 val.)

Išbandykite, kaip veikia MS Access *QBE* užklausos. Egzistuojančių lentelių pagrindu sukurkite formas, ataskaitas, kuriose matytųsi norimi laukai. Sukurkite formoje mygtuką, vedlio (wizard) pagalba priskirti jam funkciją. Panagrinėkite, kaip atrodo automatiškai sugeneruotas mygtuko funkcijos kodas modulyje. Išbandykite, kaip veikia makrokomandos.

Panagrinėkite geografinės duomenų bazės struktūrą naudojant *ArcGIS* programą. Palyginkite naudotojo sąsają su *MS Access* RDBVS sąsaja. Panagrinėkite *ArcGIS* lenteles ir užklausų sudarymo priemones.

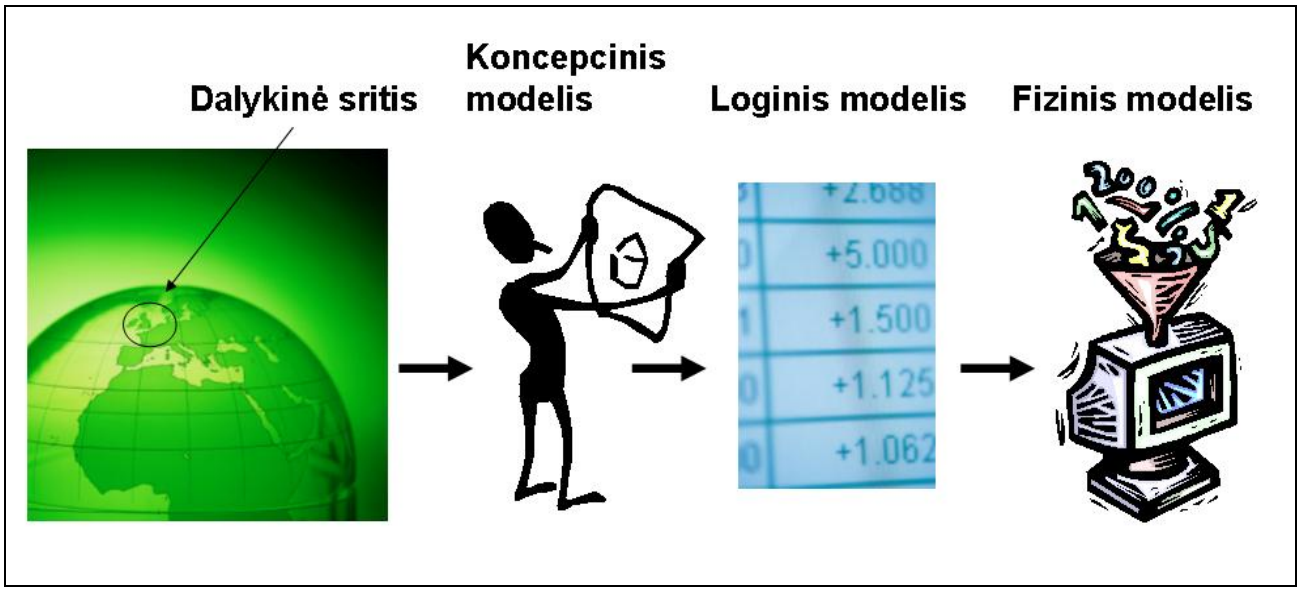
IV. DUOMENŲ BAZĖS KONCEPCINIS MODELIS

Jei tik įmanoma, pakeiskite nežinomas esybes žinomų esybių deriniais.
Bertranas Raselas

Menkiausias nukrypimas nuo tiesos bus padaugintas vėliau.
Aristotelis

IV.1 DUOMENŲ BAZĖS KŪRIMO ETAPAI

Duomenų bazės projektavimas yra ne kas kita, kaip realaus pasaulio dalies informacijos modelio kūrimas. Visiems modeliams – matematiniams, mechaniniams ar grafiniams (pavyzdžiui, žemėlapiai, trimačiai reljefo modeliai) yra bendra tai, kad juos galima apibrėžti kaip labai supaprastintus ir apibendrintus mus supančios tikrovės atvaizdžius. Bet koks modelis, taip pat ir duomenų bazės struktūra, yra kuriamas nuosekliai keliais etapais, kurie pagal taikomų metodų rūšį gali būti grupuojami į stadijas (IV-1 pav.).



IV-1 pav. Duomenų bazės kūrimo etapai

Analizės stadija

Tai pradinė duomenų bazės kūrimo stadija, kurios metu apibrėžiami kūrimo tikslai ir nustatomi patys bendriausi poreikiai. Tai reiškia, kad analizuojant labai sudėtingą ir įvairią realaus pasaulio informaciją atrenkama tai, kas yra tiesiogiai susiję su duomenų bazės kūrėjų tikslais, apie ką bus kaupiami duomenys, ir atmetama likusi informacija, be kurios galima apsieiti. Taip apytikriai nusakoma projekto dalykinė sritis (angl. *universe of discourse*).

Projektuojant duomenų bazę, labai svarbu kiek įmanoma tiksliau apibrėžti dalykinę sritį, t.y., galėti vienareikšmiškai nustatyti, ar konkretus faktas į ją patenka, ar ne. To nepadarius, rizikuojame prarasti svarbią informaciją arba atvirkščiai, paskęsti gausybėje mažai tarpusavyje susijusių ir didele dalimi nesvarbių faktų. Atliekant analizę ir apibrėžiant dalykinę sritį naudojami bendriausi mokslinių tyrimų metodai. Nėra jokių konkrečių nurodymų, ką reikia daryti, kad dalykinė sritis būtų

apibrėžta tinkamai. Tai yra tos srities specialistų kompetencijos klausimas. Neretai reikia ne tik žinių, bet ir didelės patirties, kad būtų galima priimti teisingą sprendimą atskiriant svarbią informaciją nuo ne tokios svarbios.

Projektavimo stadija

Projektavimo stadijos metu sukuriami modeliai, kurie tampa duomenų bazės pagrindu. Apibrėžus dalykinės srities ribas, reikia rasti metodą sutvarkyti, susisteminti ir aprašyti jos turinį paprastu būdu. Be to, toks aprašymas turi būti iš vienos pusės, suprantamas ir patogus žmogui, iš kitos – pakankamai griežto pavidalo tam, kad jį būtų galima perkelti į kompiuterines duomenų struktūras. Šiuo metu tokiam aprašymui naudojami koncepciniai modeliai, t.y., modeliai, kuriuose naudojamos natūralios kalbos sąvokos. Apie sąvokų klasifikavimą ir santykius plačiau rašoma XII.1 ir XII.2 skyriuose. Koncepcinio modelio pagrindu kuriami loginiai modeliai.

Koncepcinis modeliavimas – tai informacijos apie kurią nors realaus pasaulio sritį sisteminimo bei vaizdaus pateikimo metodas. Jį naudojant visa informacija aprašoma kalbos sąvokomis (iš to kilęs metodo pavadinimas) ir dažnai pavaizduojama grafiškai. Kadangi sąvokomis (žodžiais) ir tik jomis galima aprašyti viską, ką suvokiame, akivaizdu, kad koncepcinis modelis turėtų būti priimtinas žmogui. Tačiau sąvokų yra labai daug, todėl norint įvesti jų sistemoje tvarką, tenka sąvokas apibendrinti į abstrakčias klases. Apibendrinant sąvokas pagal jų prasmę gaunamas **semantinis modelis**. Semantiniai modeliai gali skirtis klasių skaičiumi ir vaizdavimo būdais. Esybių ryšių modelis yra palyginti paprastas semantinis modelis, kuriame naudojamos trys sąvokų klasės, o visa dalykinės srities informacija pateikiama diagramomis. Toliau susipažinsime su labai plačiai įvairiose srityse naudojama **esybių ryšių** koncepcinio modeliavimo technika ir išmoksime ją taikyti.

Koncepcinis dalykinės srities modelis toliau gali būti naudojamas įvairiais tikslais: pristatyti dalykinę sritį nespecialistui, palyginti ją su kita dalykine sritimi, formuoti paprastos arba geografinės duomenų bazės struktūrai, programuoti algoritmams ir t.t. Koncepcinį modelį turi būti galima lengvai atvaizduoti į formalias matematines ar logines struktūras, kurios nebūtinai gerai atspindi žmogišką tikrovės suvokimą, bet yra patogios norint apdoroti duomenis automatiškai. Tokios struktūros sudaro dalykinės srities **loginį modelį.** Toliau šioje knygoje išsamiai nagrinėsime loginio modelio atvejį – reliacinį DB modelį, kuriam būdingos lentelių pavidalo struktūros.

Projektavimo metu gali būti kuriama keletas įvairaus detalumo koncepcinių modelių, tačiau paprastai tik vienas loginis modelis. Kai kuriais atvejais projektuotojo požiūriu galima apsieiti su vienu modeliu – galima įsivaizduoti nesudėtingą reliacinę duomenų bazę arba objektinę duomenų bazę, kurios duomenų struktūros yra praktiškai identiškos koncepcinio modelio struktūroms, o atvaizdavimą automatiškai atlieka tam skirta programa. Vis dėlto, dažniausiai koncepcinis ir loginis modeliavimas yra atskiriami.

Įgyvendinimo stadija

Sukūrus loginį modelį ir išsaugojus jį DB formatu, galima sakyti, kad jau turime vietą duomenims saugoti, arba „tuščią“ duomenų bazę. Belieka ją **užpildyti** sukauptais faktais, kurie atitinka duomenų bazės struktūrą ir koncepcinį modelį. DBVS paprastai turi priemones patikrinti, ar įvedami faktai neprieštarauja šių modelių sprendimams (užtikrinti DB vientisumui).

Paskutinysis etapas, kuriame pagal loginį modelį atliekamas **fizinis** duomenų kodavimas, šiame kontekste nėra svarbus – jo rezultatas yra vienaip ar kitaip saugomi duomenys dvejetainiu formatu.

Loginio duomenų bazės modelio pavertimą fiziniu paprastai automatiškai atlieka duomenų bazių valdymo sistemos. Todėl šio etapo atskirai nenagrinėsime.

IV.2 MODELIAVIMO REIKŠMĖ

Modeliavimo svarbą galima pailiustruoti pavyzdžiu, kurį pateikia UML kūrėjai Greidis Bučas, Džeimsas Rambo ir Aivaras Jakobsonas (Grady Booch, James Rumbaugh, Ivar Jacobson).

Jei reikia būdos šuniui, galima pradėti turint keletą lentų, vinių bei pagrindinius įrankius – pjūklą, plaktuką ir matavimo juostą. Per keletą valandų šiek tiek apgalvojus iš anksto, be jokios pagalbos įmanoma sukurti pakankamai funkcionalų statinį. Jei tik jis yra pakankamai didelis ir lyjant nepraleidžia vandens, šuo turėtų būti patenkintas. Jei taip nėra, galima sukalti naują būdą arba įsigyti ne tokį reiklų šunį.

Jei norite pastatyti namą savo šeimai, taip pat galima pradėti nuo rąstų ir pagrindinių įrankių. Tačiau toks darbas nesitęs ilgai, nebent jau esatę tą anksčiau darę daug kartų. Be to, ir šeima bus žymiai reiklesnė, negu šuo. Tokiu atveju dar prieš įkalant pirmąją vinį geriau pasidaryti keletą eskizų, kaip namas turėtų atrodyti. Jei norite gero namo, kuris atitiktų jūsų šeimos poreikius ir statybas reglamentuojančius dokumentus, teks parengti detalius planus, numatant patalpų paskirtį, elektros, šilumos ir nuotėkų įrangą. Pagal tokius planus galima apytiksliai įvertinti, kiek laiko ir medžiagų prireiks statybos darbams. Nors teoriškai įmanoma viską atlikti vienam, daug paprasčiau dirbti keliese, be to dar naudojantis įvairiomis paslaugomis ir gatavais produktais. Jei perdaug nenukrysite nuo plano ir sąmatos, tikėtina, kad rezultatas bus priimtinas. Be abejo, nesėkmės atveju pasekmės bus kur kas skaudesnės.

Jei statote komercinės paskirties daugiaaukštį, pradėti nuo medžiagų ir įrankių būtų neįtikėtinai kvaila. Visi investuotojai turės reikalavimų pastato dydžiui, formai ir stiliui. Be to, pradėjus statybą, tie reikalavimai dažnai keisis. Būtina labai detaliai planuoti, nes nesėkmės kaina didelė. Darbus atliks vykdytojų grupė, kuriai reikės įvairiausių planų ir tarpusavio bendravimo priemonių. Jei žmonės bus parinkti tinkamai, o jūs nuolat stebėsite ir valdysite architekto idėjos įgyvendinimą, galima tikėtis, kad pastatas atitinks naudotojų poreikius. Darant tą daug kartų, išmokstama naudotojų reikalavimus derinti su technologijų ar finansų diktuojamomis sąlygomis ir nerizikuoti prisiimant nerealius įsipareigojimus.

Daugelio elektroninius produktus, tarp jų ir geografines duomenų bazes bei žemėlapius kuriančių įmonių problema ta, kad įgyvendindamos projektus, prilygstančius daugiaaukščių statybai, jos dirba tarsi statydamos šuns būdą. Kartais projektai pavyksta, sėkmingai susiklosčiusių aplinkybių dėka (tinkami žmonės tinkamu laiku ir pan.). Tačiau jei nesiseka, įdėtas papildomas didelis darbas visai nebūtinai pasiteisins. Šuns būda, pasiekusi daugiaaukščio dydį, sugrius nuo savo pačios svorio. Nesėkmingi projektai žlunga dėl labai įvairių priežasčių, tuo tarpu visi sėkmingi projektai yra panašūs jų valdymo požiūriu. Vienas jungiantis ypatumas – juose kryptingai taikomas projektų modeliavimas.

Geras modelis yra supaprastintas realaus objekto ar sistemos vaizdas, kuriame yra visi svarbūs elementai, ir neįtraukti tie, kurie nereikšmingi modelio mastelyje (to paties objekto modelis gali būti ir labai detalus ir labai bendras). Kiekviena sistema gali būti aprašyta skirtingais aspektais naudojant skirtingus modelius, kurių kiekvienas tokiu būdu yra semantiškai uždara abstrakcija. Modelis gali būti **struktūrinis**, aprašantis sistemos komponentus ir jų ryšius, arba **elgsenos**, aprašantis sistemos dinamiką.

Modeliai kuriami iš esmės tam, kad būtų galima geriau suprasti kuriamą sistemą, nes sudėtingų sistemų žmonės jau nesugeba vertinti kaip visumos. Modeliai leidžia susiaurinti nagrinėjamas problemas iki vieno konkretaus sistemos aspekto vienu metu ("skaldyk ir valdyk" principas) arba aprėpti visą sistemą aukštesniu abstrakcijos lygmeniu. Taip pasiekiami keturi tikslai:

1. Modelis padeda vizualizuoti esamą ar norimą sistemą;
2. Modelis leidžia aprašyti sistemos struktūrą ar elgseną;
3. Modelis naudojamas kaip sistemos kūrimo šablonas;
4. Modelis yra sprendimus aprašantis dokumentas.

Kadangi modeliavimas taip pat reikalauja darbo sąnaudų, jis turi pasiteisinti. Labai paprastuose projektuose nėra prasmės kurti sudėtingus modelius. Tačiau modeliavimas, nors ir neformalus, neišvengiamai vyksta visada.

Modelis turi būti **tinkamas**, toks, kad leistų visada įžvelgti sistemos esminius ypatumus ir galimus pavojus ją kuriant. Blogas modelis gali nukreipti dėmesį į nesvarbius aspektus. Apskritai, nuo modelio priklauso, kokia būtent sistema galų gale bus sukurta.

Modelis turi **atitikti realias sąlygas**. Vien tik idealiomis sąlygomis veikiantys modeliai nepasiteisina. Taigi, supaprastinant svarbu nepraleisti svarbių detalių.

Sudėtingoms sistemoms dažniausiai neužtenka vieno koncepcinio modelio. Tenka naudoti daug praktiškai nepriklausomų (tačiau, be abejo, susijusių tarpusavyje) modelių, aprašančių skirtingus sistemos aspektus iš skirtingų požiūrio taškų.

IV.3 KONCEPCINIO MODELIO PAGRINDINĖS SĄVOKOS

Projektuojant duomenų bazę visų pirma reikia susigaudyti įmanomų gauti duomenų chaose, juos atrinkti, kokiu nors būdu sutvarkyti ir susieti tarpusavyje. Koncepcinio modeliavimo terminas reiškia duomenų loginį struktūrizavimą pagal jų prasmę. Dažnai duomenys tvarkomi neatsižvelgiant į prasmę ar interpretaciją, pavyzdžiui, pagal abėcėlę; tuo tarpu koncepciniai modeliai asocijuojasi su realiais objektais ir jų savybėmis, tokiomis, kokias jas suvokia žmogus, todėl yra lengvai suprantami. Vienas pirmųjų koncepcinio modeliavimo metodų yra modeliavimo technologija, sukurta prieš tris dešimtmečius ir naudojama labai skirtingose projektavimo srityse. Tai **esybių-ryšių (ER)** modeliavimo technologija, kurią 1976 m. pirmą kartą pasiūlė amerikiečių mokslininkas Piteris Čenas (Peter Pin-Shan Chen). Duomenų bazė taip pat pradedama projektuoti nuo koncepcinio modelio, kurio sutartiniais žymėjimais aprašomi visi objektai, apie kuriuos numatoma saugoti informaciją.

Šiuo metu plačiausiai naudojama koncepcinio modeliavimo technologija yra **UML** (angl. *Unified Modeling Language*) – vieninga modeliavimo kalba. Nauji analizės ir projektavimo metodai vystėsi daugiausia 9-tame 20 a. dešimtmetyje kartu su objektinėmis programavimo kalbomis – *Simula-67*, *SmallTalk*, *C++*, *Eiffel*, kurios leido kurti žymiai sudėtingesnę programinę įrangą. Vieningos modeliavimo kalbos vystymas buvo pagrįstas trimis aplinkybėmis.

1. Įvairios tuo metu kuriamos modeliavimo kalbos savaime panašėjo viena į kitą.
2. Apjungiant į gerai išvystytą sistemą skirtingų modeliavimo kalbų metodus, galima pasiekti jos masinio naudojimo projektuose.
3. Atsirado galimybė perimti ankstesnių modeliavimo kalbų patirtį ir pabandyti išspręsti likusias problemas.

Unifikavimo tikslai buvo apibrėžti taip:

1. Modeliuoti sistemas nuo idėjos iki galutinio (veikiančio) produkto;
2. Tinkamai vystyti sudėtingas, tikslų požiūriu kritinės svarbos sistemas;
3. Sukurti modeliavimo kalbą, tinkamą ir žmogui, ir kompiuteriui.

Pagrindinis iššūkis kuriant UML buvo rasti pusiausvyrą tarp pageidaujamo paprastumo ir pakankamų išraiškos galimybių. Oficialiai pirmą kartą UML projektas pradėtas 1994 metais, o pirmasis kalbos variantas parengtas 1995 m. spalį. 1996 m. sukurtas UML konsorciumas iš keleto organizacijų (tarp jų Hewlett-Packard, IBM, Microsoft, Oracle, Rational), pasiryžusių investuoti į išsamios ir efektyvios modeliavimo kalbos sukūrimą.

1997 metais UML kalbos pirmoji versija buvo pasiūlyta standartizuoti bei priimta OMG (angl. *Object Management Group*) grupės. Po daugelio pataisymų 2005 m. patvirtinta UML 2.0 versija. UML specifikacijos paskutiniai dokumentai skelbiami OMG Interneto svetainėje www.omg.org.

Nė vienas objektas nėra visiškai nepanašus į jokį kitą. Todėl jį visada galima priskirti kokiai nors klasei. Yra klasės, apimančios labai didelį objektų kiekį; jos žymimos labai bendromis sąvokomis. Aristotelis tokias klases vadino kategorijomis ir išskyrė dešimt kategorijų (absoliučiai kiekviena sąvoka gali būti priskirta vienai iš jų): substancija, kiekis, kokybė, santykis, vieta, laikas ir kt. Vėliau Aristotelio kategorijos buvo dar labiau apibendrintos ir dabar bendriausios sąvokų klasės yra tik trys: esybė, savybė ir ryšys.

Esybė – tai savarankiškas, atskiriamas nuo kitų objektas, apie kurį norime turėti informaciją, viena iš trijų bendriausių sąvokų kategorijų, kalboje dažniausiai išreiškiama daiktavardžiu. **Savybė**, dar vadinama **atributu**, yra priklausoma sąvoka, išreiškianti kokio nors objekto ypatybę. Ji negali egzistuoti savaime be to objekto, kurio ypatybę nusako. Kalboje savybės kategorija dažniausiai nurodoma būdvardžiu. **Ryšys** – tai sąsaja tarp esybių ar jų savybių, paprastai nusakoma veiksmažodžio konstrukcija.

Objektu duomenų bazių kontekste vadinama esybė, kuri pasižymi apibrėžtumu ir konkretumu, pakankamu, kad apie jos egzempliorius būtų galima kaupti duomenis. Akivaizdu, kad šiai grupei nepriklausančios esybės, tokios, kaip *laimė, tvarka, Dievas* ir pan., projektuojant duomenų bazes atsiranda tik išskirtiniais atvejais, tačiau ir tada jos sukonkretinamos iki įprastos objekto sampratos (pavyzdžiui, *dievas* apibrėžiamas kaip konkretaus tikėjimo objektas, pasižymintis konkrečiomis praktiškai aprašomomis savybėmis). Todėl toliau tekste sąvokas *esybė* ir *objektas* laikysime sinonimais[1]. Analogiškai, sinonimai yra savybė (kitokiame kontekste ji gali būti ir abstrakti, neaiškiai įvardinta, nevienareikšmiškai suprantama, pavyzdžiui, *gerumas*) ir *atributas*.

Atributas yra apibrėžta esybės savybė, įgyjanti konkrečią reikšmę, kai kalbama apie konkrečią esybę. Ji padeda nustatyti esybės kokybę, kiekybę ar būseną, ją identifikuoti ar klasifikuoti. Esybės vykdomos funkcijos taip pat gali būti laikomos jos atributais. Akivaizdu, kad net ir paprasčiausias realus objektas gali turėti tūkstančius įvairių atributų, todėl būtina atrinkti tik tuos iš jų, kurie privalo būti duomenų bazėje, jei reikia, juos apibendrinti, o neretai ir sukonstruoti naujus, kuriais

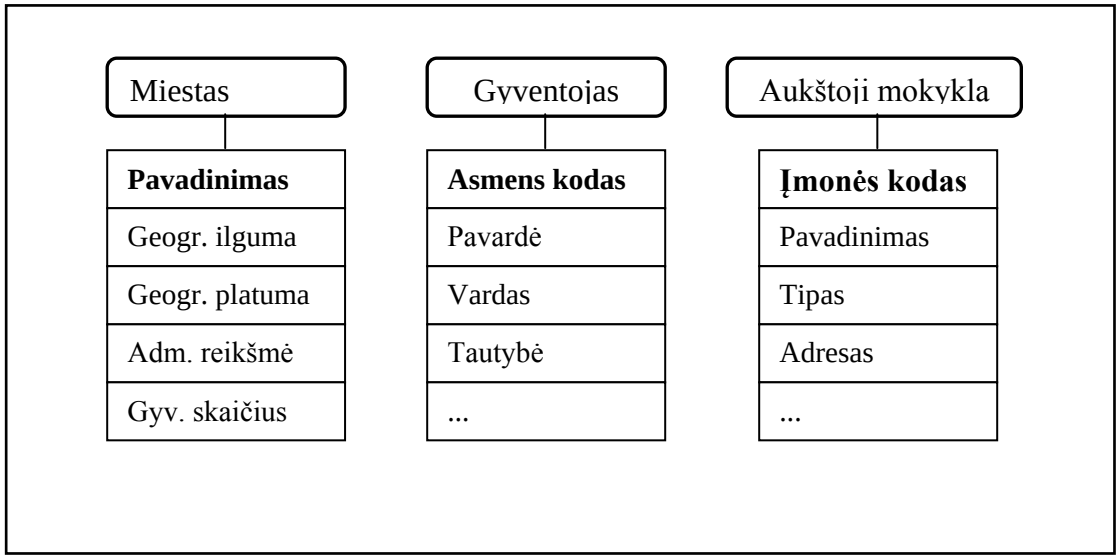
[1] *Esybė* – tai filosofijos terminas, naudojamas, kai kalbama apie bendriausią savoką, koncepciją. *Objektas* – tai informatikos terminas, dažniau suprantamas kaip konkretus daiktas arba skaitmeninis jo vaizdas duomenų bazėje, žemėlapyje ar kitoje informacinėje struktūroje. Anglų kalboje dar naudojamas terminas *feature (*dažnai klaidingai pažodžiui verčiama kaip *savybė* ar *požymis)* paprastai reiškia *geografinį objektą*, vaizduojamą žemėlapyje ar duomenų bazėje.

nepasižymi realūs objektai, pavyzdžiui, priskirti ežerams numerius. Informacinėje sistemoje esybė paprastai turi nuo dviejų iki dešimties atributų, nors kai kuriais atvejais jų gali būti ir daugiau. Gali būti ir neprivalomi atributai, kuriems leidžiama nenurodyti jų reikšmės, t.y., laikyti, kad kuriuo nors laiko momentu to atributo reikšmė yra nežinoma.

IV.4 ESYBĖS, ATRIBUTAI IR DOMENAI

Pirmasis duomenų bazės projektavimo etapas ir yra visų dominančių esybių įvardijimas, atranka, klasifikavimas bei reikšmingų esybių atributų ir jų domenų nurodymas. Norint jį teisingai atlikti, svarbu gerai suprasti dalykinę sritį, kurią apims projektuojama duomenų bazė, todėl visada reikia skirti laiko dalykinės srities analizei, konsultuojantis su tos srities specialistais ir su duomenų bazės užsakovais.

Iš tikrųjų esybė yra ne konkretus objektas, bet objekto sąvoka, abstrakcija, pavyzdžiui, *ežeras*, kuriam nurodytas atributų rinkinys pakankamas konkrečiam tikslui, kuriuo ta informacija bus saugoma duomenų bazėje, pavyzdžiui, kartografavimui, inventorizavimui, hidrocheminiams tyrimams ar kt. Konkretus ežeras, pavyzdžiui, Tauragnų ežeras, yra tos **esybės egzempliorius**, t.y., konkretus objektas su konkrečiomis *visų* jo atributų reikšmėmis. Pagal šių reikšmių rinkinį esybės egzempliorių galima atskirti nuo kitų tos pačios esybės egzempliorių. Iš principo sistemoje negali būti dviejų identiškų rinkinių, kitaip du egzemplioriai neatskiriamai sutaptų. Todėl projektuojant duomenų bazę reikalaujama nurodyti esybės savybę (arba keletą savybių), pagal kurią vienareikšmiškai galime atpažinti tos esybės kiekvieną egzempliorių. Tokia savybė (arba jų rinkinys) vadinama esybės **unikaliu indentifikatoriumi**. Asmens unikalus identifikatorius yra, pavyzdžiui, jo asmens kodas, o pavardė tokiu identifikatoriumi būti negali, nes pasitaiko vienodų pavardžių; ežeras vienareikšmiškai atpažįstamas pagal jo kranto linijos koordinates arba numerį kadastre ir pan.



IV-2 pav. Esybių ir jų atributų pavyzdžiai.

Esybės paprastame esybių ryšių modelyje žymimos stačiakampiais, kuriuose rašomi jų vardai vienaskaita. Nepriklausomai nuo to, kiek gali būti esybės egzempliorių, kiekviena esybė modelyje

yra tik viena, nes tai yra abstrakcija, klasė. Atrinkti atributai išvardijami lentelėje, sujungtoje su esybės stačiakampiu, kaip parodyta IV-2 pav. Paryškinti pasirinkti unikalūs esybių identifikatoriai.

Projektuojant duomenų bazę kiekvienam atributui turi būti apibrėžta ir jo įgyjamų reikšmių aibė – **domenas**, kuri paprastai apribojama sistemos kontekste. Pavyzdžiui, *ežero* druskingumas gali būti loginis kintamasis (taip/ne), ištirpusių mineralinių medžiagų procentas (realus skaičius intervale nuo 0 iki 100) arba kiekis gramais litre (realus teigiamas skaičius) priklausomai nuo to, kaip ir kam šio atributo reikšmės bus naudojamos. Apibrėžiant domeną, svarbu suprasti, kad jis apima ne tik reikšmes, kurias įgyja žinomų esybės egzempliorių atributai, bet **visas įmanomas** tokių atributų reikšmes. Pavyzdyje esybės *Miestas* atributas *Pavadinimas* yra iš baigtinio ilgio tekstinių simbolių eilučių domeno, o faktas, kad jis pasirinktas šios esybės unikaliu identifikatoriumi, reiškia, kad projektuojamoje duomenų bazėje niekada, jokiomis aplinkybėmis, negalės būti dviejų miestų vienodais pavadinimais. Unikalūs identifikatoriai gali būti dviejų tipų.

1. Universalūs, kurių unikalumas absoliutus. Tokie dažniausiai yra specialiai unikalumui užtikrinti sukurti identifikatoriai, pavyzdžiui, numeris, kurio reikšmės esybės egzemplioriams priskiriamos iš eilės didėjančia tvarka. Jei dėl identifikatoriaus reikšmių sutapimo atsiradusios pasekmės būtų reikšmingos, būtina naudoti universalų unikalų identifikatorių.
2. Sąlyginiai, kurių unikalumas galioja dalykinės srities ribose. Tai, pavyzdžiui, gali būti gyventojo asmens kodas Lietuvos Respublikoje (nėra garantijos, kad kuri nors jo reikšmė atsitiktinai nesutaps su kitos valstybės gyventojo asmens kodu), darbuotojo pažymėjimo numeris organizacijoje (gali sutapti skirtingų įstaigų darbuotojų pažymėjimų numeriai) ir pan. Sąlyginiai unikalūs identifikatoriai dažnai turi papildomą prasmę, yra natūrali identifikuojamo esybės egzemplioriaus savybė, todėl juo patogu naudoti, kai atsitiktinio jų reikšmių sutapimo rizika yra labai maža, o pasekmės ne kritiškos.

Ne visais atvejais lengva nustatyti ar sąvoka reiškia esybę, ar atributą. Pavyzdžiui, daiktavardis *pasas* reiškia lyg ir savarankišką esybę – dokumentą, tačiau praktiškai kur kas dažniau domina ne pasas kaip esybė, o tik jo numeris tiek, kiek tai svarbu aprašant kitos esybės (žmogus, pilietis) egzempliorius. Tada praktiškiau yra modeliuoti paso numerį kaip esybės *Žmogus* atributą. Ir atvirkščiai, dažnai laikoma, kad *Miestas*, kaip gyvenamoji vieta, yra esybės *Gyventojas* atributas, tačiau jei dalykinė sritis apima miestus ir įvairias jų charakteristikas bei ryšius, prasminga išskirti esybę *Miestas*, o su esybe *Žmogus* ja susieti ryšiu. Be to, esybė *Gyventojas* ne visiškai atitinka esybės reikalavimus – ji nėra savarankiška, nes sąvoka „gyventojas“ visada siejama su kokiu nors teritoriniu vienetu, t.y., nėra nepriklausoma. Pavyzdžiui, „Žemės gyventojas“, „kaimo gyventojas“ ir pan. Taigi, išskiriant esybes ir jų atributus būtina atsižvelgti į dalykinės srities ribas ir kitas jos sąvokas (kontekstą).

IV.5 KONCEPCINIO MODELIO ELEMENTŲ SĄSAJOS

Be ryšio.
Populiarus apibūdinimas

Intuityviai turėtų būti aišku, kad IV-2 pav. pateiktas modelis nėra informatyvus, nes nenurodyta, kodėl išskirtos būtent šios esybės ir kaip jos susijusios tarpusavyje. Duomenų bazėje visos esybės egzistuoja vienu ar kitu būdu susietos su kitomis esybėmis, kitaip jų buvimas neturi prasmės. IV-3 pav. parodyta, kaip anksčiau sukurtos esybės galėtų būti susietos tarpusavio ryšiais.

Ryšys – tai turinti pavadinimą asociacija tarp dviejų esybių, diagramose žymima esybių stačiakampius jungiančia linija. Tos esybės nebūtinai turi būti skirtingos, t.y., ryšys gali egzistuoti tarp esybės ir jos pačios. Priminsime, kad esybė yra abstrakcija, galinti turėti skirtingą egzempliorių skaičių.

IV.5.1 Ryšio savybės

Kalbant apie santykį tarp dviejų esybių kyla trys klausimai:

- kokio pobūdžio yra tas santykis;
- ar kuriame nors ryšio gale gali nebūti nė vieno esybės egzemplioriaus;
- kiek esybės egzempliorių gali būti kiekviename ryšio gale.

Kad diagrama galėtų į juos atsakyti, ryšiai visada įvardijami, t.y., užrašomas jų pavadinimas. Be to, dar nurodomos ryšių savybės: **privalomumas** ir **kardinalumas**.

Privalomumas – tai ryšio savybė, parodanti, ar atitinkamame gale gali nebūti nė vieno susietos esybės egzemplioriaus. Pavyzdžiui, žmogus gali turėti automobilį, bet gali jo ir neturėti, tuo tarpu automobilis visada turi savininką. Todėl šiuo atveju automobilio pusėje ryšys yra neprivalomas, o žmogaus pusėje - privalomas. Neprivalomas ryšys žymimas punktyru.

Kardinalumas – tai ryšio savybė, rodanti, kiek atitinkamame gale gali būti susietos esybės egzempliorių, jei jų yra. Šią savybę turi ir privalomi, ir neprivalomi ryšiai. Dažniausiai neįmanoma nustatyti nei tikslaus egzempliorių skaičiaus, nei skaičių intervalo. Todėl modelyje žymimi tik du variantai.

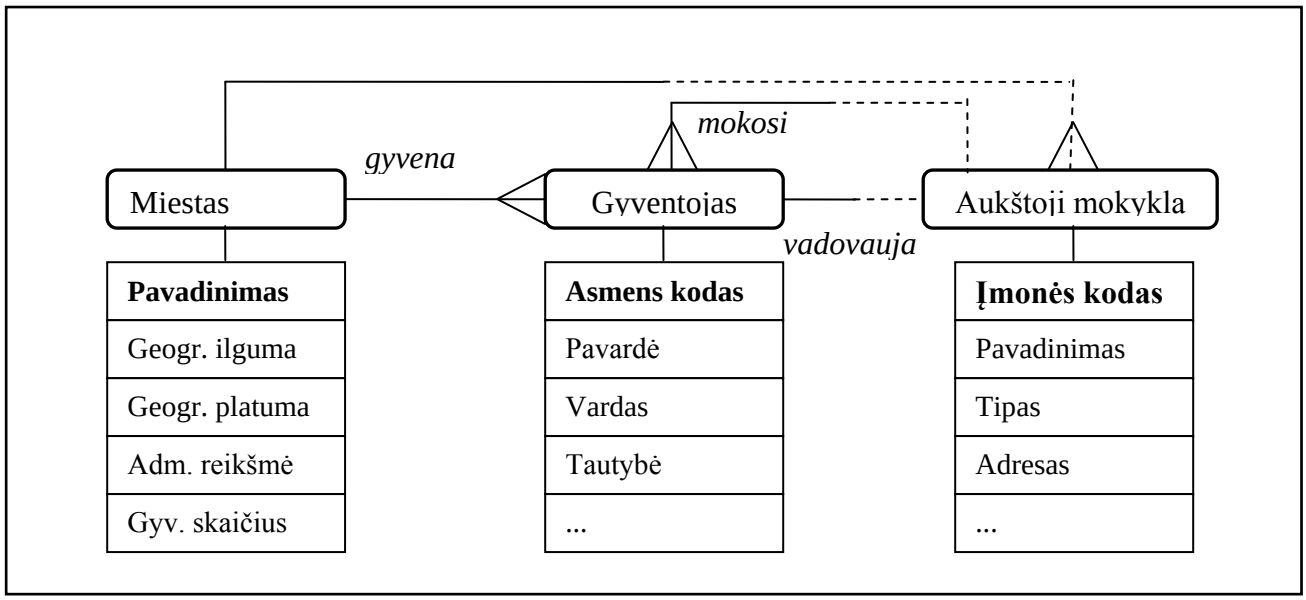
1. „Vienas“, t.y., ryšio gale gali būti ne daugiau kaip vienas esybės egzempliorius. Tokiu atveju ryšio gale yra paprasta ištisinė ar punktyrinė (jei ryšys neprivalomas) linija.
2. „Daug“, t.y., ryšio gale gali būti vienas ar daugiau esybės egzempliorių. Tokiu atveju ryšio linijos gale braižoma „šakutė“, kaip parodyta IV-3 **pav.** paveiksle. Jei galima nurodyti tikslų esybės egzempliorių skaičių, pavyzdžiui, savaitę sudaro 7 dienos, jis rašomas šalia „šakutės“. Jei galima nurodyti tik intervalą ar apribojimą, pavyzdžiui, žmogaus gimimo metai > 0, tai taip pat užrašoma modelyje.

Ryšys visada turi du galus, kurių kiekvienas yra įvardijamas taip, kad ryšį būtų galima perskaityti iš diagramos kaip sakinį. Ryšys be vardo neturi prasmės, nes tuo atveju nežinome, ką jis reiškia. Prie kiekvieno ryšio galo parodomas susietų esybių egzempliorių skaičius (viena, daug, arba konkretus skaičius) ir ryšio privalomumas (privalomas arba neprivalomas).

Ryšius tarp esybių IV-3 **pav.** paveiksle galima perskaityti iš abiejų galų naudojant formalią sintaksę.

1. MIESTE gyvena vienas arba daugiau GYVENTOJŲ. / GYVENTOJAS gyvena vieninteliame MIESTE. Išsišakojimas ryšio gale rodo, kad juo gali būti susietas daugiau negu vienas atitinkamos esybės egzempliorius.
2. MIESTE gali būti viena arba daugiau AUKŠTŲJŲ MOKYKLŲ. / AUKŠTOJI MOKYKLA būtinai yra vieninteliame MIESTE. Punktyrinė ryšio linija rodo, kad tame gale atitinkamos esybės egzemplioriaus gali nebūti, pavyzdžiui, yra miestų, kuriuose nėra nė vienos aukštosios mokyklos.
3. GYVENTOJAS gali mokytis (neprivalomas ryšys) vienintelėje AUKŠTOJOJE MOKYKLOJE. / AUKŠTOJOJE MOKYKLOJE būtinai mokosi ne mažiau negu vienas GYVENTOJAS.

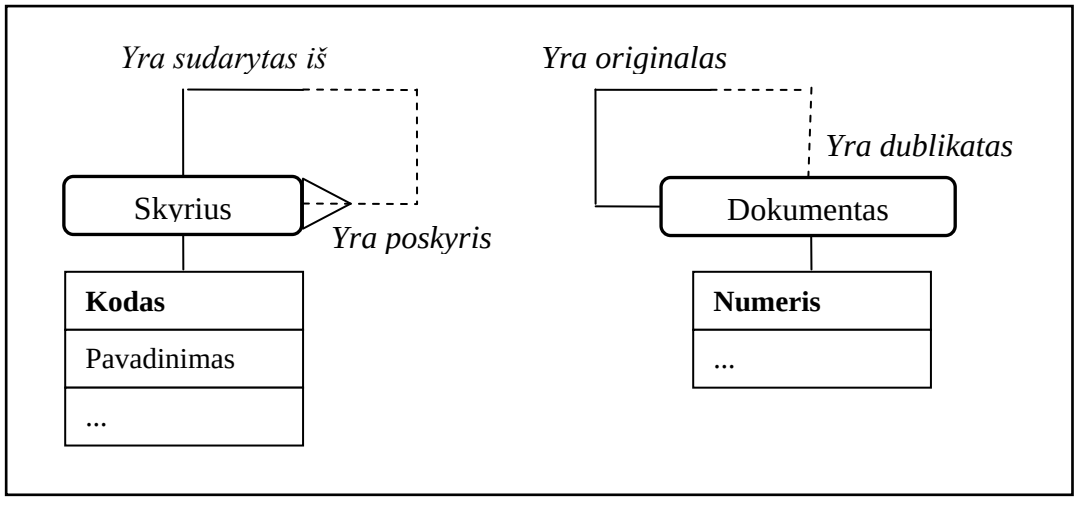
4. GYVENTOJAS gali vadovauti vienintelei AUKŠTAJAI MOKYKLAI. / AUKŠTAJAI MOKYKLAI būtinai vadovauja vienintelis GYVENTOJAS.



IV-3 pav. Esybių ryšių modelis su nurodytais ryšiais

Modeliuojant labai svarbu išvengti dviprasmybių, nenaudoti santrumpų ar žargono, nes nuo to, kiek logiškas, aiškus ir nedviprasmiškas bus koncepcinis modelis, labai priklauso tolesnis duomenų bazės projektavimas ir programavimas. Jei sudarytas esybių ryšių modelis yra visiškai aiškus, nekils jokių sunkumų perkeliant tokį modelį į loginę duomenų bazės struktūrą. Paprastas būdas patikrinti modelio tinkamumą – įsitikinti, kad jame yra vienintelė ir nekelianti abejonių vieta įrašyti kiekviena svarbiam dalykinės srities faktui.

Galimas ir rekursinis ryšys, kai esybės susieta su savimi pačia. Žinoma, tai nereiškia, kad ryšys sieja du tuos pačius esybės egzempliorius. Toks ryšys neretai pasitaiko kai naudojamas iš vienos pusės neprivalomas ryšys „vienas su vienu“ alternatyvai vaizduoti ir ryšys „daug su vienu“ hierarchijai vaizduoti (IV-4 pav.).



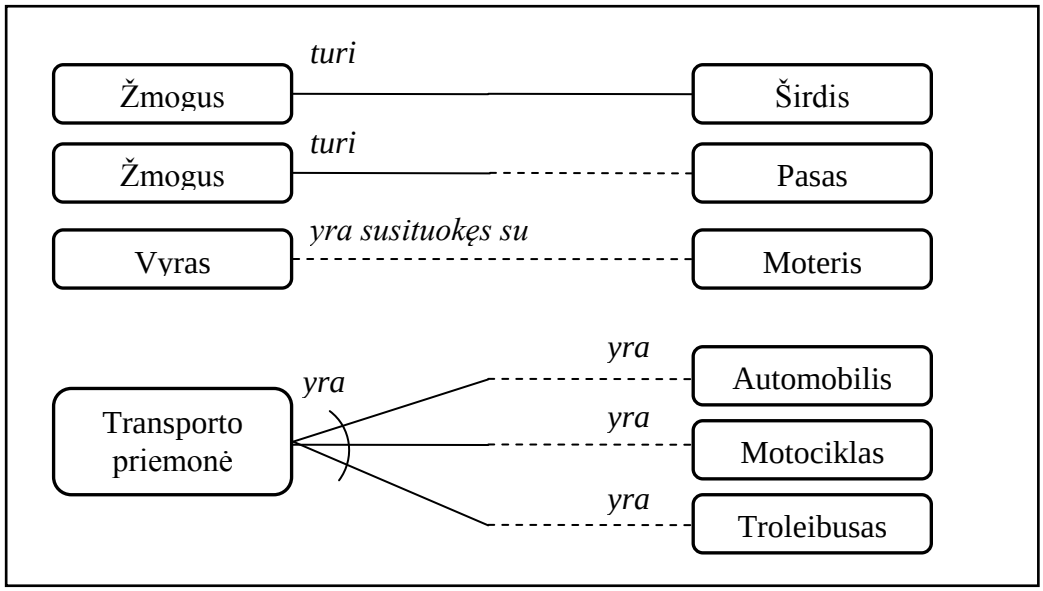
IV-4 pav. Rekursiniai ryšiai.

IV.5.2 Ryšiai „vienas su vienu“

Ryšys „vienas su vienu“ yra palyginti retas, nes reiškia labai griežtai apibrėžtą esybių sąsają.

Privalomas iš abiejų pusių ryšys rodo ypač tvirtą, nekintamą ir vienareikšmę sąsają tarp esybių, pavyzdžiui, vyro ir žmonos ryšys santuokų duomenų bazėje (nesusituokę žmonės šiuo atveju apskritai nedomina). Neretai toks ryšys pasirodo klaidingas, kai iš tikrųjų yra ne dvi skirtingos, o tik viena esybė, t.y., viena iš tokiu ryšiu sujungtų esybių nėra savarankiška. Pavyzdžiui, atrodo, kad privalomas iš abiejų pusių ryšys sieja žmogų ir jo pasą – bet iš tikrųjų pasas tik retais atvejais gali dominti kaip atskira esybė; todėl paprasčiau laikyti paso numerį piliečio atributu.

Privalomas iš vienos pusės ryšys šioje kategorijoje labiausiai įprastas. Toks yra ir IV-4 pav. alternatyvą vaizduojantis ryšys.



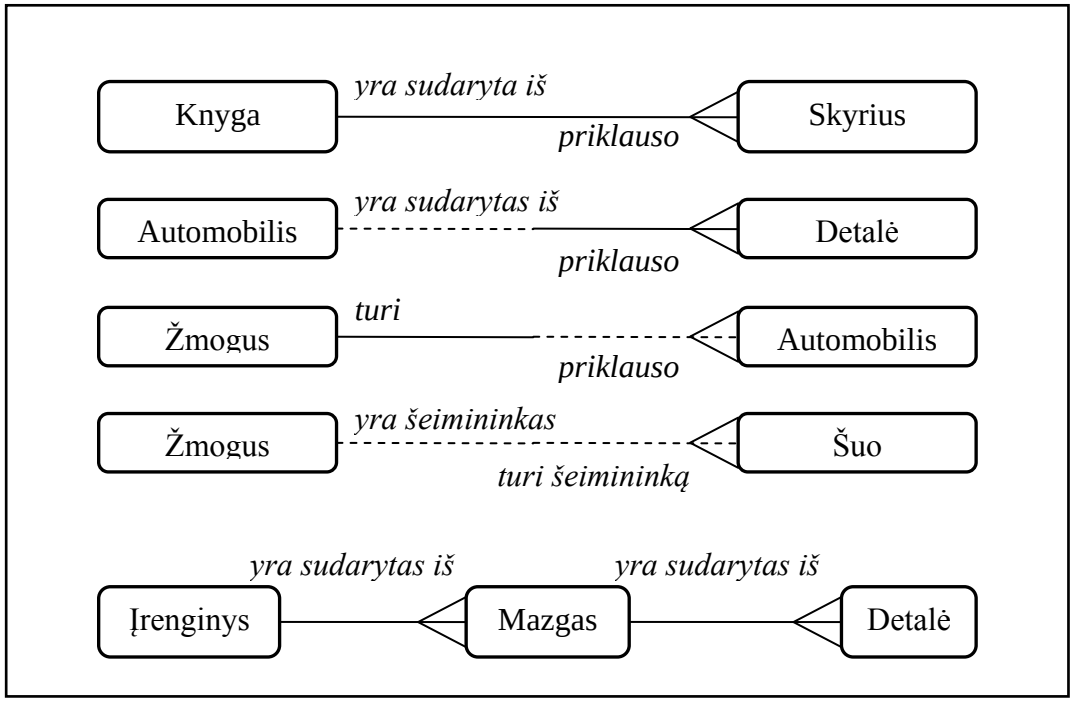
IV-5 pav. Ryšiai „vienas su vienu“.

Neprivalomas iš abiejų pusių ryšys neturi praktinės prasmės (pavyzdžiui, santuokos ryšys tarp moters ir vyro, jei laikysime, kad svarbi tik nenutraukta santuoka). Atsiradus tokiam ryšiui, dažnai tenka pasirinkti, kuri iš susietų esybių yra svarbesnė ir paversti ryšį privalomu bent iš vienos pusės. Tai padaroma įvedant tarpinę esybę (pavyzdžiui, santuokų registrą, žr. IV.5.4 skyriaus pavyzdį), arba susiaurinant dalykinę sritį (pavyzdžiui, nagrinėjant tik sutuoktinių poras) taip, kad būtų išsaugota natūrali ryšio logika.

Paveiksle parodytas dar vienas ypač dažnas ryšio „vienas su vienu“ atvejis. Jis aprašo esybių *klasifikaciją* ir skaitomas, pavyzdžiui, taip : „Transporto priemonė gali būti automobilis. Automobilis yra transporto priemonė“. Šiuo atveju *Transporto priemonė* yra apibendrinanti klasė, o *Automobilis, Motociklas, Troleibusas* – jos poklasiai (plačiau apie klasifikaciją – XII.1). Savu ruožtu, *Automobilis* gali būti lengvasis, krovininis ir kt., t.y., galima sudaryti klasifikaciniais ryšiais susietą esybių **hierarchiją** iš keleto lygmenų. Lankelis, jungiantis keletą tokių ryšių čia ir kitais atvejais reiškia, kad iš lankeliu apjungtų ryšių vienu metu galimas tik vienas.

IV.5.3 Ryšiai „daug su vienu“

Ryšys „daug su vienu“ naudojamas dažniausiai, išskyrus retesnį „neprivalomas su neprivalomu“ šio tipo ryšį, kuris dėl savo abipusio neprivalomumo yra nepakankamai apibrėžtas.

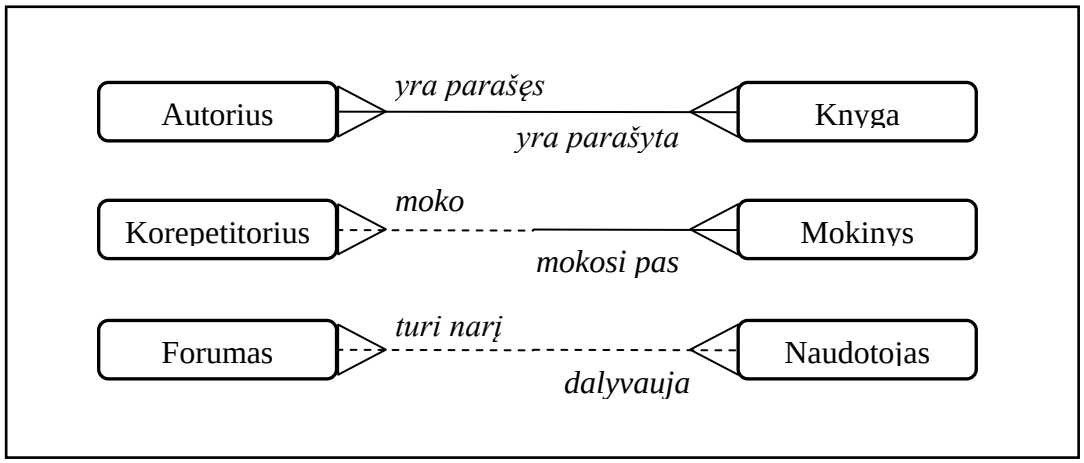


IV-6 pav. Ryšiai „daug su vienu“.

Paveiksle parodytas dar vienas dažnai pasitaikantis **hierarchinio** ryšio „daug su vienu“ tipas. Jis vadinamas **kompoziciniu ryšiu** ir reiškia, kad esybė yra kitos esybės sudedamoji dalis. Toks ryšys yra skaitomas, pavyzdžiui, taip : „Įrenginys yra sudarytas iš vieno arba daugiau mazgų. Mazgas yra vieno įrenginio dalis“. Priklausomai nuo komponentų būtinumo, kompoziciniai ryšiai gali turėti įvairias privalomumo reikšmes.

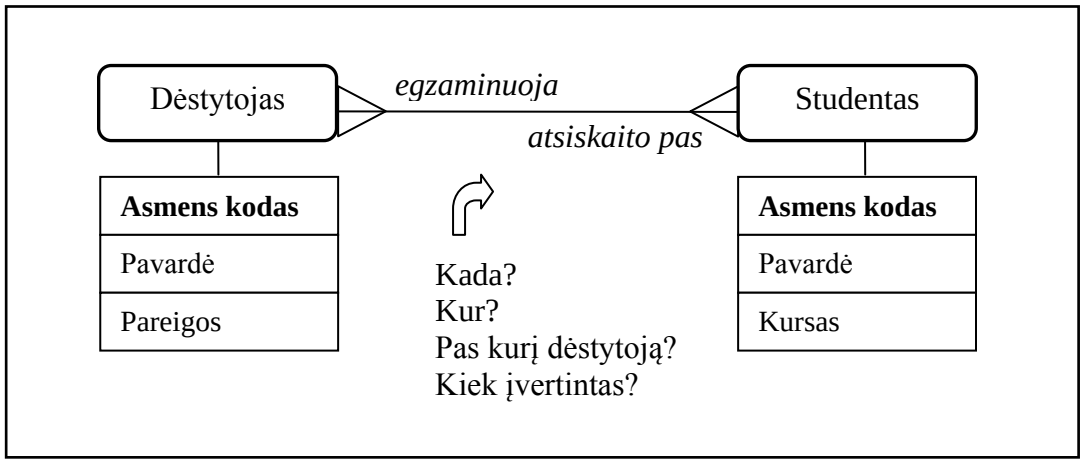
IV.5.4 Ryšiai „daug su daug“ ir ryšių skaidymas

Ryšys „daug su daug“ yra labai dažnas pradiniame modeliavimo etape ir tai nenuostabu, nes beveik visose dalykinėse srityse pirmiausia pastebimos įvairios daugiareikšmės sąsajos (IV-7 pav.).



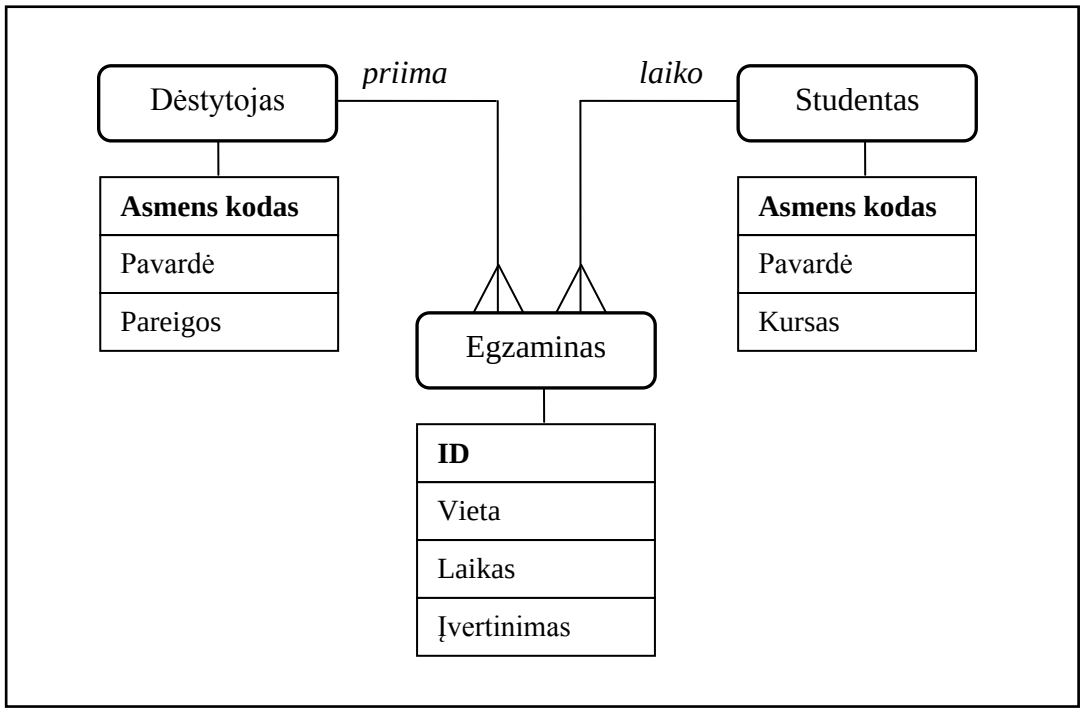
IV-7 pav. Ryšiai „daug su daug“.

Tokio ryšio pagrindinė problema visais atvejais yra ta, kad jo buvimas dar neleidžia sužinoti, su kuriais konkrečiais esybės egzemplioriais yra susietas esybės egzempliorius viename ryšio gale, taigi, praktiškai jis nėra labai naudingas, suteikia tik bendrą informaciją. Be to, kaip vėliau pamatysime, ryšių „daug su daug“ negalime atvaizduoti įprastu loginiu modeliu (lentelėse). Dar galima pastebėti, kad ryšiui „daug su daug“ būdinga turėti papildomas, ryšio savybes, kurios jį galėtų sukonkretinti. Tuo toks ryšys yra panašus į esybę su atributais.



IV-8 pav. Ryšio „daug su daug“ ypatumai.

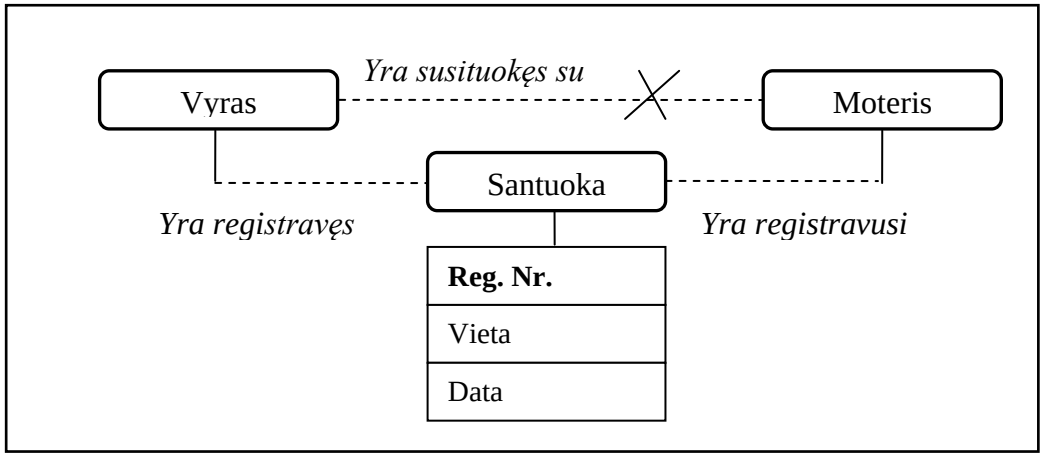
Ryšys „daug su daug“ visada turi būti išskaidytas į du įvedant tarpinę (dar vadinama ryšio arba sankirtos) esybę. Kartais tokia tarpinė esybė iš tikrųjų egzistuoja, dar dažniau ji sukuriama specialiai vaizduoti „daug su daug“ tipo ryšiui ir jo savybėms. Tokio skaidymo schema visada yra vienoda, o tarpinės esybės pavyzdys parodytas IV-9 pav.



IV-9 pav. Ryšio „daug su daug“ skaidymas į du.

Egzaminas šiuo atveju yra neapčiuopiama, bet realiai gerai suvokiama tarpinė esybė, kuri per egzaminavimo faktą susieja tarpusavyje vieną dėstytoją su vienu studentu ir tokiu būdu pakeičia IV-8 paveiksle pavaizduotą ryšį, be to, leidžia saugoti reikiamą atributinę informaciją. Buvusio ryšio „daug su daug“ abi pusės sujungtos su šia tarpine esybe, o kardinalumas “daug“ perkeliamas prie jos. Jei kuri nors „daug su daug“ ryšio pusė buvo neprivaloma, ši savybė išlieka (tarpinės esybės pusėje) ir skaidant.

Analogiškai skaidant galima išspręsti ir kitokio kardinalumo **iš abiejų pusių neprivalomus** ryšius (IV-10 pav.). Šiuo atveju apibrėžtumą įveda du ryšiai, kurie abu yra iš vienos pusės privalomi – nors ir vyras, ir moteris neprivalo būti susituokę, santuoka visada leidžia identifikuoti sutuoktinius, kitaip ji neegzistuotų. Be to, kaip ir ankstesniu atveju, atsiranda galimybė saugoti papildomą informaciją apie ryšį esybės *Santuoka* atributuose.

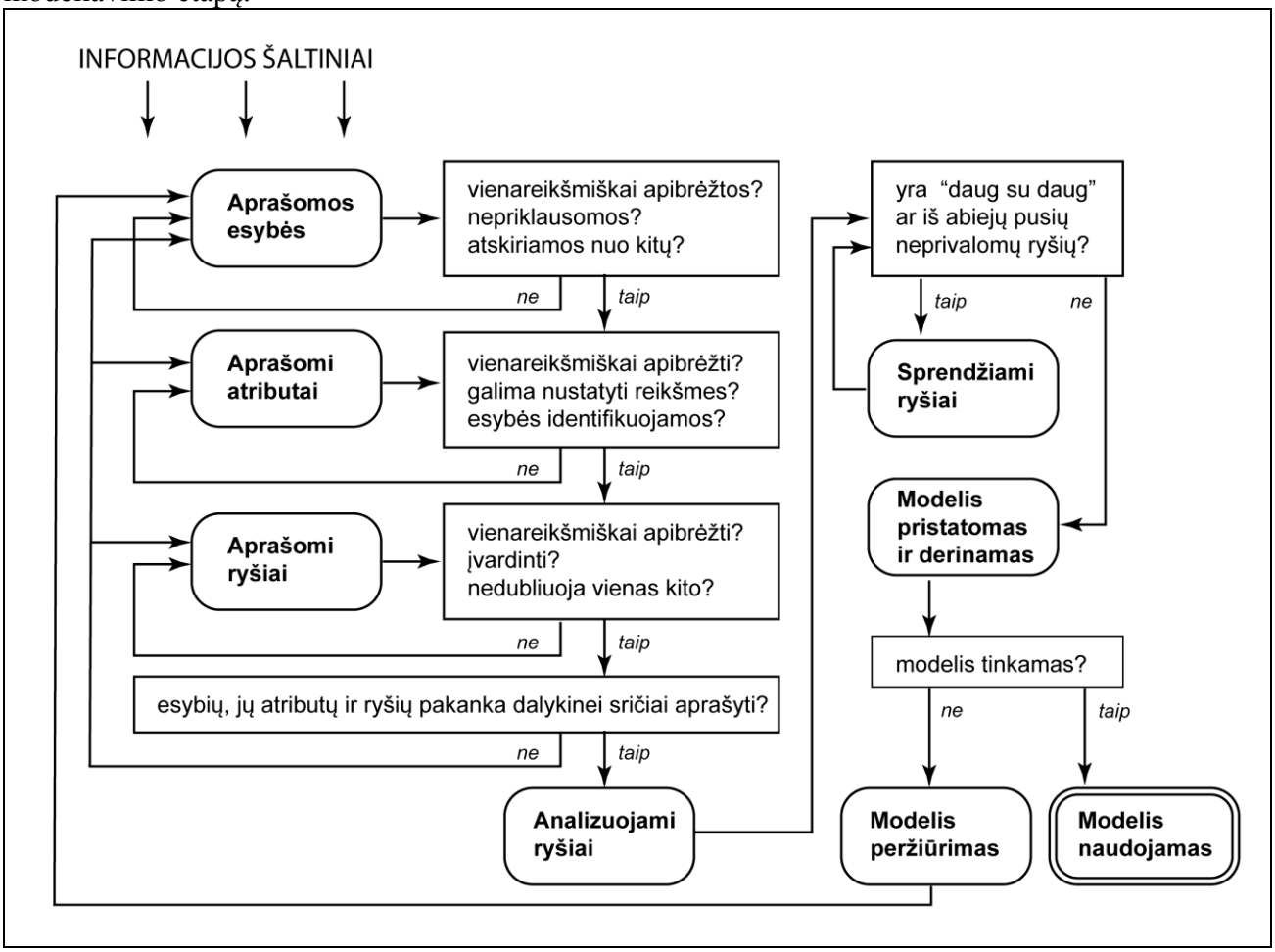


IV-10 pav. Neapibrėžto ryšio „vienas su vienu“ skaidymas į du.

Kaip pastebėjome, formaliai yra galimi įvairių tipų ryšiai, tačiau ne visus juos tikslinga naudoti koncepciniame modelyje.

IV.6 KONCEPCINIO MODELIO SUDARYMO SCHEMA IR VAIZDAVIMAS LOGINIU MODELIU

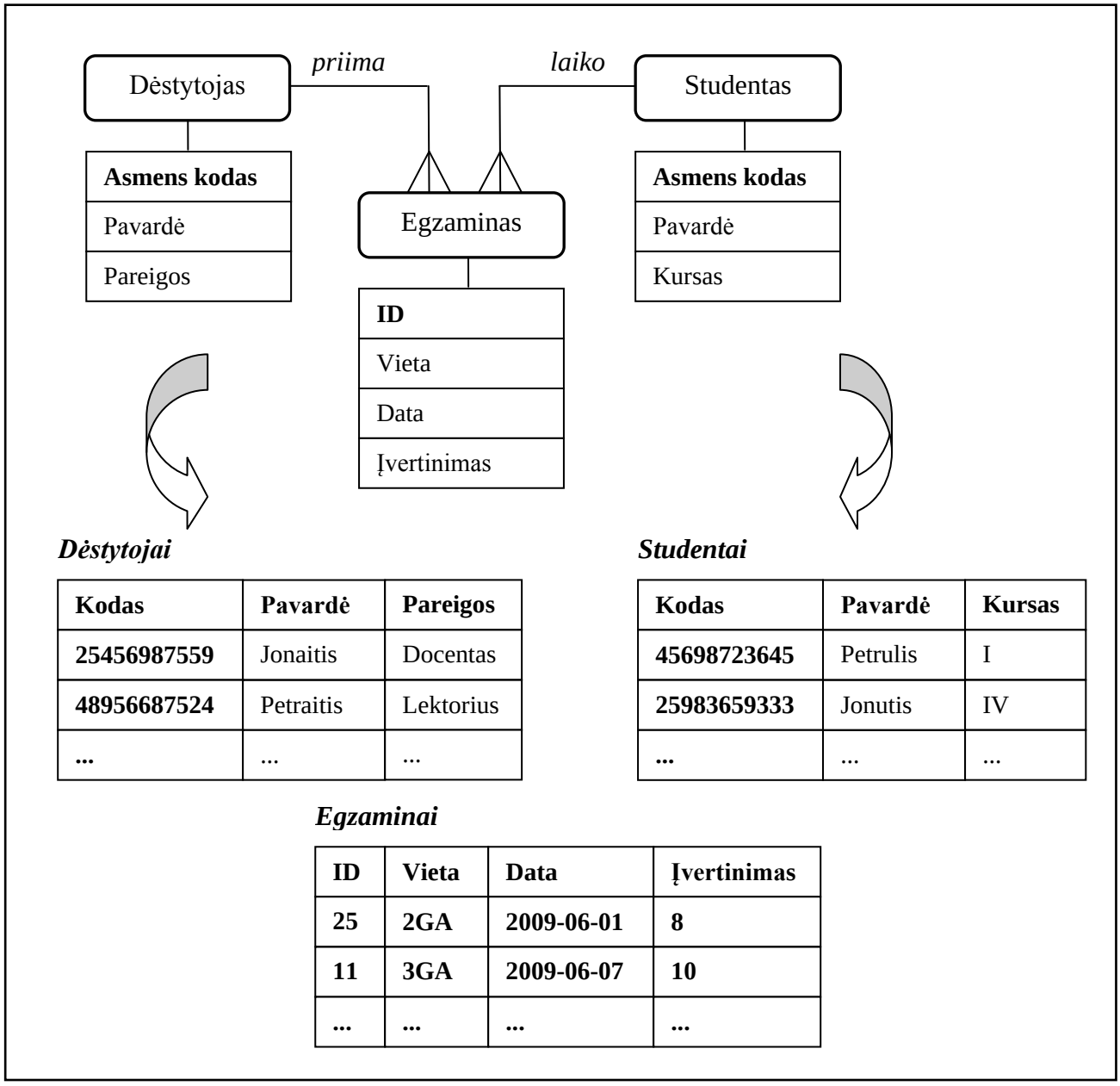
Apibendrinta koncepcinio modelio sudarymo schema parodyta paveiksle žemiau (IV-11 pav.). Jei parengtas geras koncepcinis modelis, jis yra efektyvi dalykinės srities pristatymo priemonė. Iš kitos pusės, pristatant ir aptariant modelį galima pastebėti jo trūkumus ir nesunkiai juos pataisyti. Sekantis žingsnis projektuojant duomenų bazę yra *loginio modelio* sudarymas. Reikia pastebėti, kad tvarkingo esybių ryšių modelio informacija perkeliama į dažniausiai naudojamas reliacines struktūras[1] visiškai vienareikšmiškai, laikantis paprastų ir aiškių taisyklių, kurios praktiškai neturi išimčių. Šis procesas gali būti automatizuotas – žinoma nemažai programų, kurios nubraižytas koncepcinio modelio diagramas paverčia jį atitinkančia duomenų bazės struktūra. Dar daugiau, teisingas koncepcinis modelis leidžia tikėtis, kad iš jo gauta duomenų bazės struktūra bus efektyvi ir nebereiks papildomo jos tvarkymo, tokio kaip duomenų bazės norminimas (aprašytas IX skyriuje). Tai labai retai pasiseka, jei lentelės kuriamos iš karto, praleidžiant ar supaprastinant koncepcinio modeliavimo etapą.



IV-11 pav. Koncepcinio modelio sudarymo schema

[1] Jas išsamiai aptarsime tolesniuose knygos skyriuose, tuo tarpu užtenka žinoti, kad tai yra lentelių rinkiniai.

Sudarant loginį modelį koncepcinis modelis atvaizduojamas į pasirinktos duomenų bazės struktūrą. Šiame etape duomenų prasmė ir galima interpretacija jau nebėra svarbi, o siekiama optimalaus jų struktūrizavimo, kad būtų galima lengvai rasti norimus duomenis, juos tarpusavyje susieti, be to, ir aptikti galimas klaidas, kurios neišvengiamai atsiranda duomenis įvedant. Pavyzdžiui, IV-9 paveiksle aprašytas koncepcinis modelis gali būti transformuotas į lentelių duomenų bazės struktūrą sukuriant tris lenteles, kuriose būtų aprašyti studentai, dėstytojai, egzaminai ir du juos jungiantys ryšiai.



IV-12 pav. Esybių ir atributų perkėlimas į loginį lentelių modelį.

Kaip matyti iš IV-11 paveikslo, koncepcinio modelio esybė vaizduojama duomenų bazės lentele, o esybės atributai tampa lentelės stulpeliais. Lentelės eilutėse rašomos konkrečios esybės egzempliorių atributų reikšmės.

Reliaciniame modelyje yra paprasta sukurti ir reikiamus ryšius. Susietos ryšio "daug" pusėje esybės lentelė tiesiog papildoma stulpeliais, kuriuose rašomos su ja susietų esybių egzempliorių unikalių identifikatorių reikšmės (IV-12 pav.). Taip galima rasti sąsajas tarp konkrečių egzempliorių skirtingose duomenų bazės lentelėse.

Dėstytojai

Kodas	Pavardė	Pareigos
25456987559	Jonaitis	Docentas
48956687524	Petraitis	Lektorius
...	...	...

Studentai

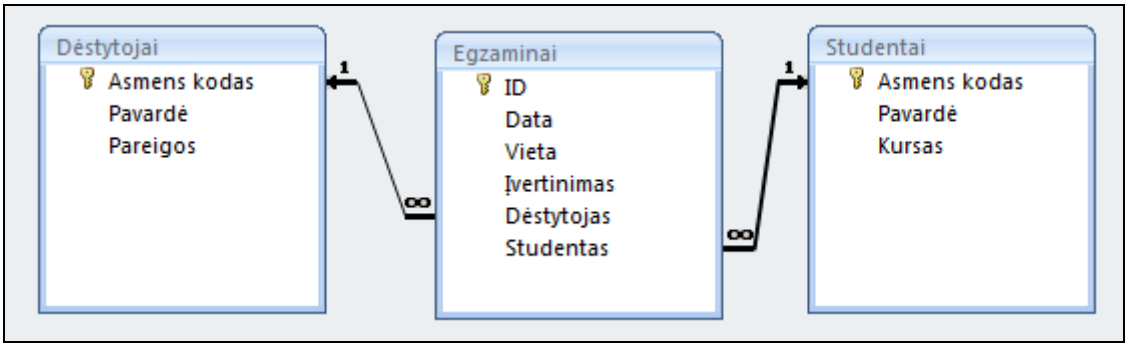
Kodas	Pavardė	Kursas
45698723645	Petrulis	I
25983659333	Jonutis	IV
...	...	...

Egzaminai

ID	Vieta	Data	Įvertinimas	Dėstytojas	Studentas
25	**2GA**	**2009-06-01**	**8**	**25456987559**	**25983659333**
11	**3GA**	**2009-06-07**	**10**	**48956687524**	**25983659333**
...	**...**	**...**	**...**	**...**	**...**

IV-13 pav. Ryšių vaizdavimas lentelių modelyje.

Jei yra sukurtos visos reikiamos lentelės, tokį loginį modelį galima pamatyti DBVS ryšių diagramoje (IV-14 pav.).



IV-14 pav. Duomenų bazės ryšių vaizdavimas *MS Access* ryšių diagrama.

Vaizduojant „vienas su vienu" tipo ryšį, gali kilti klausimas, kurioje lentelėje pridedamas papildomas stulpelis. Jei toks ryšys yra iš vienos pusės neprivalomas, tam, kad būtų efektyviau saugoma informacija, papildomas stulpelis visada pridedamas neprivalomoje ryšio pusėje. Jei ryšys simetriškas, reikia pasirinkti, kuri esybė yra svarbesnė modeliuojamos dalykinės srities požiūriu.

Klausimai diskusijai

Kodėl esybių vardai ER modelyje visada rašomi vienaskaita?
Ar studento vardas gali būti unikalus esybės identifikatorius konkrečioje studentų grupėje?
Ar galima sakyti, kad Marius turi atributą "automobilio numeris", o Darius šio atributo neturi, jei žinoma, kad Darius neturi automobilio?
Ar žmogaus turimas vaikų skaičius gali būti jo atributas?
Ar gali esybė neturėti nė vieno atributo? Turėti vienintelį atributą?
Ką reiškia sąvoka "skirtingos esybės"?
Ką reiškia faktas, kad dviejų skirtingų esybių keletas atributų sutampa?
Ar gali egistuoti ryšys tarp esybės ir jos pačios? Jei taip, pateikite pavyzdį. Jei ne, paaiškinkite, kodėl.
Sugalvokite, kokie atributai gali vienareikšmiškai nusakyti šių esybių egzempliorius: laikraščio, bet kurio planetos žmogaus, medžio, viruso.

Užduotys savarankiškam darbui

Sudarykite esybių ryšių modelius, vaizduojančius aprašytas situacijas.
1. Studentai per sesiją atsiskaito už įvairius dalykus skirtingiems dėstytojams.
2. Parduodant sandėlyje esančias prekes pirkėjams išrašomos sąskaitos faktūros.
3. Leidykloje leidžiami leidiniai parduodami skirtinguose knygynuose už sutartinę kainą.
4. Iš Vilniaus oro uosto keleiviai be tarpinių nusileidimų skrenda į įvairius miestus skirtingais lėktuvais, pilotuojamais skirtingų lakūnų.
5. Upės gali įtekėti į jūrą, kitą upę arba į vieną iš ežerų, ištekėti iš ežero, pelkės, arba prasidėti šaltiniu.
Išspręskite modeliuose atsiradusius „daug su daug" tipo ryšius.

Užduotys praktikos darbams

Susipažinkite su Valstybės informacinių išteklių valdymo įstatymo ar kito teisės akto tekstu. Sudarykite pradinį koncepcinį jo reglamentuojamos srities modelį.

Išnagrinėkite ryšių vaizdavimą *MS Access* duomenų bazių valdymo sistemoje. Sukurkite ryšius tarp esamų lentelių. Įsitikinkite, kad keičiant duomenis, jiems taikomos vientisumo taisyklės, kurias nustatėte apibrėždami ryšį. Vieną iš savarankiškai sudarytų esybių ryšių modelių atvaizduokite lentelėmis duomenų bazėje, sukurkite ryšius, įveskite duomenis. Įsitikinkite, kad ryšiai buvo suprojektuoti teisingai. Nurodykite atributų tipus, sukurkite reikiamus jų įvedimo šablonus ir patvirtinimo taisykles.

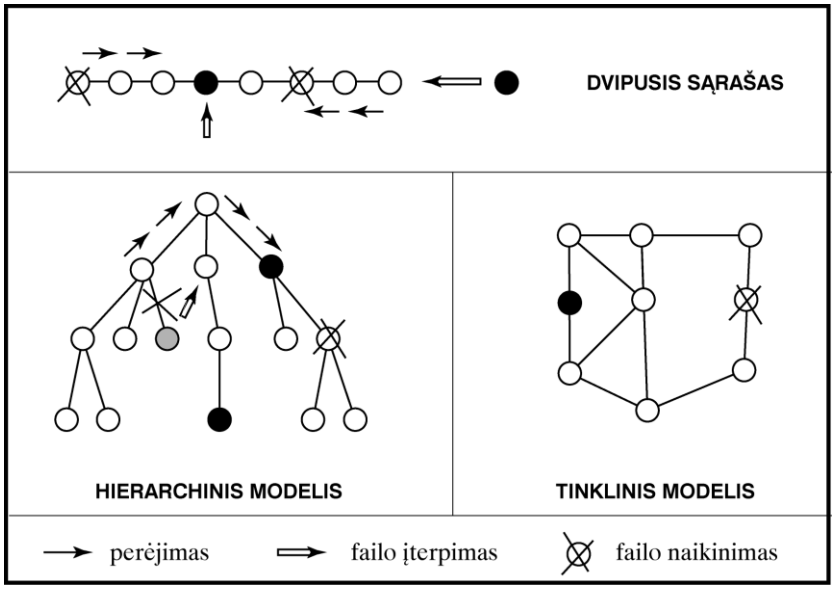
V. PAGRINDINIAI DUOMENŲ BAZIŲ MODELIAI

Sudėtinga ir blogai veikianti sistema dažnai būna išvystyta iš daug paprastesnės be priekaištų veikusios sistemos.
Merfio dėsnis

Ankstesniuose skyriuose išsiaiškinome, kad duomenų bazėje saugoma struktūrizuota informacija apie dalykinės srities esybes. Kiekviena organizacija visada turėjo vienu ar kitu būdu saugomą duomenų sistemą, kurią sudarė faktų apie išskirtas esybes rinkiniai. Tokius duomenų rinkinius kol kas vadinsime failais. Failai nekompiuterizuotose sistemose galėjo būti ir dokumentų segtuvai, knygos, žemėlapiai ar kitokio pavidalo informacija, iš dalies sutvarkyta, pavyzdžiui, segtuvai atskirose lentynose, sunumeruoti ir pan. Būdas, panaudotas failams sutvarkyti, vadinamas **duomenų bazės modeliu**.

Duomenų bazės failai gali būti sutvarkyti pagal vieną iš žinomų modelių.

1. Dvipusis sąrašas
2. Hierarchinis
3. Tinklinis (orientuotų grafų)
4. Reliacinis
5. Objektinis
6. Loginis



V-1 pav. Sąrašo, hierarchinis ir tinklinis duomenų bazės modeliai

V.1 DVIPUSIS SĄRAŠAS

Sąrašas yra paprasčiausias iš kompiuterizuotų duomenų modelių. Jame turi būti įvykdomos duomenų struktūrų pridėjimo ir atėmimo iš sąrašo galų, vėliau – įterpimo ir išmetimo iš sąrašo vidurio, dviejų sąrašų sujungimo operacijos. Kiekvienas dvipusio sąrašo elementas (failas) turi tik dvi nuorodas: į prieš jį ir po jo esančius elementus. Todėl skaityti duomenis iš sąrašo ir atlikti paiešką galima tik nuosekliai pereinant nuo vieno failo prie kito. Aišku, kad tokia struktūra nepatogi,

teikia mažai operacinių galimybių ir yra neefektyvi. Ją galima įsivaizduoti, kaip dokumentų segtuvus iš eilės sudėtus lentynoje, arba Interneto puslapių seką be hipertekstinių nuorodų, kai galima tik grįžti atgal naudojant naršyklės „Back“ mygtuką. Norint surasti norimą puslapį, reikia visus puslapius nuosekliai peržiūrėti. Taigi, paieškos dvipusiame sąraše su n elementų sudėtingumas (elementarių operacijų skaičius, reikalingas rezultatui pasiekti blogiausiu atveju) yra n.

V.2 HIERARCHINIS MODELIS

Tai sekantis žingsnis tobulinant sąrašo modelį, kai sudaroma galimybė atspindėti „vienas su daug“ tipo ryšius tarp esybių. Nuorodos iš failų sudaro hierarchinę struktūrą – medį. Tokioje struktūroje atsispindi klasių hierarchija, kurią sudarant turi būti laikomasi klasifikacijos taisyklių. Pagal hierarchinį modelį galima kur kas geriau struktūrizuoti duomenis. Paieškos dvejetainiame medyje su n viršūnių sudėtingumas yra $\log_2 n$, taigi, ji, lyginant su sąrašo modeliu, atliekama daug kartų greičiau. Hierarchiniame modelyje galimos visos sąrašo operacijos, galima įterpti ir naikinti failą bet kuriame lygmenyje. Be to, galima pridėti ar išmesti visą šaką, pakeisti priklausomybę (nuorodas į tėvą ir vaikus), pakeisti failo lygmenį hierarchijoje ir kt. Hierarchinį modelį praktiškai galime stebėti Interneto svetainėse, kurių puslapiuose yra išėjimai per hipertekstinę nuorodą į žemesnio rango puslapius, iš kurių arba turi būti grįžtama į pradinį puslapį, arba patekti į dar žemesnio rango puslapius.

Pagrindinis hierarchinio modelio trūkumas yra tas, kad ryšius galima kurti tik vertikaliai, o ne horizontaliai ar įstrižai. Tai reiškia, kad nėra ryšio tarp to paties lygio elementų, nebent jie priklauso tam pačiam pirminiam įrašui. Todėl daugelį tokios duomenų bazės failų tenka pakartoti.

V.3 TINKLINIS (ORIENTUOTŲ GRAFŲ) MODELIS

Tai yra patobulintas hierarchinis modelis, kuris ypatingas tuo, kad galimos nuorodos ne tik tarp tėvų ir vaikų, bet ir to paties lygmens failų, o dar vėliau – tarp visų sistemos failų apskritai. Gali būti vaizduojami ir „daug su daug“ ryšiai, t.y., skirtingai nuo hierarchinio modelio, tinkliniame modelyje failas gali turėti daugiau nei vieną tėvą. Tinkliniu modeliu galima aprašyti sudėtingesnius ryšius ir realizuoti kelias hierarchijas vienu metu. Be to, kiekvienai atitinkamo grafo briaunai gali būti nurodyta jos orientacija: apribojamas perėjimas į vieną kurią nors pusę. Tinkliniu modeliu pagrįstos tokios Interneto svetainės, kurių puslapiuose yra ne tik nuorodos į prieš tai buvusius ir po jų sekančius, bet ir gretimus tos pačios temos puslapius (brolius).

Didžiausias tinklinio modelio trūkumas yra sistemos sudėtingumas, nes egzistuoja daug duomenų bazės elementų ryšių, kuriuos naudotojas turi suprasti, norėdamas pasiekti failus ir atlikti su jais operacijas. Reikia atkreipti dėmesį, kad tiek hierarchinis, tiek tinklinis modeliai nepasižymi *struktūriniu nepriklausomumu*, t.y., veiksmų atlikimo su duomenimis būdas labai priklauso nuo konkrečios duomenų bazės struktūros. Naudojant hierarchinį ar tinklinį modelį ir pakeitus duomenų saugojimo būdą, tenka keisti visas taikomąsias programas, kurios naudoja hierarchinę ar tinklinę duomenų bazę. Todėl duomenų bazės ir programų valdymas gali būti labai sudėtingas ir pareikalauti daug laiko.

Pirmieji trys modeliai yra išsivystę vienas iš kito ir iš esmės panašūs. Visiškai nauja koncepcija pagrįstas yra reliacinis modelis, pradėjęs plisti nuo maždaug 1970 metų. Jis praktiškai išstūmė tinklinį modelį, perimdamas visas jo galimybes ir šiuo metu vis dar yra populiariausias.

V.4 RELIACINIS MODELIS

Šiuo metu tai plačiausiai naudojamas modelis, pagrįstas matematine reliacinės algebros teorija. Reliacinis modelis ypatingas tuo, kad esybių duomenys jame pateikiami lentelėmis ir atliekamos specifinės operacijomis su lentelėmis. Reliacinių duomenų bazių sistemų, kurias mes toliau nagrinėsime, loginiame modelyje esybės vaizduojamos lentelėmis, jų atributai tampa lentelės stulpeliais, o lentelės eilutės aprašo tos esybės egzempliorius, kurie skiriasi vieni nuo kitų savo atributų reikšmių rinkiniais. Ryšiai reliacinėse sistemose suformuojami naudojant esybių atributų reikšmes. Tai reiškia, kad tam tikri ryšiais susietų esybių atributai tampa pirminiais ir išoriniais raktais, kuriuos nagrinėsime tolesniuose skyriuose. Jei visos esybės modelyje yra susietos prasmingais ryšiais, turime pakankamai informacijos duomenų bazei sukurti ir pildyti. Reliacinis modelis pasižymi *struktūriniu nepriklausomumu*, t.y., galima keisti duomenų struktūrą nekeičiant programinės įrangos ar duomenų apdorojimo procedūrų.

Sąlyginis reliacinio modelio paprastumas turi savo kainą. Vienas iš pagrindinių reliacinio modelio trūkumų yra didelis naudojamos techninės ir programinės įrangos sudėtingumas. Tiesą sakant, kai 1970 m. E. Kodas pasiūlė reliacinį modelį, tuometiniai kompiuteriai pasirodė nepakankamai galingi jam realizuoti, o kai kurie modelio aspektai ir šiandien lieka tik teoriniai.

Dauguma reiškinių gali būti nusakyti baigtinėmis (t.y., apibrėžto dydžio) charakteristikomis, pavyzdžiui, žmogaus pavardė vargu ar gali viršyti 100 simbolių, amžius 999 metus – taigi, pavardei saugoti pakaks skirti 100 pozicijų tekstinėje eilutėje, o amžiui – triženklį sveiką skaičių. Tačiau erdviniams duomenims ne visada patogu nurodyti maksimalų dydį, pavyzdžiui, vektorinę liniją gali sudaryti du arba du milijonai mazgų. Reliacinės sistemos, nors ir labai efektyvios, turi vieną trūkumą – jose laukų dydžiai yra fiksuoti ir nurodomi naudotojo, todėl kiekvieną erdvinį objektą tenka sugoti kaip lentelę, arba juos apjungti vienoje lentelėje, kas prieštarauja koncepcinio modelio sudarymo principams ir labai apsunkina darbą su objekto duomenimis. Todėl, pavyzdžiui, erdvinius duomenis patogiau saugoti dar kito tipo sistemose. Tokios sistemos su išplėstomis galimybėmis yra objektinės-reliacinės duomenų bazių valdymo sistemos. Pagrindiniai RDBVS gamintojai (*Oracle, IBM, Microsoft*) įvairiu lygmeniu palaiko objektines technologijas.

Georeliacinis modelis

Nors šiuolaikinėse GIS sistemose visų duomenų tvarkymui naudojamos reliacinės DBVS (pavyzdžiui, tokia yra *ESRI* geoduomenų bazės), anksčiau buvo įprasta erdvinius ir atributinius geografinių objektų klasių duomenis saugoti skirtingose struktūrose. Erdviniai duomenys (taškai, linijos ir poligonai) buvo saugomi specialiu dvejetainiu formatu, o neerdviniai atributai (pavyzdžiui, kelio plotis ar dangos tipas) – standartiniu RDBVS formatu, tokiu kaip *dBASE* ar *INFO*. Tada ryšį tarp šių dviejų failų struktūrų palaikydavo taikomosios programos, užtikrindamos, kad erdviniai objektai tinkamai susiejami su jų neerdviniais atributais. Tokia duomenų bazė, kurioje duomenys skirstomi į dviejų skirtingų tipų struktūras, vadinama georeliacine arba mišrios architektūros DB. Nors tai nėra savarankiškas ar efektyvus duomenų modelis, jis dar naudojamas nemažai GIS sistemų, pavyzdžiui *Arc/INFO, MapInfo, ArcView* ir *MGE*.

V.5 OBJEKTINIS MODELIS

Tai nuo 1990 metų sparčiai besivystantis modelis, diegiamas naujose duomenų bazių valdymo sistemose. Jame visi duomenys vaizduojami objektais, taigi, dar labiau priartėjama prie „žmogiškos“

modeliuojamų esybių interpretacijos. Pagrindinės šio modelio sąvokos yra objektas, klasė, atributas, paveldimumas, metodas, įvykis ir pranešimas.

Objektai – tai vienareikšmiškai identifikuojamos esybės, apie kurias saugoma informacija, pavyzdžiui, „vandens telkinys“, „gyventojas“, „žemėlapis“ – jos pagal prasmę niekuo nesiskiria nuo esybių reliaciniame modelyje. Tačiau objektai nevaizduojami lentelėmis, o egzistuoja sistemoje kaip abstrakcijos ir skirtingų jų egzempliorių rinkiniai. Objekto *atributas*, kaip ir reliaciniame modelyje, yra išmatuojama jo savybė. To paties objekto egzemplioriai skiriasi atributų reikšmėmis. Kiekvienas objektas turi priklausyti kuriai nors *klasei*. Kad klasių hierarchija neliktų atvira ir būtų galima ją realizuoti kompiuterinėse sistemose, įvedama *metaklasės* sąvoka, kaip superklasės, kuriai priklauso bet kuri kita klasė, o taip pat ir ji pati. Objektas, priklausantis klasei, *paveldi* visus tos klasės atributus, t.y., jei „vandens telkinys“ turi atributą „vidutinis gylis“, tai ir jo subklasė „ežeras“ jį paveldi, todėl kuriant objektą „ežeras“ nebereikia nurodyti šio atributo.

Metodas – tai nauja sąvoka, kuria išreiškiama funkcija, kurią objektas „moka“ įvykdyti. Visos operacijos su objektais atliekamos inicijuojant jų metodus, pavyzdžiui, „judėti“, „susinaikinti“. Metodai taip pat yra paveldimi iš superklasės, nors ir gali būti interpretuojami skirtingai, pavyzdžiui „maitintis“ yra skirtingi metodai kiškiui ir vilkui ekosistemos modelyje, nors jie vadinasi vienodai ir priklauso „gyvūno“ superklasei.

Inkapsuliacija – tai terminas, nusakantis objekto izoliuotumą nuo jo aplinkos: niekas, išskyrus patį objektą, negali keisti jo vidinės struktūros (tą galime lengvai padaryti su lentelėmis, pavyzdžiui sunaikinti eilutę ar stulpelį, sujungti dvi panašias lenteles, išrinkti stulpelių poaibį ir pan.) ar atlikti su juo kitų manipuliacijų. Objektai gali būti keičiami tik per iš anksto užprogramuotus jų metodus, o keičiant struktūrą reikia sukurti naują objektą.

Įvykis – tai sistemos pertraukimas, susidarius tam tikroms aplinkybėms, pavyzdžiui, naudotojo veiksmas, elektros srovės dingimas, signalas iš išorinio įrenginio. Objektai, egzistuojantys sistemoje reaguoja į įvykius, atlikdami kurį nors savo metodą. Objektai tarpusavyje gali keistis *pranešimais*. Gautas pranešimas tam tikra prasme yra įvykis.

Atskiras objektų atvejis yra agentai – programinės įrangos komponentai, veikiantys kartu, siekdami konkretaus tikslo pagal nustatytus apribojimus. Geografinės informacijos sistemose jie gali atlikti, pavyzdžiui, generalizavimo funkciją. Geografiniai objektai (pastatai, keliai ir pan.) tampa aktyviais agentais ir dirba kartu atlikdami generalizavimo operacijas, tam, kad automatiniu būdu būtų gautas priimtinas rezultatas.

Objektinis modelis leidžia operuoti dideliais dokumentų objektais (tekstiniais dokumentais, multimedia failais), kurie gali turėti skirtingą vidinę struktūrą, be to, naudotojui suteikiama galimybė apibrėžti savo duomenų tipus. Tai akivaizdus objektinių technologijų privalumas teminėje kartografijoje.

Kaip ir reliacinis modelis, objektinis modelis išsaugo struktūrinį vientisumą, todėl galima keisti duomenų struktūrą nekeičiant programinės įrangos ar nuskaitymo procedūrų. Didžiausias objektinio modelių trūkumas yra standartų, tokių kaip SQL reliacinėse sistemose, nebuvimas. Objektinių struktūrų duomenų prieigos metodai gali būti sudėtingi ir tuo panašūs į hierarchinių arba tinklo modelių duomenų prieigos metodus. Todėl daugelis operacijų vykdomos daug lėčiau negu reliacinėse struktūrose.

V.6 DEDUKCINIS (LOGINIS) MODELIS

Dedukcinės duomenų bazių sistemos yra palyginti siauros paskirties sistemos, kurios leidžia daryti išvadas (dedukciją) iš sukauptų faktų, naudojant apibrėžtas taisykles, ir taip papildyti faktų aibę.

Operacijos, atliekamos su duomenimis, yra predikatų skaičiavimas. Šiai grupei galima priskirti skirtingai vadinamas duomenų bazių sistemas (ekspertinės, dedukcinės, semantinės, inferencinės, „plečiamos“ ir kt.) Jose siekiama efektyviai saugoti žinias, kurių pagalba dirbama su duomenimis. Taisyklių rinkiniai gali būti kad ir tokie: TĖVAS (A, B), TĖVAS (B,C), o išvedimo taisyklė:

TĖVAS (A, B) AND TĖVAS (B,C) => PALIKUONIS (C, A).

Išvedimo taisyklėmis tokiose sistemose grindžiami įrodymai. Tai savotiškas reliacinių duomenų ir loginio programavimo priemonių junginys, teoriškai įdomus ir pasižymintis kai kurioais privalumais, tačiau taip ir nepaplitęs praktikoje. Šiuo metu tokiose duomenų bazėse faktams, taisyklėms ir užklausoms aprašyti paprastai naudojama *Datalog* kalba.

Dabar duomenų bazės sąvoka yra taikoma tik kompiuterizuotoms sistemoms.

🕮 Klausimai diskusijai

Kaip manote, kuo elektroninės lentelės yra panašios į duomenų bazę? Kuo failų katalogas panašus į DBVS? Kokios Jūsų naudojamos sistemos (nebūtinai kompiuterinės) savo organizavimo priemonėmis primena objektinį modelį ir kuo?

✍ Užduotys savarankiškam darbui

Sudaryti kartografinio projekto koncepcinį duomenų modelį, apimantį esybes, atributus, domenus ir ryšius pagal šią informaciją.

1. Įmonėje sudaromi žemėlapiai.
2. Žemėlapis turi unikalų numerį, pavadinimą, formatą, mastelį ir nuorodą į jo dokumentą.
3. Žemėlapis priklauso vienam iš trijų skyrių: Gamta (G), Visuomenė (V), Istorija (I).
4. Žemėlapio dokumentas turi unikalų numerį, sutampantį su žemėlapio numeriu.
5. Žemėlapis turi vieną ar daugiau informacijos šaltinių.
6. Informacijos šaltinis turi unikalų inventorinį numerį, pavadinimą ir žymą apie jo būklę.
7. Įstaigoje dirba kartografai.
8. Kartografas turi asmens kodą ir pavardę bei gauna fiksuotą atlyginimą.
9. Kartografas yra atsakingas už vieną ar daugiau žemėlapių.
10. Už vieną žemėlapį atsakingas vienas ir tik vienas kartografas.

💻 Užduotys praktikos darbams

Sukurkite GIS duomenų bazę pagal koncepcinį duomenų modelį, aprašantį miestų ir valstybių atributus bei galimus ryšius.

VI. DUOMENŲ BAZIŲ VALDYMO SISTEMOS SAMPRATA IR FUNKCIJOS

Kuo toliau eini, tuo mažiau žinai.
Lao Zi

VI.1 DUOMENŲ BAZĖ IR DUOMENŲ BAZIŲ VALDYMO SISTEMA

Duomenų bazė (DB) – tai informacijos apie dominančias esybes rinkinių ir sąryšių visuma. Įrašu, arba failu vadinama tarpusavyje susijusių duomenų grupė. Elementarus duomenų vienetas vadinamas lauku. Toliau kalbėdami apie duomenų bazes ir jų valdymo sistemas, laikysime, kad jos yra tik skaitmeninės.

Duomenų bazių valdymo sistema (DBVS) paprastai įsivaizduojama kaip programinė įranga duomenų bazei valdyti ir aukšto lygio programavimo kalbos (t.y., kalbos, kuria nurodoma *ką* daryti, o ne *kaip* daryti) interpretatorius, leidžiantys sukurti kompiuterizuotą įrašų saugojimo sistemą. O plačiausia prasme duomenų bazių valdymo sistemą sudaro keturi komponentai: duomenys, aparatūra, jau minėta programinė įranga ir naudotojai. Ji skirta tenkinti *organizacijos* reikmėms. Idealiu atveju duomenų bazėje saugoma visa organizacijai reikalinga nuolatinė informacija, taigi, įdiegta skaitmeninė duomenų bazė ir ją palaikanti įranga tampa svarbia valdymo informacinės sistemos dalimi.

Būdingos duomenų bazei operacijos yra:

- Ieškoti įrašų pagal nurodytas sąlygas;
- Pridėti naujus tuščius įrašus;
- Įrašyti duomenis (reikšmes) į esamus įrašus;
- Atlikti paiešką;
- Keisti esamus įrašų duomenis;
- Šalinti esamus įrašų duomenis;
- Šalinti įrašus

> Užklausų pavyzdžiai žemiau užrašyti specialia SQL (angl. *Structured Query Language)* struktūrizuota užklausų kalba.
>
> SELECT Pavadinimas FROM Žemėlapiai
> WHERE SudarymoMetai > 1995;
>
> DELETE * FROM Žemėlapiai
> WHERE Tema = „Istorija“;
>
> UPDATE Žemėlapiai
> SET Autorius = „Petraitis“
> WHERE Autorius = „Nenurodytas“ AND Metai = 1999;

Duomenys.
Duomenų bazės gali veikti tiek didelėse mašinose, tiek asmeniniuose, ar net nešiojamuose kompiuteriuose. Dažniausiai asmeniniame kompiuteryje duomenų bazė skirta tik vienam naudotojui

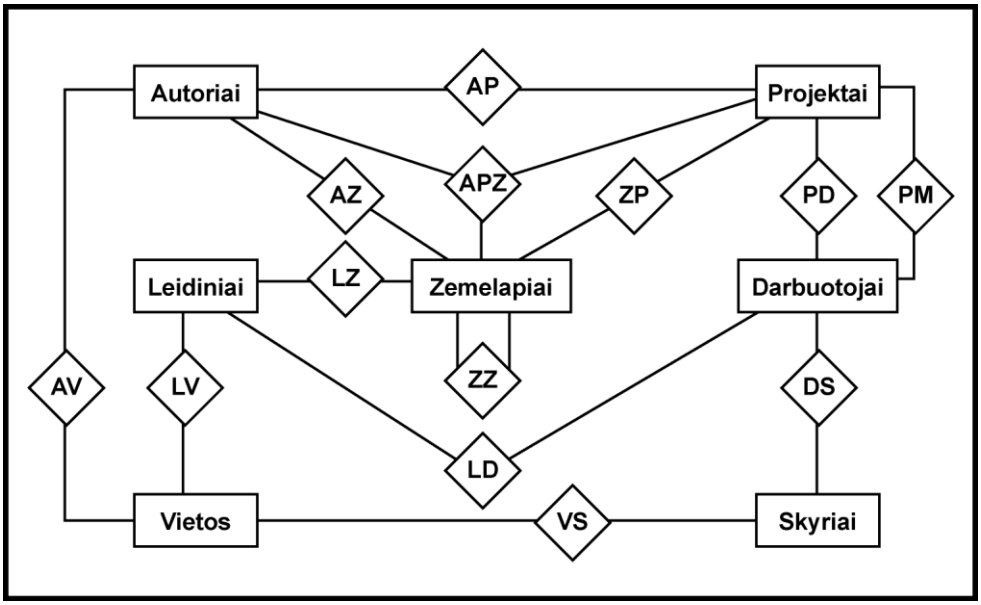
vienu metu. Dideles duomenų bazes naudoja daug žmonių vienu metu. Vieno ir daugelio naudotojų duomenų bazės skiriasi savo vidine struktūra. Apskritai laikoma, kad visi duomenys organizacijoje priklauso vienintelei duomenų bazei, nors praktiškai duomenų bazių gali būti daug. Duomenys bazėje yra bendro naudojimo, t.y., prieinami visiems, ir integruoti kaip tarpusavyje nepersidengiančių objektų prasmingas junginys. Ši savybė leidžia kiekvienam atskiram naudotojui peržiūrėti ir naudoti tik tam tikrą duomenų dalį, t.y., tą pačią duomenų bazę skirtingi naudotojai gali suvokti ir naudoti skirtingai.

Duomenys, sudarantys duomenų bazę, laikomi nuolatiniais. Be jų, dar yra tranzitiniai duomenys – įvedami, išvedami duomenys ir tarpiniai skaičiavimų rezultatai. Jie visada gali būti paversti nuolatiniais.

Aparatūros pagrindiniai komponentai yra laikmenos, procesoriai ir įvairūs išoriniai įrenginiai.
Programinis komponentas – tai DBVS ir programavimo kalba, be to, dar yra ataskaitų generatoriai, naudotojo sąsajos priemonės ir kt.
Naudotojai gali būti trijų tipų:

1. Galutiniai naudotojai, dirbantys interaktyvaus terminalo režimu. Tai didžiausia naudotojų grupė, kuria mažai rūpi loginiai duomenų modeliai ar DBVS technologija. Naudotojams gali būti suteikiamos skirtingos teisės atlikti veiksmu su DB objektais. DBVS turi garantuoti duomenų saugumą naudojant sinchroniškai. Naudotoju gali būti laikoma ir taikomoji programa, kuri kreipiasi į DBVS ir gauna iš jos duomenis, reikalingus kokiam nors uždaviniui spręsti. Žmogaus dalyvavimas tuo atveju nėra būtinas.
2. DBVS administratoriai – naudotojai, užtikrinantys tinkamą sistemos veikimą, duomenų saugumą ir efektyvų jų naudojimą.
3. Programuotojai (jie naudoja duomenis ir kuria taikomąsias programas, dirbančias su DB).

Objektai ir ryšiai.
Tarkime, verslo įmonė užsiima kartografija. Paprastai reikia saugoti informaciją apie vykdomus projektus, sudaromus žemėlapius, įmonės darbuotojus, žemėlapių autorius, leidinius, kuriuose panaudojami žemėlapiai ir pan. Visa tai yra objektai (tai, kas atvaizduojama DB). Be pagrindinių objektų, kaip ir įprastame esybių ryšių modelyje, dar egzistuoja ryšiai tarp objektų (VI-1 pav.).



VI-1 pav. Duomenų bazės objektų ir ryšių schema

Ryšiai, kaip ir esybių ryšių modelyje yra dvipusiai ir pasižymi tokiomis pačiomis savybėmis. Pavyzdžiui, „projektuose sudaromi žemėlapiai“ / „žemėlapiai sudaromi projektų metu“ (ryšys ZP). Remiantis šiuo ryšiu, galima, žinant projekto identifikatorių, surinkti informaciją apie visus projekto vykdymo metu sudarytus žemėlapius, o kiekvienam konkrečiam žemėlapiui nustatyti, kuriam projektui jis priklauso.
Ryšys (reliacinių duomenų bazių teorijoje dar vadinamas santykiu) taip pat yra duomenų dalis.
Keli momentai, į kuriuos reiktų atkreipti dėmesį schemoje:

1. Pavyzdyje yra trinaris santykis APZ. ER modeliuose tokių ryšių nebuvo, juos galima interpretuoti kaip išskaidytą ER ryšį „daug su daug“. Reikia pastebėti, kad trys atskiri ryšiai „A dirba P“, „A sudaro Z“ ir „Z sudaromas P“, dar nebūtinai pasako, kad konkretus autorius sudaro konkretų žemėlapį konkrečiam projektui (APZ). Taigi, trinaris ryšys yra informatyvesnis už paprastą jo komponentų aibę.
2. Yra unarinis santykis ZZ. Tai būdingas DB santykis, reiškiantis, kad vienas objektas gali būti tos pačios klasės objekto dalis. Mūsų atveju vieni žemėlapiai yra panaudojami kitiems sudaryti.
3. Tarp tų pačių objektų gali egzistuoti keli skirtingi ryšiai (PD – projekto vykdytojai, PM – projekto vadovai).

Tam tikra prasme ryšys taip pat yra objektas.

Atributai gali būti ir paprasti, pavyzdžiui, asmens kodas, ir labai sudėtingi, pavyzdžiui, „struktūra“. Tačiau laikysime, kad jie yra paprasti ir išreiškiami elementariais duomenų tipais, juo labiau, kad dauguma duomenų bazių valdymo sistemų ir dabar nėra pritaikytos sudėtingoms savybėms (grafikai, tekstui) vaizduoti. Labiausiai tam tinka objektinės duomenų bazės.

VI.2 KODĖL REIKALINGA DUOMENŲ BAZĖ?

Net ir vienam naudotojui yra patogu turėti skaitmeninę duomenų bazę (tai gali būti ir asmeninių muzikos įrašų katalogas, ir didelio restorano DB) dėl šių priežasčių.

1. **Kompaktiškumas**. Saugant duomenis skaitmeniniu pavidalu, jie užima mažai vietos. Nereikia saugoti popierių, kartotekų ir pan., taip taupomi darbo ir vietos resursai.
2. **Greitis**. Kompiuteris atlieka paieškos ir kitas operacijas daug greičiau, negu žmogus. Pavyzdžiui, atsakyti, kas sudarė daugiausia žemėlapių per paskutinius penkerius metus, naudojantis DBVS galima labai greitai, nors sudarytų žemėlapių gali būti tūkstančiai.
3. **Aktualumas** – iš principo tikimasi, kad skaitmeninėje duomenų bazėje „tiksli ir nauja informacija visada yra po ranka“ (nors ar taip visada bus, labai priklauso nuo tinkamo DB administravimo)
4. Galimybė **daugeliui naudotojų** naudoti tuos pačius duomenis. Dauguma šiuolaikinių DBVS leidžia vienu metu naudoti duomenų bazėje saugomus duomenis naudotojams, kurie jungiasi prie DB tiesiogiai ar per kompiuterių tinklą, ir to praktiškai nepajusti.

Kai naudotojų daug, skaitmeninių DB pranašumai dar labiau išryškėja.
Svarbiausias iš jų yra *centrinis valdymas*. Jei nėra tokio valdymo galimybės, beveik neįmanoma sistemiškai tvarkyti organizacijos duomenų. Išvardinsime pagrindinius centrinio valdymo teikiamus privalumus.

1. Sumažinamas duomenų perteklius – nebereikia kiekvienam projektui kaupti atskirų duomenų bankų, o naudojami tie patys ir visiems projektams bendri duomenys. Pavyzdžiui, jei yra

saugomas anksčiau sudarytas žemėlapis, jo sluoksniai gali būti panaudoti naujiems žemėlapiams sudaryti jų nekopijuojant. Žinoma, visiškai išvengti duomenų pertekliaus neįmanoma, o kai kada jis net reikalingas (saugumui, aiškumui, geresniam suvokimui). Vis dėlto, bet kokį atsiradusį perteklių būtina griežtai kontroliuoti.

2. Galima panaikinti prieštaravimus (jie dažniausiai atsiranda būtent dėl duomenų pertekliaus ir nesuderintų pakeitimų). Pavyzdžiui, jei yra kelios to paties žemėlapio kopijos, o vietovardžių rašyba pasikeitė, būtina vienodai ir tuo pačiu metu atnaujinti visas kopijas. Jei vienas faktas saugomas vienintelėje vietoje, pertekliaus nebus. Kai kurios DBVS turi priemones automatiškai kontroliuoti pakeitimus dublikatuose, tačiau toli gražu ne visos, be to, tos priemonės nėra tobulos.
3. Bendras naudojimas. Vienu metu su tais pačiais duomenimis be ypatingų apribojimų gali dirbti daug žmonių (pavyzdžiui, sudarinėti teminius žemėlapius naudodami tą patį geografinio pagrindo sluoksnį).
4. Galima įvesti apribojimus saugumui palaikyti. Tai daroma, pavyzdžiui, ribojant naudotojų prieigą (duomenų įvedimą, naikinimą, keitimą) prie skirtingų DB dalių. Bet apskritai, duomenų saugumo problemoms skaitmeninės DB yra jautresnės, negu senos analoginės.
5. Vientisumo palaikymas. Tai saugomų duomenų tikslumo ir teisingumo garantavimas. Net jei nėra perteklinių duomenų, kai kurie įvesti duomenys gali būti klaidingi. Pavyzdžiui, gali būti nurodyta, kad darbuotojas priklauso neegzistuojančiam padaliniui, dirba ne 40, o 400 valandų per savaitę ir pan. DBVS leidžia įvesti vientisumo taisykles ir kontroliuoti, ar įvedami/keičiami duomenys jas tenkina. Vientisumas taip pat yra svarbesnis struktūrizuotai DB, negu šiaip duomenims, nes prie duomenų bazės prieina daugiau žmonių, ir įvairias operacijas atlikti yra daug paprasčiau. Padarius klaidą, gali nukentėti daug žmonių, naudojančių tuos pačius duomenis.
6. Galimybė derinti prieštaringus reikalavimus. Panaudojant įvairias optimizavimo strategijas, tam tikriems tikslams galima skirstyti darbo greičio ir kitus prioritetus, ieškant kompromiso tarp naudotojų.
7. Standartų laikymasis. Bendras duomenų naudojimas skatina duomenų, jų apdorojimo procedūrų, perdavimo būdų standartizavimą. Standartai yra svarbiausi kai keičiamasi duomenimis, pavyzdžiui, susitarimai dėl duomenų formatų – *Unicode, ISO*. Standartiniai yra ir kompiuterinės grafikos keitimosi formatai, pavyzdžiui, EPS, taip pat visiems gerai žinomas PDF formatas. Standartai gali būti žinybiniai, korporaciniai, nacionaliniai, tarptautiniai ir kt. Kartografijoje standartizavimas yra ypač svarbus siekiant pagerinti žemėlapių komunikacinę kokybę, sudaryti galimybę tarpusavyje palyginti vienos teritorijos ar tematikos žemėlapius. Kartografiniai standartai reglamentuoja ne tik geodezinį pagrindą, projekcijas, mastelį, bet ir sutartinius ženklus, šriftus ar spalvas. 2011 m. ISO 19100 standartų serija apėmė daugiau kaip dvidešimt geografinių duomenų standartų.

Kad visi šie privalumai tikrai būtų įgyvendinti, duomenų bazė turi būti administruojama.
Yra du administravimo lygmenys:

1. *Duomenų administratorius* – tai organizacijos vadovybės lygmens specialistas, kuris atsakingas už jos didžiausią turtą – duomenis. Jis neprivalo būti informacinių technologijų specialistas, nors bendras DB technologijų išmanymas bet kuriuo atveju reikalingas. Duomenų administratorius yra gerai susipažinęs su organizacijos reikmėmis, jis sprendžia, kurie duomenys svarbiausi, kas, kada ir kaip turi teisę juos naudoti. Jis taip pat atsakingas už duomenų saugumą organizaciniu lygmeniu.
2. *Duomenų bazės administratorius* – informacinių technologijų specialistas profesionalas, kuris įgyvendina duomenų administratoriaus sprendimus. Dažnai jis turi savo darbuotojų komandą.

DBVS funkcijos.

1. *Duomenų slaptumas*. Duomenų naudojimas yra kontroliuojamas.
2. *Duomenų vientisumas ir neprieštaringumas*. Yra galimybės jį automatiškai tikrinti.
3. *Naudojimo sinchronizavimas*. Tai reiškia, kad skirtingi naudotojai naudoja vienintelę DB ir dėlto nekyla problemų.
4. *Duomenų minimizavimas*. Duomenų bazėse duomenys be būtino reikalo nėra dubliuojami.
5. *Avarinė apsauga*. Tai galimybė atkurti duomenis arba bent jų dalį, jei jie prarandami dėl programinės ar techninės įrangos sutrikimų, klaidingų naudotojo veiksmų ir pan.

DBVS problemos.

Reikia paminėti ir skaitmeninių duomenų bazių valdymo sistemų trūkumus, ar, tiksliau, silpnąsias jų vietas.

1. Didelė rizika prarasti duomenis, jei nėra tinkamos apsaugos. Užtenka keleto nesudėtingų veiksmų ir visi organizacijos duomenys gali būti negrąžinamai sunaikinti, ko neatsitinka laikant duomenis segtuvuose.
2. Didelės išlaidos tokiai sistemai įdiegti – reikalinga brangi aparatūra, programinė įranga ir kvalifikuotas personalas.
3. Įdiegus skaitmeninę DBVS, visa organizacijos veikla tampa priklausoma nuo sėkmingo operacijų įvykdymo ir nuo personalo, susipažinusio su duomenų bazės struktūra..
4. Sistema yra sudėtinga.

Reliacinės sistemos idealiai tinka duomenų bazės sampratai iliustruoti. Dar kartą priminsime reliacinių DBVS pagrindines savybes.

1. Duomenys naudotojui pateikiami kaip lentelės ir tik kaip lentelės.
2. Sistemoje yra priemonės naujoms lentelėms kurti.

VI.3 DUOMENŲ BAZĖS ARCHITEKTŪRA.

Po bendro įvado reikėtų pakomentuoti duomenų bazės struktūrą, kuri būdinga daugumai sistemų. Tai architektūra, pagrįsta ANSI SPARC (angl. *American National Standards Institute, Standards Planning And Requirements Committee*) rekomendacijomis.

DBVS lygmenys.

Sistemos lygmuo šiuo atveju suprantamas kaip schema, pagal kurią interpretuojama sistema – ji keičiasi pereinant nuo labai konkretaus „mašininio“ abstrakcijos lygmens prie apibendrinto „žmogiškojo“. Bendriausi DBVS lygmenys yra šie.

1. *Fizinis*. Tai duomenys, saugomi skaitmeniniu pavidalu. Šis lygmuo naudotojui neturi būti žinomas ir apskritai rūpėti, pavyzdžiui, ar spalva saugoma kaip koordinatės konkrečiame modelyje, ar jai paprasčiausiai priskirtas numeris, ir kurioje disko vietoje jis saugomas.
2. *Loginis*. Tai duomenų bazės struktūra, sudaryta pagal kurį nors praeitame skyriuje aprašytą duomenų modelį.
3. *Koncepcinis*. Tai lygmuo, kuriame DB informacija pateikiama naudotojams jiems patogiu pavidalu. Sukuriami virtualūs (fiziškai sistemoje nesaugomi) objektai – ataskaitos, formos, kurie yra tik realios struktūros matymo būdai; jų nėra fizinėje atmintyje.

Duomenų nepriklausomumo lygmenys

Dažnai tuos pačius duomenis tenka vaizduoti skirtingai. Pavyzdžiui, pasaulio atlasas gali būti pristatomas pagal žemėlapių temas, arba pagal regionus; informacija apie žmones klasifikuojama pagal pareigas arba pagal projektus, kuriuose jie dalyvauja. Kartais dėl to būna sunku suderinti duomenis, pavyzdžiui jei žemėlapio autoriaus identifikatorius yra paso numeris, o kartografo – asmens kodas, darant suvestinį sąrašą, gali atsirasti nesuderinamumo problema. Gerai, jei yra galimybės šiuos duomenis suvesti į vieningą formatą.

Kartais duomenų struktūrą patogu išskaidyti, arba atvirkščiai, sujungti. Tai taip pat neturi atsiliepti tų duomenų naudojimui.

Nr.	Leidinys	Autorius

⇔

Nr.	Leidinys

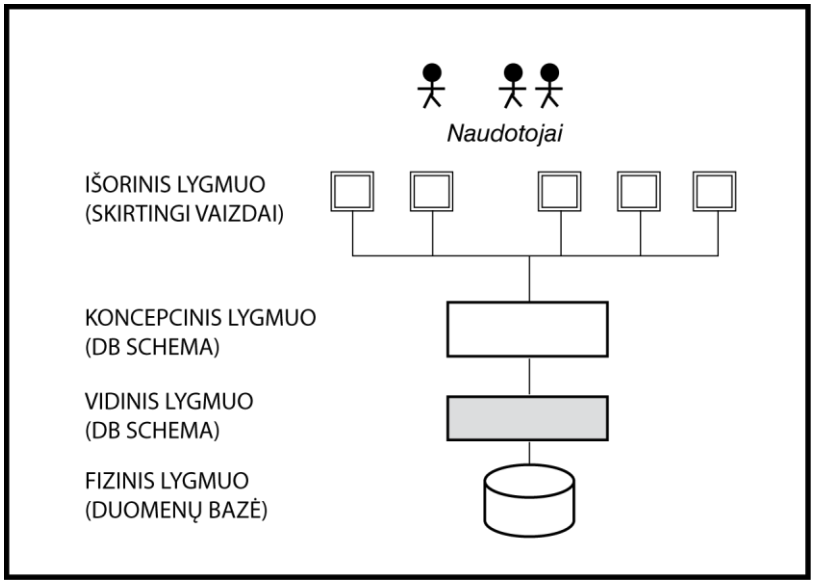
ir

Nr.	Autorius

1. *Fizinis nepriklausomumas* egzistuoja tada, kai galima keisti fizinę DB, nesikeičiant loginiam modeliui. Tam reikalinga speciali duomenų bazės architektūra, numatanti skaitinių, tekstinių ir kt. duomenų konvertavimą, matavimo vienetų keitimą ir pan.
2. *Loginis nepriklausomumas* yra, jei galima keisti loginį modelį išsaugant nepakitusius naudotojo „vaizdus“, pavyzdžiui, žemėlapis ekrane išlieka toks pat, nepriklausomai nuo to, ar jis saugomas kaip GIS duomenų bazė, ar kaip paprastas vektorinės grafikos failas .

Kalbant apie duomenų nepriklausomumą, didžiausią grėsmę jam kelia duomenų bazės augimas ir su juo susijęs struktūros keitimas

Atitinkamai išskiriami trys DB architektūros lygmenys.
Vidinis – artimas fizinio duomenų saugojimo specifikai, t.y., susijęs su informacijos saugojimo fiziniuose įrenginiuose ypatumais.
Koncepcinis – tarpinis lygmuo, kuriame atliekamas naudotojų poreikių apibendrinimas. Tai yra, gali būti keli išoriniai vaizdai, skirti atskiroms DB dalims, bet visada tik vienas koncepcinis modelis.
Išorinis lygmuo yra artimiausias naudotojams, t.y., susijęs su informacijos pateikimu skirtingoms naudotojų grupėms. Tai pats abstrakčiausias lygmuo (abstrakcija šiuo atveju reiškia žmogiškų, ne mašininių terminų naudojimą).



VI-2 pav. Duomenų bazės architektūros lygmenys.

Paveiksle (VI-2 pav.) parodyti DB architektūros lygmenys. Galutiniai naudotojai – tai prie terminalų dirbantys žmonės arba taikomosios programos. Duomenų vardai gali būti keičiami abiejuose aukštesniuose lygmenyse.

Kalbant apie architektūrą nėra labai svarbu, pagal kokį modelį sudaryta duomenų bazė. Pavyzdžiui, reliacinėje sistemoje: koncepcinis lygmuo būtinai yra reliacinis (t.y., jame matomos tik lentelės ir reliaciniai operatoriai). Išorinis lygmuo taip pat reliacinis, arba bent jau labai artimas reliaciniam. Vidinis lygmuo – NE reliacinis. Jame visi objektai atrodo lygiai taip pat, kaip ir kitų sistemų: įrašai, rodyklės, indeksai. Reliacinė teorija iš esmės nieko nepasako apie vidinį lygmenį. Ji susijusi tik su naudotojo suvokimu.

Išorinis lygmuo.
Tai individualaus naudotojo lygmuo, kuriuo dirba programuotojai arba bet kokio pasirengimo kiti naudotojai. Programuotojai su DBVS bendrauja programavimo kalba, visi kiti – užklausų kalba, naudodamiesi formomis, meniu ir kt. Užklausų kalba turi ne tik būtinus duomenų operatorius, bet ir programavimo kalbos aplinką (pavyzdžiui, MS Access – MS Visual Basic dialektas), kuri leidžia realizuoti ciklus, sąlygas ir kitas konstrukcijas.
Duomenų kalba sudaryta iš dviejų dalių: duomenų apibrėžimo kalba DDL (angl. *Data Definition Language*) ir duomenų valdymo kalba DML (angl. *Data Manipulation Language*).

Kaip jau minėta, konkretų naudotoją domina dažniausiai ne visa duomenų bazė, o tik konkreti jos dalis. Jam ji pateikiama išoriniu vaizdu, „langu“, pro kurį jis mato reikiamus duomenis taip, kaip jam patogiausia juos matyti; pavyzdžiui, informaciją apie personalą arba projektus. Atrenkama naudotojams informacija nebūtinai atkartoja fiziškai saugomus duomenis. Operacijas naudotojas taip pat atlieka ne su saugomais, bet su išoriniais duomenimis. Už tai, kad taip būtų, yra atsakingas duomenų bazės administratorius.

Koncepcinis lygmuo.
Šiuo lygmeniu matoma DB atitinka tikrąją informacijos struktūrą ir apimtį, bet dar niekaip nėra susijusi su fiziniais duomenų saugojimo aspektais. Įmonės atveju koncepcinis lygmuo dažnai turi apimti ne tik duomenis, bet ir taisykles, aprašančias kaip ir kam jie naudojami. Iš tikrųjų dar nė viena iš komercinių sistemų to nepalaiko.

Uždaviniai, kuriuos atlieka duomenų bazės administratorius.

1. Koncepcinės schemos sudarymas. Į DBVS perkeliamas duomenų administratoriaus sudarytas loginis modelis.
2. Vidinės schemos (kaip iš tikrųjų saugomi duomenys) sudarymas.
3. Bendravimas su naudotojais. Duomenų bazės administratorius teikia jiems reikalingus duomenis, kuria išorinius vaizdus, konsultuoja.
4. Saugumo ir vientisumo taisyklių apibrėžimas (tai – svarbi koncepcinės schemos dalis).
5. Atsarginis kopijavimas ir atkūrimas. Kai tik įmonė perduoda savo duomenis DBVS, ji tampa kritiškai priklausoma nuo jos sėkmingo funkcionavimo. Jei kuri nors DB dalis pažeidžiama dėl žmogaus klaidos, technikos gedimų ar programų sutrikimo, reikia turėti priemones kuo greičiau jai atkurti, tuo pačiu kiek galima mažiau paliečiant kitas, nepažeistas, dalis. Duomenų bazės administratorius turi numatyti galimus tokių įvykių scenarijus ir atitinkamus veiksmus bei juos užtikrinti. Duomenų bazės administratorius atlieka ir dampingą – DB dalių reguliarų išsaugojimą išorinėse laikmenose.

Čia išryškėja vienas duomenų laikymo ne vienoje vietoje privalumas – skirtingoms duomenų bazėms dampingo intervalai gali būti skirtingi.

6. Efektyvumo palaikymas reaguojant į pasikeitusius reikalavimus ir reorganizuojant DB.

DBVS – tai, visų pirma, programinė įranga, valdanti duomenų bazės naudojimą. Valdymas vyksta taip:

1. Naudotojas prašo leidimo atlikti veiksmą su duomenų baze (dažnai SQL kalba);
2. DBVS priima ir analizuoja prašymą;
3. DBVS atrenka ir pateikia naudotojui jo išorinę schemą;
4. DBVS atlieka reikiamas operacijas.

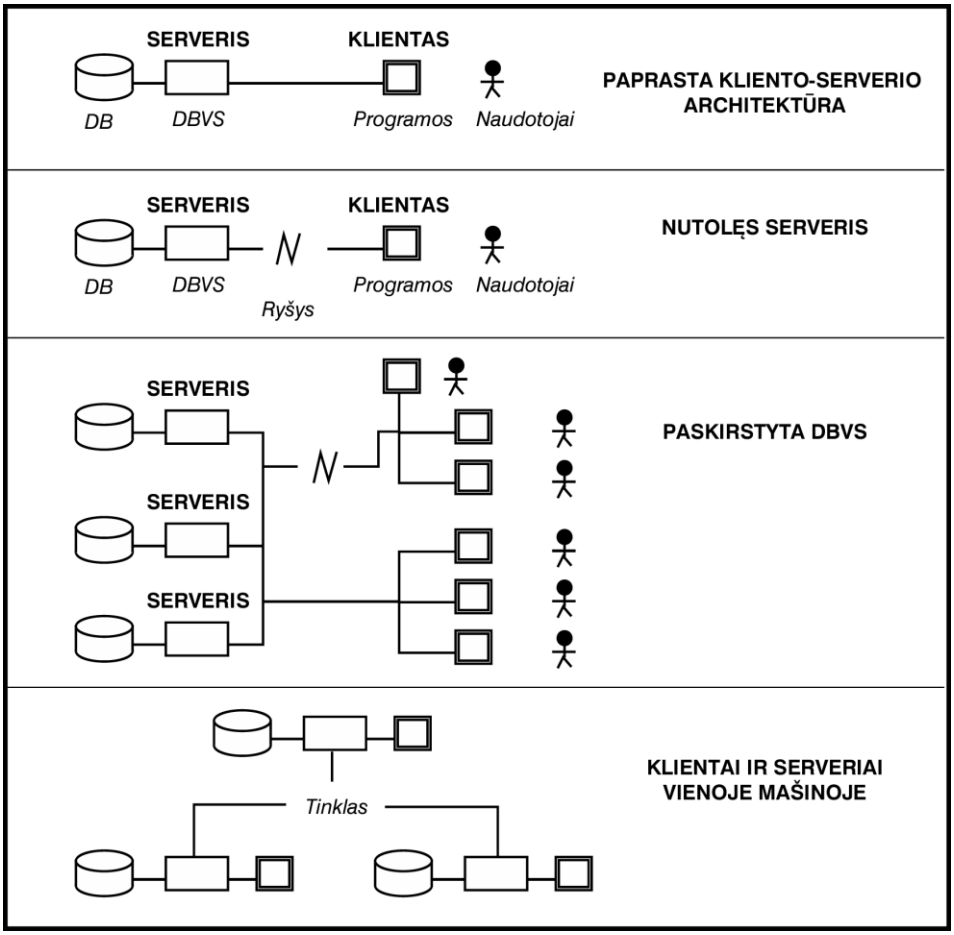
Grafinės naudotojo sąsajos priemonės – tai riba, už kurios naudotojas nebemato nieko. Naudotojo sąsaja priklauso išoriniam lygmeniui, nors praktiškai, bent jau dabartinėse DBVS, išorinis lygmuo nedaug skiriasi nuo koncepcinio.

Failų sistema ir DBVS.
Failų sistema iš pirmo žvilgsnio atlieka panašias operacijas: trinti, kurti, keisti duomenis. Bet ji turi keletą esminių trūkumų palyginus su DBVS:

1. Negalima atsižvelgti į įrašų vidinę struktūrą, t.y., ir vykdyti išrinkimo pagal sąlygą užklausų.
2. Nepalaikomas saugumas, vientisumas, atstatymas ir atsarginės kopijos.
3. Mažai palaikomas duomenų nepriklausomumas.

VI.4 KLIENTO-SERVERIO ARCHITEKTŪRA

Duomenų bazės *serveriu* vadinama techninė platforma, kurioje funkcionuoja DBVS (visuose trijuose lygmenyse). *Klientas* yra taikomoji programa arba individualus naudotojas.



VI-3 pav. Kliento-serverio architektūros tipai

Paveiksle (VI-3 pav.) parodytos paprasta ir paskirstytos duomenų bazių valdymo sistemos. Tais atvejais, kai sistema aptarnauja daug naudotojų, serveris ir klientas yra skirtingos mašinos. Galima ne tik atskirti klientą ir serverį, bet ir sujungti į tinklą daugelį mašinų taip, kad jos dirbtų lygiagrečiai.

Galima paskirstyti ir pačią duomenų bazę tarp keleto mašinų, taip sudarant DB serverių tinklą. Pavyzdžiui, Lietuvos gyventojų registro Alytaus dalį prasminga saugoti Alytuje, o Vilniaus – Vilniuje. Jei klientas tokioje sistemoje vienu metu kreipiasi tik į vieną serverį, ji kokybiškai nesiskiria nuo paprastos. Jei yra galimybė kliento užklausą apdoroti keliuose serveriuose vienu metu – tai yra *paskirstyta* DBVS. Bet kuriuo atveju klientui neturi būti svarbu, kas fiziškai sudaro serverį (viena ar kelios mašinos) ir kur jis yra.

Paskirstytų sistemų privalumai:

1. Kai lygiagrečiai dirba keletas procesorių, operacijos atliekamos greičiau.
2. Serveris gali būti specialiai sukurtas siekiant didesnio efektyvumo.
3. Klientas gali būti paprastas asmeninis kompiuteris, kuris tenkina ir kitas naudotojo reikmes bei turi jam patogiausią sąsają.
4. Vieną DB tuo pačiu metu gali naudoti skirtingos sistemos (pvz., Windows, MacOS, Unix).

Paskirstytos sistemos, dinamiškai skirstančios serverių resursus ir duomenis yra **debesų kompiuterijos** (angl. *cloud computing*) – virtualios informacinių technologijų infrastruktūros,

pasiekiamos Internetu kaip paslauga, architektūros pagrindas. Įdomu tai, kad nuo decentralizuotų kliento-serverio sistemų debesų kompiuterijoje tarsi vėl grįžtama prie centralizuotų sprendimų (bendrai naudojami, "fermose" koncentruoti galingi serveriai), kurie leidžia taupyti laiką ir kaštus.

VI.5 RELIACINĖS DUOMENŲ BAZIŲ VALDYMO SISTEMOS

Šiuo metu beveik visi duomenų bazių modeliai yra reliaciniai (angl. *relation* – santykis, lentelė). Šiuo metu yra naudojama daugiau kaip 200 komercinių RDBVS produktų. MS Access taip pat yra reliacinė duomenų bazių valdymo sistema. Reliacinio modelio principus pirmą kartą 1969–70 metais išdėstė Edgaras Kodas. Tai abstrakti duomenų teorija, pagrįsta matematiniais metodais – aibių teorija ir predikatų logika. Kodas, būdamas matematiku, įdiegė duomenų bazių valdymo sistemose griežtus principus ir tikslumą – tai ko jose iki tol trūko. Reliacinį modelį detaliai nagrinėsime VII skyriuje, o kol kas apibrėšime reliacines sistemas ir pagrindines jų sąvokas neformaliai, remdamiesi pavyzdžiais. Be to, šio paminėsime keletą bendrų duomenų bazių valdymo sistemų aspektų, kurie būdingi ne vien reliacinėms sistemoms, tačiau geriausiai iliustruojami būtent jų pavyzdžiu. Tai optimizavimas, sistemos katalogas, vaizdai ir indeksai.

Reliacinė sistema dažniausiai suprantama kaip sistema, tenkinanti dvi sąlygas:

1. Duomenys naudotojui pateikiami kaip lentelės ir tik kaip lentelės.
2. Naudotojui yra prieinami operatoriai kurti naujoms lentelėms: kaip minimumas – išrinkti duomenų eilutėms (RESTRICT/SELECT), atrinkti stulpeliams (PROJECT) ir sujungti lentelėms (JOIN).

Tarkime, turime duomenų bazę su dviem lentelėmis:

Projektai

ProjNr	Pavadinimas	Biudžetas
P1	Nacionalinis atlasas	2 000 000
P2	Istorijos atlasas	100 000
P3	...	

Kartografai

AsmKodas	Pavardė	ProjNr	Pareigos
0024	Jonaitis	P1	Projektų vadovas
1233	Petraitis	P3	Vyr. kartografas
1234		...	...

1. SELECT – iš lentelių gali būti išrenkamos norimos eilutės (pavyzdžiui, projektai, kurių biudžetas yra didesnis už 100 000 iš lentelės ***Projektai***)
2. PROJECT – iš lentelių gali būti išrenkama norima stulpelių aibė (pavyzdžiui, AsmKodas ir Pavardė iš lentelės ***Kartografai***)
3. JOIN – lentelės gali būti sujungtos į vieną (pavyzdžiui, Kartografai ir Projektai per jų bendrą stulpelį ProjNr)

Kad būtų galima sujungti, turi būti vienas bendras stulpelis abiejose lentelėse, o jungiama pagal *vienodas* reikšmes jame. Jei kurioje nors lentelėje nėra atitinkančios reikšmės, eilutė iš viso neįtraukiama.

Reliacinės sistemos uždarumas.

Kiekvienos operacijos rezultatas visada yra lentelė, net jei išrenkama vienintelė reikšmė, pavyzdžiui, kartografas, kurio asmens kodas yra 0024, tai nėra šiaip skaičius, o lentelė su vienu stulpeliu ir viena eilute. Todėl vieną reliacinę operaciją gali sekti kita, pavyzdžiui JOIN–PROJECT–SELECT. Taigi, galima sudaryti sudėtines išraiškas, kuriose operandai yra ne lentelių vardai, o kitos išraiškos.

Dar vienas svarbus momentas yra tas, kad reliacinės operacijos taikomos ne kiekvienai eilutei po vieną kartą, o iš karto visai lentelei, t.y., reliacinių operatorių operandai yra **aibės**. Aibių apdorojimo galimybė – viena svarbiausių RDBVS charakteristikų.

Kita puiki RDBVS savybė yra ta, kad visas informacinis DB turinys pateikiamas vieninteliu – **išreikštiniu** pavidalu. Toks būdas yra vienintelis leidžiamas. Nėra jokių rodyklių, siejančių lenteles. Be abejo, ryšys tarp lentelių „Projektai“ ir „Kartografai“ egzistuoja, tačiau jis realizuojamas ne per nuorodas, o per ProjNr reikšmę lentelėje „Kartografai“. Kitose, nereliacinėse, sistemose ryšys gali būti sukuriamas kaip nuorodos, kurios apsunkina darbą su sistema. Žinoma, RDBVS jos taip pat atsiranda, bet tik fiziniame lygmenyje, kuris yra naudotojui nematomas. Reikia nepamiršti, kad lentelės – tai tik loginės struktūros, kurios skirtos naudotojui ir slepia nuo jo vidinį duomenų modelį.

Visos reikšmės RDB lentelėse yra atomai (nedalomos). Tai yra, eilutės ir stulpelio sankirtoje gali būti vienintelė reikšmė, o ne reikšmių grupė. Neleistina tokia situacija:

Kartografai

AsmKodas	SudarytiŽemėlapiai	Atlyginimas
0664	Z1, Z2	1000

Stulpelis ProjNr pavyzdyje yra tai, kas vadinama pasikartojimo grupe. Reliacinėse DB pasikartojimo grupės neleistinos, todėl sistemos tampa paprastesnėmis.

Kartografai

AsmKodas	SudarytiŽemėlapiai	Atlyginimas
0664	Z1	600
0664	Z2	400

Reliacinis modelis susijęs su trimis duomenų aspektais: struktūra (lentelės), apdorojimu (operatoriai) ir vientisumu. Pirmuosius du aspektus parodėme pavyzdžiu, liko paaiškinti vientisumo sąvoką.

Vientisumas.

Grįžkime prie duomenų bazės lentelių ***Kartografai*** ir ***Projektai***. Jų jungimas neturi prasmės, jei nėra taikomos dar ir įvairios taisyklės, susijusios su lentelėse saugomų duomenų interpretacija. Saugomi duomenys turi atitikti kokiu nors būdu aprašytas realias sąlygas taip, kad nebūtų galima įvesti akivaizdžiai klaidingų duomenų. Tokios sąlygos vadinamos *vientisumo taisyklėmis*, kurių duomenų bazėje reikia laikytis, kad būtų patenkinti reliacinio modelio reikalavimai. Pačios bendriausios vientisumo taisyklės gali būti suformuluotos taip.

1. Kiekviena eilutė lentelėje ***Projektai*** turi turėti vienintelę (nesikartojančią) atributo ProjNr reikšmę. Kiekviena eilutė Kartografų lentelėje turi turėti nesikartojančią atributo AsmKodas reikšmę.
2. Kiekviena ProjNr reikšmė lentelėje ***Kartografai*** turi sutapti su kuria nors ProjNr reikšme iš Projektų lentelės – taip turi būti, jei norime, kad kiekvienas dirbantis kartografas būtų priskirtas tik iš tikrųjų egzistuojančiam projektui.

ProjNr ir AsmKodas yra atitinkamų lentelių *pirminiai raktai*. ProjNr lentelėje ***Kartografai*** yra šios lentelės išorinis raktas, susietas su Projektų lentelės pirminiu raktu.

Beje, reliacinis modelis yra teorinis. Veikianti sistema nebūtinai turi šimtu procentų palaikyti šią teoriją, tam kad galėtų vadintis reliacine pagal apibrėžimą. Kol kas rinkoje nėra produkto, kuris absoliučiai atitiktų teorinius reikalavimus. Tai nereiškia, kad kai kurie teorijos aspektai nėra svarbūs – atvirkščiai, kiekviena smulkmena yra susijusi su praktika. Teorijos tikslas yra sukurti pagrindą atsirasti absoliučiai praktiškai sistemai (kaip ir kuriant kitas programų sistemas, pavyzdžiui, grafiniai redaktoriai niekaip iki galo nepakeičia rankų darbo, tačiau to yra nuosekliai siekiama).

Optimizavimas.
Kaip minėjome, reliacinės operacijos, pavyzdžiui, SELECT, PROJECT ir JOIN, vykdomos su aibėmis. Todėl reliacinės kalbos, tokios, kaip SQL, dažnai vadinamos neprocedūrinėmis kalbomis. Tai reiškia, kad naudotojas tik nurodo, ką reikia padaryti, kokio rezultato jis nori, o pačia procedūra užsiima sistema, atlikdama reikiamus veiksmus duomenų bazėje. Tai labai palengvina sistemos programavimą, nes vietoje keliolikos eilučių kodo procedūrine kalba norimą rezultatą galima gauti parašius nesudėtingą kreipimąsį į sistemą, kuris vadinamas *užklausa* (angl. *query*). Tačiau sudėtingos užklausos gali būti sistemos vykdomos naudojant skirtingus algoritmus, o operacijos atliekamos skirtinga tvarka.

Optimizavimas gali daug kartų padidinti operacijų vykdymo greitį ir taip sumažinti skaičiavimų sąnaudas. Už tai, kaip atliekama automatinė navigacija duomenų bazėje, yra atsakingas svarbus RDBVS komponentas – optimizatorius. Jo paskirtis yra kiekvienai užklausai parinkti efektyviausią jos įvykdymo būdą. Optimizavimas gali būti pagrįstas skaičiavimo sąnaudų vertinimu (angl. *cost-based optimization*), euristinis (naudojantis apytikslius metodus) arba mišrus.

Pavyzdžiui, turime užklausą, kuri išrenka eilutę ir priskiria atitinkamą jos stulpelį kintamajam ManoProjektas:

ManoProjektas:= (***Kartografai*** WHERE AsmKodas = „0664“) [ProjNr].

Net ir tokiu paprastu atveju yra bent du galimi duomenų paieškos būdai.

1. Nuosekliai peržiūrima lentelė ***Kartografai*** tol, kol nerandamas ieškomas įrašas.
2. Jei stulpeliui AsmKodas yra sukurtas indeksas (o jis greičiausiai sukurtas pagal DBVS reikalavimus, nes AsmKodas yra lentelės pirminis raktas, kuris paprastai yra indeksuojamas), pagal indeksą keliais žingsniais galima pereiti prie įrašo su AsmKodas reikšme „0664“ duomenų.

Optimizatorius pasirenka vieną iš galimų strategijų remdamasis šia informacija:

1. Į kokias lenteles kreipiamasi užklausoje.
2. Kokio dydžio yra tos lentelės.
3. Kokie yra sukurti indeksai.
4. Kaip duomenys fiziškai sugrupuoti diske.
5. Kokios operacijos naudojamos.

Sąnaudų vertinimu pagrįstas optimizavimas yra efektyvus, bet sudėtingas, nes kiekvieną kartą jį atliekant turi būti įvertinta DB statistika (įrašų, ryšių, reikšmių skaičius). Euristinis optimizavimas naudoja kelias paprastas taisykles:

- kuo anksčiau atlikti išrinkimo operacijas (SELECT), taip sumažinant analizuojamų įrašų skaičių;

- anksti atlikti projektavimo operacijas (PROJECT), taip sumažinant analizuojamų stulpelių skaičių;
- labiausiai rezultatą ribojančias išrinkimo ir jungimo operacijas atlikti anksčiau, negu mažiau ribojančias panašias operacijas.

Naudotojas optimizavimo proceso nemato ir jo paveikti negali.

Katalogas.

Kiekviena reliacinė DBVS privalo turėti katalogo (duomenų žodyno) funkciją.
Kataloge saugoma detali informacija apie visus sistemoje esančius objektus – objektų deskriptoriai, metaduomenys. Sistemos objektai – tai, pavyzdžiui, lentelės, indeksai, vientisumo taisyklės ir kt. Katalogas saugomas kartu su visomis (vidinėmis, išorinėmis ir koncepcinėmis) schemomis. Jo duomenis naudoja ir optimizatorius.

Puiki RDBVS savybė yra ta, kad katalogą taip pat sudaro lentelės. Tačiau tai yra sisteminės lentelės, kurios skiriasi nuo naudotojų sukurtų. Naudotojas turi galimybę kreiptis į katalogą. Pavyzdžiui, sistemose yra katalogo lentelės ***TABLES*** (laukai: Name, ColCount, RowCount) bei ***COLUMNS*** (laukai: TabName, ColName). Tai sisteminės lentelės, kurios aprašo duomenų bazėje esančias lenteles ir stulpelius. Į jas įtrauktos tik tos lentelės, kurios turi vardus, skirtingai nuo tarpinių skaičiavimų rezultatų – bevardžių lentelių.

Katalogas aprašo pats save: ***TABLES*** lentelėje yra eilutė vardui „TABLES“.

Su katalogu galima atlikti užklausas, pavyzdžiui, kokius stulpelius turi lentelė ***Projektai***:
(COLUMNS WHERE TabName = „Projektai“)[ColName]

Galima rasti lenteles, kuriose yra stulpeliai „Nr“:
(COLUMNS WHERE ColName = „Nr“)[TabName]

Pagrindinės lentelės ir vaizdai.

Pradinės lentelės, nuolat saugomos duomenų bazėje, vadinamos pagrindinėmis. Lentelės, kurios gaunamos iš pradinių kokiomis nors operacijomis, yra išvestinės. Pagrindinės lentelės egzistuoja savarankiškai, išvestinės lentelės priklauso nuo pagrindinių.

Išvestinė lentelė – tokia, kuri apibrėžiama per kitas lenteles.

Pagrindinė lentelė – tai lentelė, kuri nėra išvestinė.

RDBVS visų pirma turi būti priemonės sukurti pagrindinėms lentelėms. SQL kalboje tą daro komanda CREATE TABLE, o MS Access ji gali būti realizuojama ir per grafines sąveikos priemones.

Pagrindinės lentelės visada turi vardus, kurie turi būti nurodomi jas sukuriant. Dauguma išvestinių lentelių yra bevardės. Tačiau palaikomas bent vienas išvestinių lentelių tipas, kurios turi vardus. Tai – vaizdai (MS Access vaizdus atitinka formos). Jie priklauso nuo pagrindinių lentelių.

Pavyzdžiui,
CREATE VIEW Geriausi AS (Kartografai WHERE SudarytaŽemelapių > 100) [AsmKodas, Atlyginimas]

Atlikus šią operaciją, sistema įsimena duomenis po žodžio AS, t.y., išsaugo struktūrą kataloge vardu „Geriausi“. Tačiau duomenys nėra skaičiuojami ir nauja lentelė nesukuriama, nors naudotojui šis sukurtas vaizdas tampa lyg ir realia duomenų bazės lentele, su savo stulpeliais ir eilutėmis. Tai –

virtuali lentelė, kurią būtų galima laikyti rezultatu, jei jos kūrimo instrukcijoje nurodyta išraiška iš tikrųjų būtų išskaičiuota. Tuo tarpu ji visai neturi būti skaičiuojama, nes vaizdas yra tik „langas“, pro kurį matoma pagrindinė lentelė „Kartografai“.

Bet kokie pakeitimai padaryti pagrindinėje lentelėje automatiškai ir nedelsiant tampa matomi tokiame „lange“, žinoma, jei tik jie yra susiję su atitinkamu vaizdu. Ir atvirkščiai, bet kokie vaizdo pakeitimai automatiškai ir nedelsiant atliekami ir pagrindinėje lentelėje.

Su vaizdais galima atlikti operacijas, kurios sistemoje konvertuojamos į ekvivalenčias operacijas su pagrindinėmis lentelėmis. Vaizdas gali būti sukurtas iš kiek norima daug pagrindinių lentelių. Skirtumas tarp pagrindinės lentelės ir vaizdo nusakomas taip:

- Pagrindinės lentelės yra realios, jos saugomos diske.
- Vaizdai realiai neegzistuoja, jie yra tik skirtingi pagrindinių lentelių peržiūros būdai, matomi vartotojui kaip užklausos rezultatas. Jie pateikiami formose arba ataskaitose.

Užklausa gali būti apibrėžta iš anksto, arba papildyta vartotojo.

Virtualios lentelės turi šuos privalumus.

1. Logiškai nepriklauso nuo realių lentelių.
2. Galima suformuoti virtualias lenteles pagal poreikius skirtingoms naudotojų grupėms.
3. Lengviau suvokiamos
4. Duomenys yra paslėpti nuo tų, kam jų nereikia.

Kai kuriais atvejais vartotojas gali keisti virtualios lentelės laukų reikšmes. Praktiškai tai įmanoma tik tada, jei virtuali lentelė yra vienintelės pagrindinės lentelės poaibis. Negalima modifikuoti lentelių sąjungos

Virtualių lentelių modifikavimo taisyklės:

1. Jei laukas gautas iš aritmetinio reiškinio ar konstantos, galima DELETE, bet ne INSERT ar UPDATE operacijos.
2. Jei laukas gautas kaip standartinės funkcijos rezultatas – laukas nemodifikuojamas.
3. Jei lentelė sudaryta sujungiant keletą bazinių lentelių – laukas nemodifikuojamas.
4. Jei likę nepanaudoti virtualioje lentelėje bazinės lentelės laukai yra specifikuoti kaip NOT NULL, negalimos INSERT ar UPDATE operacijos.

Indeksavimas.

Jei įrašai lentelėje saugomi išrūšiuoti pagal raktą ar kitą atributą, dažnai sudaromas indeksas – lentelė, į kurią kreipiantis pagal rakto reikšmę, gaunamas įrašo su nurodytu raktu adresas.

Pavyzdžiui, reikia rasti gyventojus, gyvenančius Vilniuje ar kokiame kitame mieste. Jei tokios užklausos vykdomos dažnai, yra prasmė sukurti efektyvų duomenų saugojimo būdą – du failus, kurių viename yra duomenys apie gyventojus, o kitame – duomenys apie miestus, be to, miestai sutvarkyti pagal abėcėlę. Antrasis failas vadinamas darbuotojų lentelės *indeksu*.Yra dvi galimos strategijos:

a) išrinkti visus įrašus, kuriuose atributo Miestas reikšmė lygi „Vilnius“;
b) miestų sąraše išrinkti įrašą „Vilnius“ ir pasinaudoti iš jo einančiomis nuorodomis.

Antruoju atveju atliekama mažiau darbo.

Naudojant indeksą, lengviau patikrinti ir ar apskritai egzistuoja reikiamas įrašas, pavyzdžiui, ar yra gyventojas Šilutėje.

Indeksas – tai specialus saugomas failas, su kuriuo susieta procedūra, padedanti greitai surasti įrašus pagal atributų reikšmes. Šį failą sudaro duomenys (atitinkantys indeksuoto lauko duomenis) ir

rodyklės. Indeksą galima palyginti su reikšminių žodžių rodykle knygos gale – tai „suspausta“ hierarchinė struktūra.

Dvi svarbiausios indeksų naudojimo priežastys yra šios.

1. Pirminio rakto unikalumo palaikymas
2. Paieškos ir lentelių sujungimo pagreitinimas

Indeksų lentelėje gali būti daug, jie naudojami ir kartu, ir po vieną. Indeksuoti galima ir pagal laukų grupę, pavyzdžiui, miesto ir gatvės pavadinimus kartu.

Jei įrašai išrūšiuoti pagal raktą, indekso rodyklės gali rodyti ne į vieną įrašą, o į jų bloką (puslapį), kuriame bus ieškoma. Tas puslapis užkraunamas į operatyvinę atmintį, taip sumažinant „lėtų“ įvedimo/išvedimo operacijų skaičių.

Pavyzdžiui, gyventojų pavardėms sudaromas aukščiausio lygio indeksas pagal pirmąją raidę:

A	→	Aa	Ab	Ac	Ad	…
B	→	Ba	Bb	Bc	Bd	…
C	→	Ca	…			
…						

Tai – nuoseklus indeksas, jis yra efektyvesnis už tiesioginį, kuriame indeksas rodo į įrašą, o ne į įrašų bloką. Tačiau, kad būtų galima sukurti nuoseklų indeksą, failas turi būti išrūšiuotas pagal raktą. Kadangi neįmanoma failo išrūšiuoti pagal keletą raktų, paprastai rūšiuojama pagal dažniausiai naudojamą. Indeksai pagreitina išrinkimą, tačiau apsunkina atnaujinimą.

Dvylika Kodo taisyklių

Siekdamas pasipriešinti DBVS platintojų bandymams supaprastinti ar kitaip nukrypti nuo reliacinio modelio reikalavimų, ypač perdarant anksčiau sukurtus nereliacinius produktus, E. Kodas suformulavo trylika taisyklių (“dvylika” Kodo taisyklių, numeruojamų nuo nulio iki 12), apibrėžiančių, kokią DBVS galima laikyti reliacine. Deja, nė viena iš paplitusių DBVS neatitinka visų taisyklių. Tik mažiau žinomos, akademinėje aplinkoje vystomos DBVS rimtai siekia jas tenkinti. 2011 m. pradžioje vienintelis “beveik tikros” reliacinės komercinės DBVS pavyzdys buvo *Dataphor*.

Kai kurios taisyklės, ypač trečioji, kelia daug diskusijų dėl savo kontraversiškumo.

0. DBVS turi būti reliacinė tiek duomenų bazės, tiek valdymo prasme.
RDBVS duomenų bazei valdyti turi naudoti vien tik reliacines priemones.

1. Informacijos taisyklė
Visa DB informacija turi būti vaizduojama vieninteliu būdu, t.y., pateikiant reikšmes lentelių stulpelių ir eilučių sankirtose.

2. Užtikrintos prieigos taisyklė
Visus duomenis turi būti galima pasiekti. Tai perfrazuotas pirminio rakto reikalavimas, reiškiantis, kad bet kuri skaliarinė reikšmė duomenų bazėje privalo būti logiškai adresuojama nurodant lentelės ir jos stulpelio, kuriuose reikšmė saugoma, vardus bei eilutės, kurioje reikšmė saugoma, pirminio rakto reikšmę.

3. Nuoseklaus neapibrėžtų reikšmių naudojimo taisyklė
DBVS turi sudaryti galimybę palikti bet kurį lauką neužpildytą, be jokios reikšmės, bei tinkamai vaizduoti specifines reikšmes "nepateikta informacija" ir "netaikoma informacija" nepriklausomai nuo jų duomenų tipo.

4. Aktyvaus reliacinio katalogo taisyklė
DBVS turi palaikyti reliacinį katalogą (DB struktūros aprašą), kurį tam turintis teises naudotojas gali pasiekti kaip bet kuriuos kitus duomenis, naudodamas tą pačią užklausų kalbą.

5. Duomenų kalbos taisyklė
DBVS turi palaikyti bent vieną reliacinę kalbą, kurios sintaksė būtų tiesinė (t.y., kodas interpretuojamas iš kairės į dešinę). Kalba turi būti naudojama ir per naudotojo sąsają, ir taikomosiose programose bei palaikyti duomenų (taip pat jų vaizdų) apibrėžimo bei manipuliavimo operacijas, saugumo ir vientisumo apribojimus bei transakcijų valdymą (pradėti, patvirtinti ir atšaukti transakcijas).

6. Vaizdo atnaujinimo taisyklė
Sistema turi galėti atnaujinti visus vaizdus, kuriuos teoriškai galima atnaujinti.

7. Aukšto lygmens įterpimo, pakeitimo ir šalinimo operacijos
Sistema turi palaikyti įterpimo, pakeitimo ir šalinimo operatorius, taikomus aibėms. Tai reiškia, kad duomenis iš RDBVS turi būti galima gauti kaip aibes, surinktas iš daugelio eilučių ir (arba) lentelių, o įterpimo, pakeitimo ir šalinimo operacijos atliekamos ne su pavienėmis lentelių eilutėmis, o su bet kokiomis aibėmis.

8. Fizinis duomenų nepriklausomumas
Pasikeitimai fiziniame lygmenyje, t.y., fizinis duomenų saugojimo būdas turi būti atliekami nekeičiant duomenų struktūras naudojančių taikomųjų programų.

9. Loginis duomenų nepriklausomumas
Pasikeitimai loginiame lygmenyje, t.y., lentelių, stulpelių ar eilučių pakeitimai, turi būti atliekami nekeičiant duomenų struktūras naudojančių taikomųjų programų. Loginį nepriklausomumą sunkiau pasiekti, nei fizinį.

10. Vientisumo nepriklausomumas
Vientisumo reikalavimai turi būti nustatomi nepriklausomai nuo taikomųjų programų ir saugomi kataloge. Prireikus turi būti galima pakeisti vientisumo reikalavimus kiek įmanoma išvengiant keitimų taikomosiose programose.

11. Paskirstymo nepriklausomumas
Jei duomenų bazės dalys saugomos keliose skirtingose vietose, duomenų bazės naudotojai neturi to pastebėti. Taikomosios programos turi sėkmingai dirbti tiek su pirmą kartą paskirstyta, tiek su sistemos ribose perskirstyta duomenų baze.

12. Negriovimo taisyklė
Jei DBVS palaiko įrašų lygmens sąsają (t.y., vienu metu galima pasiekti vieną įrašą), tokia sąsaja negali būti panaudota sistemai griauti, pavyzdžiui, apeinant saugumo ar vientisumo apribojimus.

Klausimai diskusijai

Paaiškinkite duomenų nepriklausomumo sąvoką remdamiesi praktiniu pavyzdžiu.
Kaip manote, kokie yra centralizuoto duomenų valdymo trūkumai?
Kas atsitiktų, jei organizacijoje nebūtų duomenų administratoriaus? Duomenų bazės administratoriaus?

Užduotys savarankiškam darbui

Sudarykite giminės medžio informacijai saugoti reikalingos duomenų bazės koncepcinį modelį pagal sąlygas.

1. Kiekvienas žmogus turi unikalų asmens kodą ar numerį.
2. Kiekvienas žmogus turi tėvą ir motiną, kurie gali būti nežinomi.
3. Vyras ir moteris vienu metu gali sudaryti vieną santuoką.
4. Žmogus gali nutraukti vieną santuoką ir sudaryti kitą (kiek norima daug).
5. Turi būti saugoma susituokusios moters mergautinė pavardė.

Pabandykite optimizuoti modelį taip, kad liktų ne daugiau kaip dvi lentelės. Sutvarkykite ryšius.

Užduotys praktikos darbams

Sukurkite duomenų bazę pagal sudarytą giminės medžio koncepcinį modelį. Užpildykite lenteles duomenimis.

VII. RELIACINIS MODELIS

Nebijokite tobulumo! Jis jums tikrai negresia.
Salvador Dalí

Reliacinis modelis yra visų šiuolaikinių duomenų bazių valdymo technologijų pagrindas. Tai kelių dešimtmečių mokslinių tyrimų rezultatas, paremtas griežtais matematiniais principais. Jo pagrindinė ypatybė yra tai, kad duomenys pagal šį modelį saugomi lentelių pavidalu ir absoliučiai visi duomenų bazės objektai yra vaizduojami kaip *lentelės*. Iš to išplaukia, kad reliacinėje duomenų bazių sistemoje turi būti dar ir specialios *operacijos* įvairiems veiksmams su lentelėmis atlikti. Be to, turi būti taisyklės, padedančios kontroliuoti, ar duomenys lentelėse atitinka tikrovę, t.y., ar duomenų bazė tenkina taip vadinamus *vientisumo* reikalavimus.

Bet kurios duomenų bazių valdymo sistemos aprašymas nesiremiant reliaciniu modeliu gali būti tik paviršutiniškas. Nors pats modelis, be abejo, evoliucionuoja, jo pagrindas yra pakankamai stabilus. Toliau išsamiau panagrinėsime visus tris aukščiau minėtus duomenų aspektus: struktūrą (objektus), duomenų apdorojimą ir vientisumą reliacinės teorijos terminais.

VII.1 RELIACINIAI OBJEKTAI: DOMENAI IR SANTYKIAI

Kaip ir kiekvienoje teorijoje, reliacinėje teorijoje naudojami specialūs terminai. Svarbiausi iš jų yra: santykis, kortežas, kardinalumas, atributas, domenas, laipsnis ir pirminis raktas.

Neformalios šių terminų apibrėžtys galėtų būti tokios:

- *Santykis* – pagrindinis reliacinis objektas; tai, ką paprastai vadiname lentele. Santykio tipo kintamieji dažnai žymimi didžiąja raide R (nuo angliško žodžio *Relation*).
- *Atributas* atitinka esybės atributą ar lentelės stulpelį.
- *Kortežas* (angl. *tuple*) atitinka lentelės eilutę, kurią sudaro esybės atributų reikšmių rinkinys, tarsi lydintis svarbiausią rinkinio elementą – unikalų identifikatorių (iš to ir kilęs šis terminas).
- *Kardinalumas* – tai kortežų skaičius santykyje, dar žymimas *card*.
- *Santykio laipsnis* – tai santykio atributų skaičius, dar žymimas *deg* (angl. *degree*).
- *Pirminis raktas* – tai unikalus identifikatorius lentelėje, t.y., stulpelis ar jų derinys toks, kad jokiu momentu negali atsirasti dviejų lentelės eilučių, kuriose būtų vienodos to stulpelio (ar derinio) reikšmės. Pirminis raktas žymimas PK (angl. *Primary Key*)
- *Domenas* – tai atributo reikšmių aibė.

Reikia pastebėti, kad dauguma šiuo metu egzistuojančių reliacinių sistemų nepalaiko domenų, nors jie yra fundamentalus reliacinės teorijos komponentas.Todėl juos aptarsime detaliau.

Domenas.
Paimkime kaip atramos tašką mažiausią semantinį duomenų vienetą, kuris yra savarankiškas, pavyzdžiui, asmens kodas. Tokią reikšme vadinsime skaliarine arba atomu (neskaidoma reikšme). Laikoma, kad atomas neturi vidinės struktūros, žinoma, ne absoliučiai, o tik duomenų bazės kontekste. Pavyzdžiui, miesto pavadinimas gali būti išskaidytas į atskirus simbolius, tačiau prasmę jis turi tik tada, kai yra visų tų simbolių, išdėstytų nustatyta tvarka, junginys.

Domenas – tai vieno tipo skaliarinių reikšmių aibė, kuriai suteiktas vardas. Domenas gali būti, pavyzdžiui, visi įmanomi asmens kodai, atlyginimai ir pan. Iš domenų imamos realios atributų reikšmės. Tiksliau, kiekvienas atributas turi būti apibrėžtas vieninteliame domene. Žinoma, domeną sudaro visos apskritai galimos reikšmės, ne tik tos, kurios yra naudojamos duomenų bazėje konkrečiu momentu.

Pagrindinė domenų prasmė – apriboti galimas palyginimo operacijas. Tarkime, turime dvi užklausas:

```
SELECT *
FROM Projektai, Kartografai
WHERE Projektai.ProjNr = Kartografai.ProjNr
```

```
SELECT *
FROM Projektai, Kartografai
WHERE Projektai.Biudžetas = Kartografai.AsmKodas
```

Akivaizdu, kad pirmoji užklausa turi prasmę, tuo tarpu antroji – greičiausiai ne. Taip yra todėl, kad atributų reikšmes yra prasminga lyginti tik tada, kai jos priklauso tam pačiam domenui. Neužtenka, kad sutaptų baziniai duomenų formatai, pavyzdžiui, abi reikšmės būtų sveiki skaičiai, bet turi būti ir ta pati duomenų prasmė.

Jei sistema palaiko domenus, galima išvengti didelių klaidų atrenkant, jungiant ir atliekant kitas operacijas su lentelėmis, kai sulyginami pagal prasmę skirtingi atributai. SQL kalba, deja, domenų nepalaiko, todėl antroji užklausa yra praktiškai įmanoma ir būtų sistemos įvykdyta, jei tik sutaptų lyginamų duomenų tipai, nors jos rezultatas ir beprasmiškas.

Domenas – tai tik sąvoka, jis nebūtinai turi būti saugomas duomenų bazėje kaip reikšmių rinkinys. Bet jei sistema juos palaikytų, domenai būtų kuriami specialiu operatoriumi ir nurodomi kiekvienam atributui. Hipotetine reliacine kalba tai atrodytų maždaug taip:

```
CREATE DOMAIN AsmKodas char[9]
```

Bet kuriuo atveju yra prasmė kiekvieną atributą pavadinti taip, kad pavadinime atsispindėtų jo domenas (atributo ir domeno vardai sutapti neturėtų, nes keli atributai gali įgauti reikšmes iš to paties domeno, tačiau galimi vardų ir domeną rodančių priesagų deriniai, pavyzdžiui, Pavardė_dAsmenvardis). Teoriškai būtų patogu, jei su domenais būtų galima atlikti užklausas, pavyzdžiui, „Kokie santykiai duomenų bazėje turi atributą, apibrėžtą domene *Asmenvardis*“. Kol kas tai yra visiškai neįmanoma.

Nesunku pastebėti, kad domenas yra nei daugiau, nei mažiau, negu duomenų tipas. Tokį duomenų tipą galima sukurti dauguma programavimo kalbų:

```
Typedef Mokslas = {Geografija, Istorija, … Matematika};
Mokslas m;
```

Čia nurodoma, kad kintamasis m gali įgauti reikšmes iš domeno Mokslas. Analogiškai duomenų bazėje jo stulpelyje galėtų būti tik domeno Mokslas reikšmės, o ne bet kokie žodžiai. Tiesa, duomenų bazių valdymo sistemos palaiko pačius primityviausius domenus, t.y., pagrindinius duomenų tipus: sveikus, realius skaičius, simbolių eilutes ir pan.

Santykis.

Santykis (angl. *relation*) – tai matematinis terminas, reiškiantis aibę, dažnai vaizduojamą lentele. Terminas greičiausiai yra kilęs iš loginės sąvokų santykio, ryšio tarp esybių sampratos.

Dabar galime patikslinti lentelės apibrėžimą: konkreti duomenų lentelė nėra santykis, bet santykio tipo kintamasis.

Formalus santykio apibrėžimas.

Santykis, apibrėžtas domenų (nebūtinai skirtingų) aibėje $\{D_1, D_2, \ldots D_n\}$yra sudarytas iš dviejų dalių: antraštės ir kūno. Aišku, kad sveikasis skaičius *n* yra lygus santykio laipsniui. Jei įsivaizduosime santykį kaip lentelę, tai antraštė atitiks stulpelių vardų eilutę, o kūnas – duomenų eilučių aibę.

1. **Antraštė** yra sudaryta iš nekintamos atributų ir jų domenų porų aibės: $\{(A_1, D_1) (A_2, D_2), \ldots (A_n, D_n)\}$. Kiekvieną atributą A_j atitinka vienas ir tik vienas domenas iš domenų aibės. Visi atributų vardai yra skirtingi.
2. **Kūnas** yra sudarytas iš kortežų. Kortežas yra aibė, kurią sudaro poros $\{(A_1, r_1) (A_2, r_2), \ldots (A_m, r_m)\}$, kur $r_j \in D_j$. Kortežе kiekvienam antraštės atributui tokia pora (*atributo vardas*, *atributo reikšmė*) egzistuoja. Kiekviena atributo reikšmė yra iš jo domeno.

Kortežų skaičius *m* yra santykio kardinalumas, kuris nuolat kinta pridedant ar naikinant lentelės eilutes.

Pavyzdys

Žemėlapiai

Kodas	Teritorija	Žemėlapio pavadinimas	Išleidimo metai
1359	Lietuva	Reljefo žemėlapis	1998
2887	Lietuva	Kraštovaizdžio žemėlapis	2010
1344	Europa	Reljefo žemėlapis	2002
1345	Europa	Reljefo žemėlapis	2012
1349	Europa	Gyventojų žemėlapis	2012

Santykis ***Žemėlapiai*** yra apibrėžtas keturių domenų aibėje. Jis turi antraštę ir kūną, kurį sudaro atributų reikšmės iš tų domenų – penki kortežai. Santykio tipo kintamasis gali turėti skirtingą kortežų skaičių, tačiau antraštė visada yra ta pati.

Taip interpretuojamos lentelės. Tačiaus santykis ir lentelė nėra identiškos sąvokos. Lentelė yra tik viena iš galimų santykio vaizdavimo formų, be to, ji gali įteigti kai kuriuos neteisingus faktus apie santykį, pavyzdžiui, kad kortežai yra sutvarkyti iš viršaus į apačią.

Santykio laipsnis atitinka ryšio kardinalumą – gali būti unarinis, binarinis ir t.t., santykiai. Gali kilti klausimas, ar domenas gali būti laikomas unariniu santykiu (jis atrodo kaip lentelė su vieninteliu stulpeliu). Iš tikrųjų taip nėra, nes domenas statiškas (jame yra visos galimos reikšmės, todėl jis negali būti plečiamas), tuo tarpu santykis gali dinamiškai kisti.

Santykių savybės.

Paminėsime keturias svarbiausias santykių savybes, kurios tiesiogiai neišplaukia iš apibrėžimo.

1. Santykyje nėra vienodų kortežų.
2. Santykyje kortežai nėra sutvarkyti iš viršaus į apačią.
3. Santykyje atributai nėra sutvarkyti iš kairės į dešinę.
4. Visos atributų reikšmės yra atomai.

Panagrinėsime, ką tai reiškia.

1.

Santykyje nėra vienodų kortežų. Tai reiškia, kad santykio kūnas matematiškai yra aibė, kuri pagal apibrėžimą neturi pasikartojančių elementų. Tai iliustruoja faktą, kad lentelė griežtai kalbant nėra santykis (juk lentelėje gali būti vienodų eilučių). Deja, SQL leidžia sukurti lenteles su pasikartojančiais įrašais.

Iš to, kad santykyje nėra vienodų kortežų išplaukia, kad visada egzistuoja pirminis raktas, tiksliau, potencialus pirminis raktas. Nesant vienodų kortežų bent jau visų atributų kartu derinys nusakys pirminį raktą. Praktiškai visų atributų dažniausiai neprireikia, o pirminiam raktui dar keliamas minimalumo reikalavimas, t.y., jei galima atmesti kurią nors rakto dalį nepažeidžiant unikalumo, tai nėra pirminis raktas. Pavyzdžiui, nors {*Kodas, Teritorija*} yra unikalus derinys, jis nėra santykio ***Žemėlapiai*** pirminis raktas, nes galima atmesti jo dalį „Teritorija“ išsaugant rakto unikalumo savybę.

2.
Santykyje kortežai nėra sutvarkyti iš viršaus į apačią. Ši savybė taip pat išplaukia iš to, kad santykio kūnas yra aibė. Aibės matematikoje yra nesutvarkytos, taigi, ir duomenų bazės lentelėse nėra sekos ir pozicinio adresavimo, todėl tokios sąvokos kaip „penktasis kortežas“ ar „paskutinė eilutė“ neturi prasmės.

3.
Santykyje atributai nėra sutvarkyti iš kairės į dešinę. Tai seka iš to, kad santykio antraštė taip pat yra apibrėžta kaip atributų aibė. Negali būti „pirmojo“ ar „paskutinio“ atributo, t.y., į atributą visada yra kreipiamasi pagal vardą, o ne pagal jo numerį ar padėtį. Taip išvengiama galimo atributų informacijos persidengimo.

4.
Visos atributų reikšmės yra nedalomos. Tai išplaukia iš to, kad visi domenai sudaryti tik iš atomų.

Pasikartojimo grupė – tai stulpelis, arba jų derinys, kuriuose yra po keletą reikšmių vienoje eilutėje. Santykyje negali būti pasikartojimo grupių. Jei tenkinama šita sąlyga, santykis vadinamas **norminiu** (pirmosios norminės formos 1NF santykiu). Reliaciniame modelyje visi santykiai yra norminiai, nors matematiškai to nėra reikalaujama. Yra ir daugiau norminių formų, kurias aptarsime vėliau.

Pavyzdys
Turime matematiškai aprašytą santykį Darbuotojai, kuris yra antrojo laipsnio. Jame yra du atributai, kurių pirmasis – AsmKodas, pavyzdžiui, keturių pozicijų simbolių eilutė, o antrasis yra antro laipsnio santykio tipo. Toks santykis reliacinėje duomenų bazėje būtų neįmanomas, nors formaliai teisingas (galima nesunkiai įsitikinti, kad jis tenkina santykio apibrėžimą).

Darbuotojai

AsmKodas	Darbai
0664	ProjNr | Atlyginimas P1 | 1000 P2 | 500
1728	ProjNr | Atlyginimas P1 | 300 P3 | 800
2122	ProjNr | Atlyginimas P3 | 500

Norminame šį santykį taip, kad būtų išsaugota visa turėta informacija:

Darbuotojai

AsmKodas	ProjNr	Atlyginimas
0664	P1	1000
0664	P2	500
1728	P1	300
1728	P3	800
2122	P3	500

Norminimas reikalingas jau vien tam, kad matematiškai 1NF santykis yra paprasčiausios struktūros. Iš tiesų, pirmuoju atveju reikia turėti po du skirtingus operatorius INSERT, UPDATE ir kitoms operacijoms: vienu atveju, kai reikia įtraukti visą įrašą, kitu – kai reikia įtraukti įrašą į antrojo atributo santykį.

Išvardinsime dažniau pasitaikančius santykių tipus.

1. Pagrindinis santykis – tai duomenų bazėje saugomas santykis, kuriam suteiktas vardas.
2. Vaizdas – išvestinis santykis, kuriam priskirtas vardas, tačiau jo duomenys duomenų bazėje nesaugomi, jie gaunami iš pagrindinių santykių duomenų arba apskaičiuojami.
3. Momentinis vaizdas (angl. *snapshot*) – realus periodiškai atnaujinamas santykis-kopija.
4. Tarpinis ar galutinis rezultatas. Tai išvestinis arba apskaičiuojamas bevardis santykis, prireikus išsaugomas kaip vardinis santykis.

Santykiai ir predikatai.

Įmonės darbuotojus aprašantis santykis ***Darbuotojai*** galėtų būti interpretuojamas taip: darbuotojas su apibrėžtu asmens kodu turi nurodytą pavardę ir dirba konkrečiame projekte, už kurį gauna nurodytą atlyginimą, be to, nėra darbuotojų su vienodais asmens kodais.

Tai – predikatas, kurio teisingumas priklauso nuo jo atributų reikšmių. Pavyzdžiui, galima teigti, kad kartografo, kurio asmens kodas yra 32215150000 pavardė vargu ar gali būti „Jonaitytė“, nes Lietuvos piliečių asmens kodai, prasidedantys skaičiumi 3, nurodo vyrišką lytį.

Naujas kortežas, bandomas įtraukti į reliacinę duomenų bazę, turėtų būti priimtas tik tada, kai jis neprieštarauja atitinkamam predikatui. Tačiau kadangi visų įmanomų predikatų RDBVS nurodyti neįmanoma, ji tikrina svarbiausias bendras taisykles.

1. Domenus: ar visos atributų reikšmės yra iš teisingų (palaikomų) domenų .
2. Pirminio rakto unikalumą.
3. Naudotojo nurodytas specifines konkrečiai duomenų bazei vientisumo taisykles.

Dabar galime dar kartą apibrėžti reliacinę duomenų bazę – tai yra duomenų bazė, pateikiama naudotojui kaip norminių santykių rinkinys.

VII.2 RELIACINIŲ DUOMENŲ VIENTISUMO SAMPRATA

Duomenų vientisumo teorija ir atitinkama reliacinio modelio dalis ypač dažnai keitėsi vystantis modeliui. Turbūt tai reiškia, kad ši dalis yra silpniausiai teoriškai pagrįsta. Trumpai apžvelgsime dabartinę situaciją ir vientisumo sampratą.

1. Bet kuriuo metu duomenų bazėje yra saugoma tam tikra informacija – duomenų aibė, daugiau ar mažiau išsamiai aprašanti pasirinktą dalykinę sritį. Laikoma, kad ši duomenų aibė atitinka tikrovę. Šiaip bet kokia duomenų bazės konfigūracija neturi prasmės, jei saugomos reikšmės neatitinka realios situacijos. Tarkime, beprasmiška būtų lentelė, kurioje atsirastų kalnas, kurio aukštis yra –1.

2. Taigi, projektuojant duomenų bazę dar turi būti atsižvelgta į taip vadinamas **vientisumo taisykles**. Šių taisyklių paskirtis yra papildyti duomenų bazę informacija apie realaus pasaulio duomenų apribojimus ir apsaugoti ją nuo neleistinų reikšmių įvedimo. Tai reiškia, kad turi būti kontroliuojamos ir, jei reikia, nutraukiamos UPDATE ir INSERT operacijos.

Dauguma duomenų bazių valdymo sistemų gali palaikyti aibę įvairių vientisumo reikalavimų (juos apibrėžiame kurdami lenteles kartu su laukų tipais). Kiekviena apibrėžta taisyklė yra specifinė konkrečiai duomenų bazei. Tačiau, be specifinių taisyklių, reliaciniame modelyje yra dvi bendrosios vientisumo taisyklės, kurios privalomos visoms duomenų bazėms. Jos aprašo **pirminius** ir **išorinius raktus**, apie kuriuos kalbėsime toliau. Laikysime, kad vientisumo taisyklės galioja tik baziniams santykiams (nors iš tikrųjų ir išvestiniai santykiai turėtų paveldėti nustatytas taisykles; vienintelė problema yra ta, kad tada taisyklės gali tapti prieštaringos).

Potencialūs raktai.

Pirminio rakto (PK – angl. *primary key*) sąvoka jau buvo ne kartą minėta. Negriežtai kalbant, PK yra paprastas unikalus kokio nors duomenų bazės santykio identifikatorius. Tačiau iš tikrųjų pirminio rakto sąvoka yra bendresnės potencialaus rakto (CK – angl. *candidate key*) sąvokos dalis.

Apibrėžimas.
Tegu R yra santykis. CK – to santykio potencialus raktas – tai poaibis R atributų, turintis dvi savybes.

1. Unikalumas (santykyje R nėra dviejų skirtingų kortežų su vienodomis CK reikšmėmis).
2. Neperteklingumas (nė vienas CK poaibis nėra unikalus).

Atkreipsime dėmesį į tai, kad kiekvienas santykis turi bent vieną potencialų raktą, nes negali būti vienodų kortežų pagal santykio apibrėžimą. Blogiausiu atveju, CK tampa visa atributų aibė. Tada galimi du atvejai.

1. Tas derinys neperteklinis (tokiu atveju jis yra vienintelis galimas pirminis raktas).
2. Egzistuoja bent vienas poaibis, turintis abi reikalaujamas savybes.

Gana retai pasitaiko situacija, kai potencialus raktas yra visų atributų derinys (t.y., pats santykis yra potencialus raktas).

CK apibrėžimas liečia ne santykio kintamojo egzempliorių (konkrečias santykio reikšmes), o santykio kintamąjį. Tai yra, turima omenyje ne konkretus santykis, o visos galimos (iš principo įmanomos) reikšmės. Pavyzdžiui, jei sakome, kad derinys {AsmKodas, Pavardė} yra potencialus santykio ***Darbuotojai*** raktas, tai nereiškia, kad tik konkrečiu laiko momentu santykyje nėra dviejų kortežų su vienodomis CK reikšmėmis (t.y., dviejų vienodų faktų, kad darbuotojas su konkrečiu asmens kodu turi konkrečią pavardę); tai reiškia, kad vienodos CK reikšmės yra iš principo neįmanomos pagal duomenų prasmę (t.y., dalykinėje srityje neįmanomas asmens kodo ir pavardės derinių sutapimas).

Jei kalbėsime apie santykio tipo kintamojo egzempliorių, reikia papildyti CK apibrėžimą taip.
Tegu R – santykis. CK – to santykio potencialus raktas – poaibis R atributų, **visada** turintis dvi savybes.

1. Unikalumas (**bet kuriuo konkrečiu momentu** R nėra dviejų skirtingų kortežų su vienodomis CK reikšmėmis).
2. Neperteklingumas (nė vienas CK poaibis nėra unikalus).

Šį potencialaus rakto apibrėžimą ir naudosime.

Nors praktiškai santykis dažniausiai turi tik vieną potencialų raktą, jų gali būti ir keletas. Pavyzdžiui, periodinėje cheminių elementų lentelėje yra bent trys potencialūs raktai: pavadinimas, atominis masės skaičius ir elemento sutartinis žymėjimas.

Potencialus raktas, sudarytas iš daugiau negu vieno atributo, vadinamas sudėtiniu, kitaip – paprastuoju.

Neperteklingumo prasmė. Jei apibrėšime CK, kuris yra perteklinis, sistema to nežinos ir negalės palaikyti vientisumo taisyklių šiam raktui. Pavyzdžiui, jei CK pasirinksime aibę {AsmKodas, PasoNr}, sistema nepalaikys AsmKodas unikalumo atskirai, o tik viso derinio unikalumą.

Yra dar viena rimta priežastis reikalauti neperteklingumo. Tai – išoriniai raktai, su kuriais CK yra siejamas (kitaip nebūtų galima duomenų bazės norminti). Neperteklingumo prasme kartais naudojamas terminas „minimalumas“, bet tai nėra visai teisinga, pavyzdžiui, iš dviejų galimų CK „minimalus“ bus tas, kurį sudaro mažiau atributų, tačiau jie abu gali nebūti pertekliniai.

Negalima painioti potencialaus rakto ir indekso sąvokų, nors daugumoje sistemų CK indeksuojami. Mums tai neturi rūpėti, nes indeksas – tai tik duomenų pasiekimo kelias ir logiškai niekaip nėra susijęs su CK.

Kodėl reikalingi potencialūs raktai.
Potencialių raktų svarba slypi tame, kad jie yra pagrindas mechanizmo, kuriuo adresuojami reliacinės sistemos kortežai. Vienintelis sistemos garantuojamas būdas tiksliai nurodyti kurį nors kortežą – nurodyti atitinkamo potencialaus rakto reikšmę. Pavyzdžiui, įvykdę užklausą

SELECT * FROM Kartografai WHERE AsmKodas = „0664“

garantuotai gausime ne daugiau vieno kortežo. Taip pat bus, jei ieškosime pagal paso numerį. Bet jei sąlyga bus

SELECT * FROM Kartografai WHERE Miestas = „Vilnius“,

įrašų skaičius yra neprognozuojamas.

Taigi, potencialus raktas DB funkcionavimui turi tokią pat reikšmę, kaip atminties adresavimas PC darbui. Jei santykiai neturėtų potencialių raktų, t.y., būtų galimi kortežų dublikatai, būtų labai sunku aptikti nukrypimus nuo užduotų taisyklių.

Pirminiai ir alternatyvūs raktai.
Kaip jau įsitikinome, pagrindinis santykis gali turėti daugiau nei vieną potencialų raktą (nors tai nedažnai pasitaiko). Tuo atveju vienas iš potencialių raktų turi būti pasirinktas pirminiu (PK), o kiti vadinami alternatyviais (AK – angl. *alternative key*). Pavyzdžiui, darbuotojų lentelėje AsmKodas tampa PK, o paso numeris – AK. Jei yra vienintelis CK, jis savaime tampa pirminiu. Turėti vieną pirminį raktą yra patogu, nors didelis trūkumas yra tas, kad sistema bendru atveju negali kontroliuoti potencialių raktų vientisumo ar palaikyti nuorodų į bet kurį potencialų raktą. MS Access yra leistini tik pirminiai raktai.

Išoriniai raktai.
Panagrinėkime santykius ***Miestai***, ***Valstybės*** ir ***Sostinės***.

Miestai

MiestoPavadinimas	GyventojuSkaicius
Vilnius	800 000
Kaunas	400 000
Helsinkis	750 000

Valstybės

ValstPavadinimas	Santvarka
Suomija	Demokratinė respublika
Lietuva	Demokratinė respublika
D.Britanija	Konstitucinė monarchija

Sostinės

MiestoPavadinimas	ValstPavadinimas
Vilnius	Lietuva
Helsinkis	Suomija

Lentelės ***Sostinės*** atributas MiestoPavadinimas gali įgauti kokią nors reikšmę tik tuo atveju, jei tokia pat reikšmė yra kuri nors pirminio rakto reikšmė santykyje ***Miestai***. Taip pat nebūtų prasmės įtraukti į lentelę ***Sostinės*** valstybės, kurios pavadinimo nėra valstybes aprašančiame santykyje. Atributai MiestoPavadinimas ir ValstPavadinimas lentelėje ***Sostinės*** yra pavyzdžiai to, ką toliau vadinsime išoriniu raktu (FK – angl. *foreign key*).

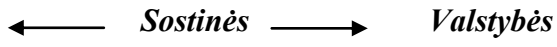
Reliacinis modelis reikalauja, kad išoriniai raktai tiksliai atitiktų būtent pirminius, o ne šiaip bet kokius potencialius raktus.

Apibrėžimas.
Tegu R2 yra pagrindinis santykis. Tada santykio R2 išorinis raktas FK – tai toks santykio atributų poaibis, kad:

1. egzistuoja pagrindinis santykis R1 (kuris nebūtinai yra skirtingas nuo R2) su potencialiu raktu CK;
2. kiekviena FK reikšmė santykio R2 kortežuose bet kuriuo momentu sutampa su kurio nors R1 kortežo CK reikšme.

Pastabos.

1. Pabrėšime, kad išoriniai raktai, kaip ir potencialūs raktai, yra atributų aibės.
2. Pagal apibrėžimą kiekvieną FK reikšmė turi atitikti egzistuojančio PK reikšmę. Tačiau atvirkštinis reikalavimas negalioja: PK, susietas su kuriuo nors FK, gali turėti reikšmių, kurių neatitinka jokia FK reikšmė. Tai atitinka realią situaciją, pavyzdžiui, valstybės sostinė būtinai yra miestas, tačiau gali egzistuoti miestas, kuris nėra jokios valstybės sostinė.
3. FK yra sudėtinis (t.y., sudarytas iš daugiau negu vieno atributo) tada ir tik tada, kai atitinkamas PK yra sudėtinis. FK yra paprastasis tada ir tik tada, kai atitinkamas PK yra paprastasis.
4. Kiekvienas atributas, įeinantis į FK turi būti apibrėžtas tame pačiame domene, kaip ir atitinkamas CK atributas.
5. Išoriniam raktui nėra reikalaujama, kad jis būtų PK ar CK savo santykyje komponentu, nors neretai jis toks būna. Pavyzdžiui, santykyje ***Sostinės*** atributai MiestoPavadinimas ir ValstPavadinimas kartu sudaro PK. Išoriniu raktu gali būti bet koks atributas ar jų junginys.
6. FK reikšmė praktiškai yra nuoroda į kortežą su atitinkama PK reikšme. Problema yra ta, kaip garantuoti, kad duomenų bazėje neatsirastų neteisingų FK reikšmių – tai nuorodų vientisumo problema. Apribojimas, pagal kurį FK reikšmės turi būti adekvačios atitinkamo PK reikšmėms – tai nuorodų vientisumo (angl. *referential integrity*) reikalavimas. Lentelė, į kurios pirminį raktą rodo FK, vadinama tiksline lentele (angl. *target relation*).
7. Diagramos. DB egzistuojančius nuorodų apribojimus patogu vaizduoti diagramomis.



MS Access sistemoje yra galimybė grafiškai nurodyti ryšius. Rodyklė rodo iš FK į atitinkamą PK. Kartais prasminga nurodyti, per kurį būtent atributą santykiai susiejami, ypač jei yra keli CK.

8. Santykis vienu metu gali būti ir tikslinis, ir susietas.

R1* => *R2* => *R3

Toks šiuo atveju yra santykis R2. Nuorodų kelias duomenų bazėje – tai eilutė rodyklių, siejanti santykius.

9. R1 ir R2 iš FK apibrėžimo nebūtinai yra skirtingi, tai yra, santykyje gali būti FK, kurio reikšmės atitinka to paties santykio PK reikšmes

Darbuotojai

AsmKodas	Atlyginimas	Viršininkas
0664	600	0665
0665	400	1123

Pavyzdžiui, lentelėje atributas Viršininkas yra FK, rodantis į tos pačios lentelės PK AsmKodas.

10. Nuorodos į save – tai paprasčiausias atvejis bendros situacijos, kai egzistuoja nuorodų ciklas, t.y., kelias iš santykio į patį save.

R1* => *R2* => …=> *R1

11. Nuorodos ***FK* => *PK*** - tai „klijai“, kurie laiko DB kaip visumą.

Beje, paprastas skirtingų santykių kurių nors atributų reikšmių sutapimas dar nereiškia, kad tie santykiai yra tarpusavyje susieti ar kad atitinkami atributai privalo tapti raktais.

Nuorodų vientisumo taisyklė

Duomenų bazėje neturi būti nesuderintų FK reikšmių, t.y., tokių FK reikšmių, kurioms nebūtų atitinkančios PK reikšmės tikslinio santykio kortеže. Praktiškai tai reiškia, kad jei egzistuoja nuoroda ***A* => *B*** , tai ***B*** privalo egzistuoti.

Išorinių raktų taisyklės

Vientisumo taisyklės taikomos duomenų bazės būsenoms, t.y., yra siekiama, kad darbo metu neatsirastų nekorektiškų būsenų. O kaip jų išvengti, taisyklės nenurodo.

Paprasčiausia yra uždrausti visas operacijas, dėl kurių galėtų atsirasti nekorektiškos reikšmės. Tačiau kartais yra geriau operacijas leisti, o paskui susitvarkyti su pasekmėmis taip, kad nuorodų vientisumas liktų nepažeistas.

Pavyzdžiui, išmetant įrašą apie darbuotoją iš lentelės ***Darbuotojai***, sistema pati gali ištrinti visus su juo susijusius įrašus iš lentelių, kuriose yra išorinis raktas, susietas su šiuo santykiu (pavyzdžiui, lentelės, kuriose saugoma užduočių, kurias atlieka darbuotojai ar skyrių, kuriems darbuotojai vadovauja, informacija), be naudotojo įsikišimo. Žinoma, ne visais atvejais reikia elgtis būtent taip, pavyzdžiui, išėjus iš darbo skyriaus vadovui, nereiškia, kad panaikinamas skyrius. .

Taigi, duomenų bazės administratorius privalo turėti galimybę nurodyti, kokias operacijas uždrausti, kokias leisti, ar reikia leistoms operacijoms kompensavimo veiksmų, o jei taip, tai kokių.

Apžvelgsime esamas galimybes, nors jos ir yra už reliacinio modelio ribų. Idėja tokia: apibrėžiant kiekvieną išorinį raktą reikia atsakyti į du klausimus:

1. Kas turi įvykti bandant išmesti išorinio rakto nuorodos objektą, pavyzdžiui, naikinant valstybę, kurioje yra duomenų bazėje saugomi miestai. Yra bent dvi galimybės:
   a) sustabdyti išmetimo operaciją (RESTRICT) iki momento, kai nebebus nė vieno su naikinamos valstybės įrašu susieto miesto. Kol tokių yra, operacija uždraudžiama.
   b) Išmesti visus susietus miestų kortežus „kaskadomis“ (CASCADES).
2. Kas turi įvykti bandant atnaujinti PK, į kurį yra nuorodų, pavyzdžiui, keičiant pavadinimą valstybės, kurioje yra duomenų bazėje saugomi miestai. Yra analogiškos dvi galimybės:
   a) Uždrausti operaciją iki momento, kai duomenys bus sutvarkyti, t.y., nebebus miestų, kuriems nurodyta, kad jie priklauso keičiamai valstybei (RESTRICT).
   b) Atnaujinti valstybės pavadinimo FK reikšmes visuose susietuose santykiuose (CASCADES).

Iš to, kas pasakyta, aišku, kad susiejant santykius būtina ne tik nurodyti jų FK, bet ir specialias papildomas taisykles kiekvienam FK.

Aišku, kad RESTRICT ir CASCADES pasirinkimai neapima visų įmanomų veiksmų – tai tik du praktikoje dažniausiai naudojami atvejai. Gali būti atliekamos ir kitokios operacijos, pavyzdžiui:

- Inicijuojamas dialogas su naudotoju;
- Informacija ne ištrinama, o perkeliama į archyvą;
- Miestui priskiriama kita valstybė.

Visų veiksmų nusakyti neįmanoma, tačiau jie gali būti užprogramuoti kaip išorinės procedūros, kurias DBVS įvykdo susidarius atitinkamai situacijai.

Tegu R1 ir R2 – du kortežai, susieti raktu R2 => R1. Tegu nuorodai nustatyta kaskadinio trynimo taisyklė, t.y., išmetus kortežą iš santykio R1, bus naikinami ir atitinkami R2 kortežai. Dar tarkime, kad dar egzistuoja nuoroda R3 => R2. Tokiu atveju netiesioginis R2 trynimas bus interpretuojamas kaip tiesioginis, t.y., pagal taisykles, nustatytas R3 išoriniam raktui. Pavyzdžiui, jei R3 kaskadinio trynimo operacija uždrausta, nebus įvykdyta ir visa trynimo operacija ir taip bet kuriam skaičiui lygmenų (rekursija). Analogiškai vykdomas duomenų atnaujinimas.

DB duomenų operacijos visada yra vientisos: arba įvykdoma visa operacija, arba nevykdoma jokia jos dalis (tai svarbu, jei operacija susideda iš kelių etapų kaip kaskadavimo atveju).

Tai, kad gali atsirasti nuorodų ciklai, reiškia, kad kai kurie apribojimų patikrinimai negali būti atlikti atnaujinimo metu, todėl tai turi būti padaryta vėliau ir, jei reikia, operacija atšaukta.

VII.3 NEAPIBRĖŽTOS REIKŠMĖS

Išdėstėme pagrindines CK ir FK idėjas reliaciniame modelyje. Deja, yra vienas apsunkinantis faktorius, kurį iki šiol sąmoningai ignoravome. Tai – neapibrėžtos reikšmės, keliančios daug diskusijų dėl jų sąsajos su trireikšme logika bei specifinio jų naudojimo agreguojančiose funkcijose, SQL grupavimo ir lentelių jungimo operacijose.

Dabar susipažinsime ir su šia samprata. Neapibrėžtos reikšmės yra susijusios su informacijos nebuvimu. Realiame gyvenime ši problema labai dažna, pavyzdžiui, nuolat naudojami išsireiškimai „gimimo data nežinoma“, „pranešėjas bus paskelbtas“, „gyvenamoji vieta nenustatyta“ ir pan. Ir duomenų bazių valdymo sistemose kažkaip reikia į tokią situaciją atsižvelgti.

E. Kodas įvedė specialias žymas nurodyti trūkstamai ar netaikomai informacijai, t.y., taip vadinamas NULL reikšmes, duomenų bazių teorijoje dar žymimas ω. NULL taip pat yra tokioms reikšmėms rezervuotas žodis SQL kalboje. Jei korteže yra tokia reikšmė, tai reiškia, kad atitinkamo

atributo reikšmės nėra, pavyzdžiui, paso numeris nenurodytas, nors žmogus, be abejo, jį turi. Beje, dėl šių dviejų atvejų 1990 m. savo knygoje „The Relational Model for Database Management, Version 2“ Kodas pabrėžė, kad vienintelės NULL reikšmės, kurią palaiko SQL standartas, neužtenka. Pasiūlytos dvi žymos, atitinkamai reiškiančios „nenurodytas, bet galimas“ (angl. *A-Values*) ir nenurodytas ir negalimas“ (angl. *I-Values*) reikšmes. Tačiau įgyvendinti šiai idėjai jau reiktų ne trireikšmės, o keturių reikšmių logikos, todėl ji nebuvo visuotinai priimta.

Neapibrėžtos reikšmės – ne tas pats, kas skaitiniai nuliai, tarpai ar tuščios teksto eilutės. Jos nepriklauso jokiam domenui, t.y., net negali būti laikomos „reikšmėmis“, tik trūkstamos reikšmės žymomis. Todėl palyginimo su neapibrėžta reikšme rezultatas negali būti teisingas ar klaidingas, o neišvengiamai trečioji loginė reikšmė.

Kiekvienam atributui galima leisti arba uždrausti neapibrėžtas reikšmes. Sprendimas priklauso nuo to, kaip buvo apibrėžtas atributas. Čia remiamasi savotiška logika, kurioje atsiranda „trečioji reikšmė“. Daugelis DBVS specialistų mano, kad neapibrėžtos reikšmės formalioje sistemoje yra nesusipratimas ir jų neturėtų būti, nes jų problema iki galo nėra apgalvota ir jokio patenkinamo sprendimo dar nepasiūlyta. Nors yra specialios funkcijos ir predikatai darbui su neapibrėžtomis reikšmėmis, jų priešininkai įsitikinę, kad tokie sprendimai tik be reikalo padidina reliacinio duomenų bazių modelio sudėtingumą ir nevientisumo riziką.

Trireikšmė arba trivalentė logika – tai tokia loginė sistema, kurioje galimos trys teisingumo reikšmės – teisinga, klaidinga ir trečioji reikšmė, kuri gali būti skirtinga priklausomai nuo konkrečios sistemos, tačiau visada skiriasi nuo pirmųjų dviejų, t.y., nėra nei teisinga, nei klaidinga. Tuo trireikšmė logika iš esmės skiriasi nuo mums įprastos dvireikšmės Bulio logikos.

Kaip ir dvireikšmėje logikoje, teisingumo reikšmės gali būti išreikštos skaičiais iš trejetainės sistemos. Keletas dažnų pavyzdžių:

1. 1 – teisinga, 2 – klaidinga, 0 – nežinoma arba nesvarbi reikšmė,
2. 0 – klaidinga, 1 – teisinga, tuo tarpu trečioji reikšmė nėra iš sveikųjų skaičių domeno, pavyzdžiui, tai gali būti ½ , „#“ ar pan.
3. +1 – teisinga, -1 – klaidinga, 0 – trečioji reikšmė.

Taip pat egzistuoja trireikšmė predikatų logika, kuri yra klasikinės logikos plėtinys. Joje kvantorių reikšmės gali būti traktuojamos skirtingai, taip pat gali būti įvesti papildomi ar alternatyvūs kvantoriai.

Galima sudaryti ir trireikšmių elementarių loginių operacijų teisingumo reikšmių lentelę (naudojant trečiajame pavyzdyje parodytą reikšmių kodavimą).

p	**q**	**p ∨ q**	**p & q**	**¬p**
+1	+1	+1	+1	-1
+1	0	+1	0	-1
+1	-1	+1	-1	-1
0	+1	+1	0	0
0	0	0	0	0
0	-1	0	-1	0
-1	+1	+1	-1	1
-1	0	0	-1	1
-1	-1	-1	-1	1

Matome, kad trireikšmėje logikoje galioja saviti logikos dėsniai. Trečiąją reikšmę galima laikyti „dėžute“, kurioje yra viena iš dviejų įprastų (teisinga, klaidinga) reikšmių, o operacijos su šia reikšme pagrįstos galimybe tiksliai žinoti rezultatą pagal kitus operandus. Kai tokios galimybės nėra, rezultatas yra nežinomas – ta pati trečioji reikšmė.

Trireikšmė logika naudojama duomenų bazių SQL kalbos realizacijose tam, kad būtų galima operuoti neapibrėžtomis duomenų bazės laukų reikšmėmis. Pavyzdžiui, jei lauke *Miestas* miesto pavadinimas yra neapibrėžtas, SQL išraiškos *Miestas = 'Vilnius'* rezultatas bus ne klaidingas, o nežinomas, t.y., NULL interpretuojamas taip: miesto pavadinimas su vienoda tikimybe gali būti arba nebūti „Vilnius“. SQL neapibrėžtos reikšmės nėra įvedamos ir nepriskiriamos jokiam duomenų tipui.

Reikia pabrėžti, kad nors SQL naudoja trireikšmę logiką, duomenų operacijose SELECT ir UPDATE nežinoma reikšmė prilyginama klaidingai, t.y., naudotojas suvokia operacijas kaip paklūstančias dvireikšmės logikos dėsniams. Tačiau tikrinant apribojimus, elgiamasi priešingai – nežinomas rezultatas laikomas tenkinančiu apribojimą. Dėl to, kad neapibrėžtos reikšmės visada lemia neapibrėžtą palyginimo operacijos rezultatą, o išrinkimo sakinio sąlygose klaidingos ir neapibrėžtos reikšmės neskiriamos, paprastas žemiau pateiktas SQL sakinys demonstruoja būdingą su NULL susijusią klaidą. Ši užklausa niekada negrąžins nė vienos eilutės, nors stulpelyje *Atributas1* gali būti neapibrėžtų reikšmių.

```
SELECT *
FROM Lentelė
WHERE Atributas1 = NULL;
```

Svarbu žinoti dar keletą NULL interpretavimo RDBVS ypatumų.

- Vykdant INSERT, UPDATE, ir DELETE užklausas vienodai ignoruojamos eilutės, kurioms predikatas suskaičiuoja ir neteisingas, ir neapibrėžtas reikšmes. Jei reikia į tokias reikšmes atsižvelgti, turi būti naudojami specialūs palyginimo operatoriai IS NULL ir IS NOT NULL.
- Jungiant lenteles, kai trūksta susijusių reikšmių, jos automatiškai pakeičiamos NULL reikšmėmis. Ypač atsargiai reikia jungti lenteles kai sąlygoje naudojamas atributas kurio reikšmės gali būti neapibrėžtos. Kadangi NULL nėra lygus jokiam kitam NULL, pagal tokias reikšmes jungiama nebus.
- Agreguojančios funkcijos, tokios kaip suma ar vidurkis, paprastai ignoruoja neapibrėžtas reikšmes. Tačiau skaičiuojant įrašų skaičių (funkcija COUNT), skaičiuojami ir įrašai su neapibrėžtomis reikšmėmis.
- UNION, INTERSECT, EXCEPT operatoriai bei DISTINCT sąlyga interpretuoja visas NULL reikšmes kaip vienodas. Rūšiavimo tvarkos, kai yra NULL reikšmės, SQL standartas aiškiai neapibrėžia, todėl nuo DBVS gamintojo priklauso, ar NULL reikšmės atsidurs sąrašo pradžioje, ar pabaigoje.

Potencialūs raktai ir NULL reikšmės.

Reliacinis modelis reikalauja, kad PK būtų pasirinkta viena iš CK reikšmių. Kartu su PK apibrėžimu modelyje atsiranda ir *objektų vientisumo taisyklė*:

Nė vienas PK elementas negali turėti NULL reikšmių.

Šią taisyklę galima paaiškinti taip.

1. Santykiai atitinka realaus pasaulio esybes. Esybių egzemplioriai yra skirtingi pagal apibrėžimą, t.y., jie turi būti atpažįstami.
2. Reliaciniame modelyje PK atlieka unikalaus identifikatoriaus funkciją ir būtent pagal jį atpažįstama atitinkama esybė.

Tarkime, kad santykyje turime kortežą su NULL reikšme AsmKodas atribute. Tai praktiškai reiškia, kad egzistuoja darbuotojas, kuriam asmens kodas nenurodytas ir jis nėra identifikuojamas. Dar svarbu, kokia yra tikroji NULL prasmė – ar tai tik „nežinoma“, ar apskritai „netaikoma“ savybė. Jei savybė netaikoma, toks kortežas neturi prasmės, nes asmens kodas, kaip unikalus identifikatorius, yra privalomas pagal apibrėžimą. Jei NULL reiškia nežinomą reikšmę, kyla dar daugiau problemų, pavyzdžiui, ar toks kortežas aprašo žinomą darbuotoją, ar tik pageidaujamą kitų savybių rinkinį, ar kiti duomenys yra žinomi, ir t.t. Be to, nebegalime pasinaudodami šiuo atributu suskaičiuoti darbuotojų.

Objektas, neturintis identifikatoriaus neegzistuoja, tai prieštara, paradoksas. Analogiški argumentai galioja bet kuriam sudėtinio PK poaibiui. Jei objektas toks svarbus, kad norime jį išsaugoti duomenų bazėje, jis turi būti tiksliai ir vienareikšmiškai identifikuojamas. Jei to padaryti negalime, tai ir į DB objekto neturėtume įtraukti.

Taigi, vientisumo taisyklė akcentuoja ne tai, kad PK yra unikalūs (ši jų savybė tieiogiai išplaukia iš apibrėžimo), o tai, kad juose nėra neapibrėžtų reikšmių.

- Vientisumo taisyklė taikoma tik pagrindiniams santykiams, Tai blogai, nes iš principo galima sukurti išvestinį objektą ir jį išsaugoti su vardu – pavyzdžiui, į išvestinę darbuotojų lentelę gali būti išrenkami tik pasų numeriai, tuo tarpu jiems buvo leista būti NULL, nes šis atributas netapo PK baziniame santykyje, nors jis neabejotinai turėtų tapti PK naujame išvestiniame santykyje.
- Vientisumo taisyklė taikoma tik pirminiams raktams. Alternatyviems raktams ji neprivaloma ir nurodoma pagal pasirinkimą. Bet jei alternatyvus raktas yra toks, kad galimos NULL reikšmės, jis jau nebegali tapti PK. Tai vėl nelogiška, iš tiesų vientisumo taisyklė turėtų galioti visiems CK.

Kyla klausimas, ar apskritai logiškai galimos NULL reikšmės. Kas atsitinka, jei visi kortežo atributai turi NULL reikšmes?

Galima atsisakyti NULL reikšmių, o vietoje jų naudoti numatytasias (angl. *default*) sutartines reikšmes. Praktiškai taip dažnai ir daroma, pavyzdžiui, nežinomą atlyginimą būtų galima koduoti 0, vardą – „???“. Taip būtų išvengta minėtų problemų, be to pagal sutartines reikšmes galima ieškoti, išrinkti ir atlikti kai kurias kitas operacijas. Tačiau kiltų problema, kaip atskirti tikrąsias ir sutartines reikšmes (jos privalo būti iš to paties domeno) kai skaičiuojamos sumos, vidutinės reikšmės ar atliekamos kitos apibendrinančios operacijos.

Išoriniai raktai ir NULL reikšmės.

Tarkime, kad lentelėje ***Darbuotojai*** yra darbuotojas, nedirbantis jokiame projekte (ProjNr reikšmė turėtų būti NULL). Tai logiška ir dažnai pasitaikanti situacija, todėl išoriniuose raktuose NULL reikšmės nėra uždraustos. Dabar galima papildyti FK apibrėžimą:

Tegu R2 yra pagrindinis santykis. Tada santykio R2 išorinis raktas FK – toks santykio atributų poaibis, kad:

1. egzistuoja pagrindinis santykis R1 (kuris nebūtinai yra skirtingas nuo R2) su potencialiu raktu CK;
2. Kiekviena FK reikšmė R2 bet kuriuo momentu **arba** sutampa su kurio nors R1 kortežo CK reikšme **arba yra NULL**.

Taip nepažeidžiama taisyklė, kad kiekvieną FK reikšmę atitinka PK reikšmė (vientisumo taisyklė).

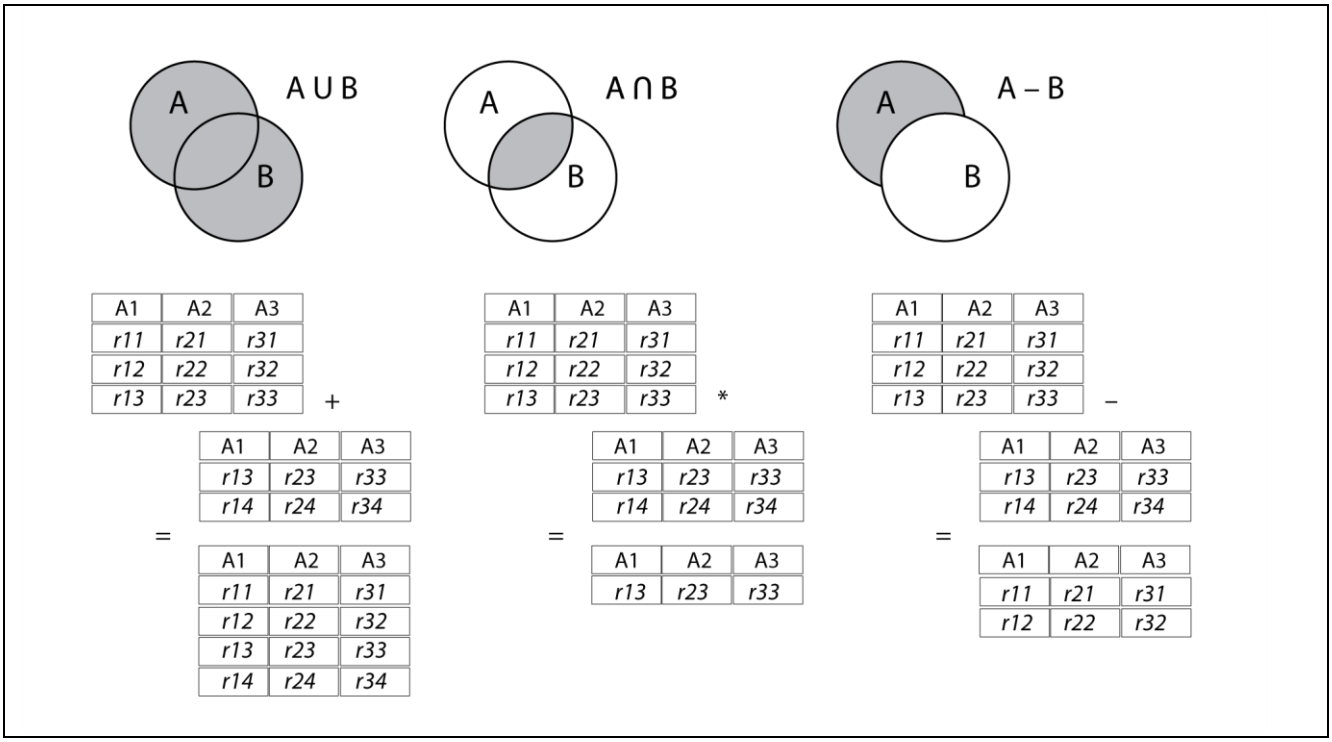
Atsižvelgiant į NULL reikšmes galima nurodyti dar vieną vientisumo palaikymo metodą:

1. Leisti arba uždrausti NULL reikšmes FK arba jo dalims.
2. Jei FK leistinos NULL reikšmės, turi būti dar viena galimybė išmetant arba keičiant su juo susietą PK reikšmę.

NULLIFY (padaryti neapibrėžtu) metodo pasirinkimas reiškia, kad išmetus ar pakeitus susieto PK reikšmes, atitinkamos FK reikšmės paverčiamos NULL.
Tačiau iš principo NULLIFY metodą visada galima pakeisti reikšmių pagal nutylėjimą įrašymu.

VII.4 RELIACINĖ ALGEBRA IR RELIACINIS SKAIČIAVIMAS

Reliacinė algebra.
E.Kodas apibrėžė „pradinę“ reliacinę algebrą su aštuonių pagrindinių operacijų rinkiniu. Iš tiesų operacijų gali būti daugiau, svarbu tik tai, kad ir operandai ir rezultatai būtų santykiai. Čia panagrinėsime aštuonias pagrindines reliacines operacijas.



VII-1 pav. Aibių sąjungos, sankirtos ir skirtumo operacijos

1. **Sąjunga** (įprasta aibių sąjunga)

Santykių sąjungą sudaro visų kortežų, kurie priklauso jungiamiems santykiams, aibė. Reikia pastebėti, kad, jei jungiamų santykių (operandų) kortežuose esančios reikšmės yra iš skirtingų domenų, arba jei skiriasi atributų skaičius, tokia sąjunga nebus santykis. Todėl reikalaujama, kad operandai būtų vienodos formos, t.y., visiškai suderinami pagal tipą.

1) Kiekvienas iš jungiamų santykių turi turėti tą pačią atributų aibę. Tai reiškia, kad ir santykių laipsniai sutampa.
2) Visi atitinkami atributai, t.y., abiejų santykių atributai su vienodais vardais, turi būti apibrėžti tuose pačiuose domenuose.

Sąjunga yra tokios pačios struktūros santykis, kaip abu operandai. Vienodų kortežų jame nėra.

2. **Sankirta**.

Santykių sankirta – tai visų kortežų, kurie priklauso abiems jungiamiems santykiams, aibė. Kaip ir sąjungos atveju, reikalaujama, kad operandai būtų to paties tipo, t.y., vienodos struktūros santykiai.

3. **Atimtis**.

Santykių skirtumas – tai visų kortežų, kurie priklauso pirmajam santykiui ir nepriklauso antrajam, aibė. Ir šiuo atveju operacija galima tik su to paties tipo santykiais; to paties tipo yra ir rezultatas (VII-1 pav.).

4. **Dekarto daugyba**.

Pagal aibių algebros taisykles, dviejų kortežų Dekarto sandauga turėtų būti kortežų porų aibė, kur pirmasis narys yra iš pirmojo santykio, o antrasis – iš antrojo. Reliacinėje algebroje tai ne pora, o vienas kortežas, sudarytas iš abiejų sujungtų kortežų atributų reikšmių. Jei santykiuose yra atributų vienodais vardais, jie turi būti pervadinti.

Rezultatas turi tiek atributų, kiek jų buvo abiejuose operanduose, kortežų skaičius lygus galimų abiejų operandų eilučių derinių skaičiui:

$$\text{card } A \times B = \text{card } A * \text{card } B;$$
$$\deg A \times B = \deg A + \deg B.$$

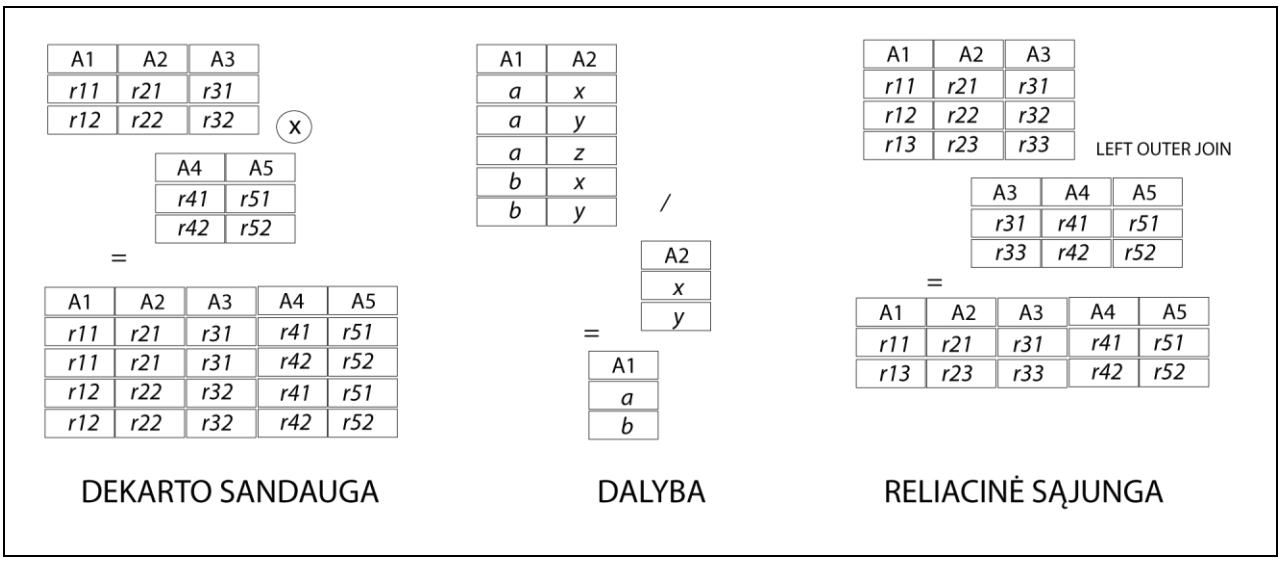
Dekarto daugyba taikoma tik ypač sudėtingoms užklausoms – apskritai ši operacija jokios naujos informacijos nesuteikia. Tačiau Dekarto sandauga dažnai būna netinkamai užrašytos ar praleistos instrukcijos WHERE užklausoje SQL kalba rezultatas. Pavyzdžiui, įvykdžius užklausą

```
SELECT Žmonės.*, Automobiliai.*
FROM Žmonės, Automobiliai;
```

Su praleista sąlyga “WHERE Automobiliai.Savininkas = Žmonės.ID” ,

bus sukurta Dekarto sandauga, kurioje vietoje norimų eilučių, siejančių automobilių informaciją su jų savininkų informacija, bus pateiktos tiesiog visos galimos žmonių ir automobilių poros, nepriklausomai nuo to, ar konkretus žmogus yra jo eilutėje esančio automobilio savininkas, ar ne.

Pirmosios keturios operacijos – tai įprastos aibių operacijos, pritaikytos santykiams. Kitos keturios operacijos yra specifinės, jas galima atlikti tik su santykiais.



VII-2 pav. Aibių Dekarto sandaugos, dalybos ir reliacinės sąjungos operacijos

5. **Išrinkimas**.

Tai jau anksčiau aptarta reliacinė operacija RESTRICT/SELECT, turinti vienintelį operandą. Ją įvykdžius išrenkamas santykio kortežų poaibis.

6. **Projekcija**.

Tai jau anksčiau aptarta reliacinė operacija PROJECT, taip pat turinti vienintelį operandą. Jos rezultatas – išrinktas santykio atributų poaibis.

7. **Dalyba**

Santykio A dalybos iš santykio B rezultatas – tai santykio A atributų poaibio X reikšmės, tokios, kad joms atitinkančios santykio A atributų poaibio Y reikšmės apima visas santykio B reikšmes. Tokį veiksmą atlikti galima tik tada, kai santykyje A yra visi santykio B atributai, t.y., B yra A poaibis. Jo rezultatas – santykio A atributų aibė, nesutampanti su santykio B atributais:

deg A/B = deg A – deg B.

Dalyba yra atvirkštinis Dekarto sandaugai veiksmas.

8. **Reliacinė sąjunga** (sąjunga, susiejanti aibių elementus pagal vienodas reikšmes, dar vadinama natūralia aibių sąjunga).

Du santykiai sujungiami pagal jiems bendro atributo vienodas reikšmes. Žinoma, toks bendras atributas turi būti, o rezultate jis įtraukiamas tik vieną kartą:

deg A×B = deg A + deg B.

Tuo reliacinė sąjunga skiriasi nuo Dekarto sandaugos, į kurią ji yra panaši. Reliacinės sąjungos rezultato kardinalumas yra lygus skaičiui kortežų, kurie turi sutampančias reikšmes abiejuose operanduose.

Galimas ir kitoks, nesimetrinis, reliacinės sąjungos variantas, kai iš vieno santykio imami visi kortežai, prie jų prijungiami antrojo santykio kortežai, turintys sutampančias bendro atributo reikšmes, o jei tokių nėra – prijungiami antrojo santykio atributų kortežai su neapibrėžtomis reikšmėmis juose. Tokioje sąjungoje kortežų skaičius lygus pirmojo santykio kortežų skaičiui.

Algebra yra reikalinga dėl dviejų priežasčių.

1. Reikia, kad būtų galima vienodai ir aiškiai aprašyti operacijas. Tai galima padaryti tik naudojant aukšto abstrakcijos lygmens simbolius.
2. Reliacinė algebra yra optimizavimo pagrindas (optimizuojant nurodoma, kokia reliacinės algebros operacija po kokios seka).

Vystant reliacines sistemas, prie aštuonių pagrindinių vėliau pridėtos ir kitos operacijos.

1. **Išplėtimas**. Tai operacija, naudojama apskaičiuojant papildomų atributų reikšmes.
Pavyzdžiui, jei duomenų bazėje saugomas atlyginimo koeficientas, o dar reikia ir viso apskaičiuoto atlyginimo, pasinaudojame šia operacija:

EXTEND Darbuotojai ADD AtlKoef *200 AS Atlyginimas

2. **Agregavimas**. Tai operacija, leidžianti panaudoti agreguojančias funkcijas, tokias kaip suma, vidurkis, įrašų skaičius, minimali ir maksimali reikšmė.
Pavyzdžiui, galime suskaičiuoti bendrą visų darbuotojų atlyginimą:

SUMMARIZE Darbuotojai BY Atlyginimas ADD COUNT (Atlyginimas) AS Skaicius

3. **Atnaujinimo operacijos**.Tai operacijos INSERT, UPDATE ir DELETE, kurių pagalba keičiamos, naikinamos ar įterpiamos reikšmės.

Reliacinės operacijos yra užklausų, apie kurias rašoma VIII skyruje, vykdymo pagrindas.

Reliacinis skaičiavimas yra loginis reliacinės algebros atitikmuo.

Pavyzdžiui, reikia gauti pavadinimus skyriaus, atsakingų už planavimą (VII-3 pav.).

Projektai

ProjNr	Projektas	SkNr
P1	Administravimas	S1
P2	Mokymai	
P3	Planavimas	S2
P4	Inovacijos	S5

Skyriai

SkNr	Pavadinimas
S1	Administracija
S2	Planavimo skyrius
S3	Rinkodaros skyrius
S4	Technologijų skyrius
S5	Plėtros skyrius

VII-3 pav. Lentelės, aprašančios projektus ir skyrius

Remdamiesi reliacine algebra, turėtume atlikti tokias operacijas:
1. Reliacinė sąjunga lentelių ***Projektai*** ir ***Skyriai*** pagal SkNr.
2. Išrinkimas rezultate kortežų, kuriuose ProjNr lygus „P3“.
3. Rezultato projekcija į atributus Projektas ir SkNr.

Reliaciniame skaičiavime tai išreiškiama sakiniu:
Gauti atributus Projektas iš lentelės ***Projektai*** ir SkNr iš lentelės ***Skyriai***, kuriems egzistuoja įrašas lentelėje ***Skyriai*** su sutampančia SkNr reikšme bei ProjNr reikšme lygia „P3“.

Abi operacijos yra lygiavertės. Kiekvienai algebrinei operacijai egzistuoja ekvivalenti operacija reliaciniame skaičiavime ir atvirkščiai. Aibės ir predikatai taip pat yra susiję vienareikšmiškai.

Reliaciniame skaičiavime naudojami loginiai operatoriai AND, OR, NOT ir egzistavimo bei tarpusavyje priklausomi visuotinumo kvantoriai EXISTS ir FORALL. Jų naudojimo pavyzdžiai pateikiami VIII skyruje.

Ne viską paprasta padaryti naudojantis algebra ar reliaciniu skaičiavimu, pavyzdžiui, išrinkti tris daugiausiai uždirbančius darbuotojus – tokio tipo uždaviniai yra labai sudėtingas. Kai kuriose sistemose jiems net yra sukurtas specialus operatorius.

Klausimai diskusijai

Aptarkite neapibrėžtų reikšmių poreikį ir trireikšmės logikos taikymo praktinėms reikmėms privalumus bei alternatyvas.
Paaiškinkite, kuo nuorodų diagramos skiriasi nuo ryšių diagramų.

Užduotys savarankiškam darbui

Patikrinkite, ar MS Access leidžia saugoti lentelėje vienodas eilutes. Pabandykite, kaip veikia automatinis numeravimas ir pagalvokite, kam jis reikalingas.

Duota informacija apie aukštąsias universitetines mokyklas Lietuvos miestuose:
-- mokyklos pavadinimas, tipas, įkūrimo metai, įkūrėjas (neprivalomas);
-- įvykę istoriniai pasikeitimai: pasikeitę pavadinimai, išnykimas, skilimas į keletą, persikėlimas į kitą miestą, filialo įkūrimas.
Sudarykite tokio teminio žemėlapio duomenų bazės schemą: aprašykite santykius, atributus ir jų domenus. Sudarykite ryšių diagramą.

Duotos tokios lentelės, aprašančios apie universitete dėstomus kursus. Skliausteliuose patekti stulpelių vardai:
-- Kursai (kurso numeris, pavadinimas);
-- Prerekvizitai (kursas, jam klausyti būtinas išklausytas kursas);
-- Paskaitos (kurso numeris, pasiūlymo numeris, data, vieta);
-- Dėstymas (kurso numeris, pasiūlymo numeris, darbuotojo ID);
-- Dėstytojų registracija (kurso numeris, pasiūlymo numeris, darbuotojo ID, studijų programa);
-- Darbuotojai (darbuotojo ID, pavardė, pareigos).

Sudarykite šią situaciją atitinkančią duomenų bazės nuorodų diagramą.

Užduotys praktikos darbams

Panagrinėkite, kokie yra potencialūs, pirminiai, alternatyvūs ir išoriniai raktai ankstesnėms užduotims sukurtose duomenų bazėse. Nurodykite pirminius ir išorinius raktus giminės medžio duomenų bazėje. Sukurkite ryšius tarp lentelių ir apibrėžkite vientisumo palaikymo metodus kiekvienam išoriniam raktui. Įsitikinkite, kad domenų apribojimai ir vientisumo taisyklės yra tinkamai taikomi įvedant naujus duomenis. Sukurkite nuorodų diagramas.

VIII. STRUKTŪRIZUOTA UŽKLAUSŲ KALBA

Bet kas pradės veikti, jei pakankamai ilgai prie to krapštysitės.
Wyszkowski taisyklė

VIII.1 SQL SAVYBĖS

Struktūrizuota užklausų kalba (SQL; angl. *Structured Query Language*) yra kalba, specialiai sukurta tvarkyti duomenims RDBVS. Ji remiasi reliacine algebra ir reliaciniu skaičiavimu ir apima duomenų paiešką, įterpimą, keitimą, šalinimą, DB struktūros kūrimą ir keitimą bei prieigos prie duomenų kontrolę. Nors SQL yra gerokai nutolusi nuo E. Kodo teorinio reliacinio modelio, būtent ji yra pirmoji reliaciniam modeliui sukurta ir šiuo metu labiausiai paplitusi DBVS kalba.

SQL sukūrė *IBM Research* mokslininkai Donaldas Čemberlenas (Donald D. Chamberlin) ir Reimondas Boisas (Raymond Boyce) apie 1970 m. Tada ji vadinta SEQUEL (angl. *Structured English Query Language*) ir naudota tik IBM sukurtoje sistemoje *System R*. 1986 m. SQL tapo ANSI (angl. *American National Standards Institute*), o nuo 1987 m. ISO (angl. *International Standardisation Organization*) standartu. Nuo tada standartas buvo keletą kartų papildytas, tačiau ir dabar yra skirtumų tarp SQL kodo skirtinguose RDBVS produktuose, kurie atsiranda dėl nevisiško atitikimo standartui ar dėl skirtingų standarto interpretacijų.

Dauguma šiuo metu paplitusių reliacinių duomenų bazių valdymo sistemų palaiko kurį nors SQL programavimo kalbos variantą reliacinėms operacijoms aprašyti. SQL yra sudėtinga kalba, jos standartas yra daugiau kaip 600 puslapių dokumentas. Čia aptarsime tik pagrindinius šios kalbos aspektus, be to, kalbėdami apie SQL nuo reliacinių terminų vėl grįšime prie įprastų: lentelė, stulpelis, eilutė.
Svarbiausi SQL kalbos bruožai yra šie:

1. SQL palaikomos duomenų įterpimo (INSERT), duomenų atnaujinimo (UPDATE) ir naikinimo (DELETE) operacijos.
2. SQL palaikomas standartinis katalogas, vadinamas informacine DB schema.
3. SQL operatorius galima iškviesti tiesiogiai (MS Access – tai galimybė sukurti užklausą kaip *Query* objektą) arba iš taikomosios programos (pavyzdžiui, panaudojant *Access Basic* DoCmd.RunSQL metodą).

SQL kalboje išskiriamos tokios kalbos elementų grupės:

- **užklausos** (angl. *query*), kurios išrenka duomenis pagal nurodytus kriterijus, ir yra svarbiausias SQL elementas;
- **sąlygos arba instrukcijos** (angl. *clause*), kurios naudojamos teiginiams ir užklausoms sudaryti;
- **išraiškos** (angl. *expression*), kurių rezultatas yra skaitinės reikšmės arba lentelės;
- **predikatai** (angl. *predicate*), aprašantys reikalavimus, kuriuos turi atitikti duomenys, ir kurių teisingumo reikšmes galima nustatyti;
- **sakiniai** (angl. *statement*), kuriais vykdomos operacijos su duomenų įrašais ar DB struktūra bei kontroliuojamas programų vykdymas. Tarpai SQL sakiniuose ignoruojami.

SQL sakinio gale visada dedamas kabliataškis, kuris reiškia, kad sakinys baigtas ir paruoštas vykdyti. Tarpai SQL sakiniuose ignoruojami.

Kaip jau minėta, SQL ne visiškai atitinka reliacinio modelio logiką. Ji kritikuojama ir dėl kitų priežasčių:

- nepakankamas suderinamumas tarp skirtingų gamintojų ir daugeliu atveju neatitikimas standartui, ypač skirtumai dėl datos ir laiko naudojimo sintaksės, tekstinių eilučių jungimo, neapibrėžtų reikšmių bei didžiųjų ir mažųjų raidžių traktavimo;
- didelė tikimybė nenurodžius WHERE sąlygos predikato gauti Dekarto sąjungą, t.y., visus galimus jungiamų eilučių derinius (to reikia taip retai, kad Dekarto sąjungai sukurti 1992 m. įvesta speciali instrukcija CROSS JOIN);
- keičiant ar trinant įrašus taip pat nesunku praleisti WHERE sąlygą, dėl ko bus pakeista ar ištrinta daugiau eilučių, negu reikia.

Alternatyva SQL gali būti QBE (angl. *Query by Example*) – turinti grafinę naudotojo sąsają ir *IBM Research* kurta lygiagrečiai su SQL užklausų kalba. Naudotojo nurodyti atributai ir sąlygos interpretuojami ir duomenų manipuliavimo sakinys suformuojamas automatiškai, todėl naudotojui nebūtina įvedinėti rankomis vardų bei prisiminti (ir suprasti) SQL sintaksės. Microsoft Access, kaip ir daugelis reliacinių DBVS, palaiko QBE.

VIII.2 DUOMENŲ APIBRĖŽIMAS

Panagrinėsime, kaip sukuriamos pagrindinės lentelės. Deja, skirtingai nuo reliacinio modelio, SQL leidžia būti vienodoms eilutėms lentelėse, todėl formaliai potencialus raktas nebūtinai turi egzistuoti, be to, atributai yra sutvarkyti iš kairės į dešinę, t.y., pirmas, antras ir t.t. stulpeliai. Pagrindinė lentelė yra sukuriama sakiniu CREATE.

```
CREATE TABLE Žemėlapiai
(AsmKodas text NOT NULL, ProjNr text NOT NULL,  ZNr Integer NOT NULL, ZTema Text
NOT NULL, ZTeritorija Text NOT NULL,
PRIMARY KEY (ZNr),
FOREIGN KEY (AsmKodas) REFERENCES Darbuotojai,
FOREIGN KEY (ProjNr) REFERENCES Projektai
);
```

Visada turi būti nurodytas kuriamos lentelės vardas, po jo skliausteliuose nurodomi kableliu atskirti stulpelių apibrėžimai: <atributo vardas> <atributo tipas> <papildomi nurodymai>. Galima iš karto nurodyti, kuris atributas taps pirminiu raktu. Jei aprašomas ir išorinis raktas, būtina po rezervuoto žodžio REFERENCES įrašyti susietos lentelės vardą. Papildomi nurodymai prie kuriamo atributo yra susiję su išorinių raktų vientisumo taisyklėmis ir domenais.

Įvykis	**Galimi pasirinkimai susietų įrašų šalinimo arba keitimo atveju**			
ON DELETE	SET DEFAULT (priskiriama reikšmė pagal nutylėjimą)	CASCADE (šalinami ar keičiami visi susieti įrašai)	NO ACTION (neleidžiama nieko daryti)	SET NULL (priskiriama neapibrėžta reikšmė)
ON UPDATE	SET DEFAULT	CASCADE	NO ACTION	SET NULL

```
FOREIGN KEY (AsmKodas) REFERENCES Darbuotojai ON DELETE SET DEFAULT ON
UPDATE CASCADE;
```

Domeno apribojimai išvardinami po rezervuoto žodžio CHECK.
CHECK (AsmKodas > 0).

Užklausomis galima ir modifikuoti jau sukurtas lenteles: gali būti pridėti nauji stulpeliai arba išmesti esami, pridėti ar išmesti vientisumo apribojimai, sunaikinta pati lentelė.

ALTER TABLE Žemėlapiai
ADD COLUMN Uzsakovas Text;

ALTER TABLE Žemėlapiai
DROP COLUMN Uzsakovas;

DROP TABLE Žemėlapiai;

VIII.3 DUOMENŲ IŠRINKIMO OPERACIJOS IR FUNKCIJOS

Duomenų išrinkimas yra dažniausiai naudojama SQL operacija, vykdoma SELECT sakiniu, kuris surenka duomenis iš vienos ar daugiau nurodytų lentelių ir išraiškų. Įprastas SELECT sakinys niekaip nekeičia duomenų bazės. Juo naudotojas nurodo, kokių duomenų pageidauja, o DBVS atlieka planavimą, optimizavimą ir konkrečius veiksmus fiziniu lygmeniu, kurie būtini rezultatui gauti ir pateikti.

Reliacinio modelio PROJECT ir JOIN operacijos SQL atliekamos taip pat naudojant SELECT sakinį. Dėl SELECT universalumo, kad nebūtų painiavos, paprastas eilučių išrinkimo sakinys kartais vadinamas RESTRICT.

Bendras **SELECT** sakinio pavidalas yra

SELECT [DISTINCT] <atributų sąrašas, atskirti kableliais>
FROM <lentelių sąrašas, atskirtos kableliais>
WHERE <sąrašas predikatų, sujungtų loginiais operatoriais>
[GROUP BY <atributų sąrašas> [HAVING <sąlyga>]]
[ORDER BY <atributų sąrašas| išraiška – ASC | DESC>];

Atkreipkite dėmesį, kad panaudojus modifikatorių DISTINCT iš rezultato lentelės bus išmetamos pasikartojančios eilutės. Pavyzdžiui, užklausos „SELECT DISTINCT Amžius FROM Darbuotojai;“ rezultatas bus eilutės, kuriose yra visos skirtingos darbuotojų amžiaus reikšmės. Beje, šis modifikatorius visas NULL reikšmes laiko vienodomis.

Atributai (stulpeliai) nurodomi užrašant lentelės vardą ir atributo vardą (lentelė.atributas). Jei užklausoje kreipiamasi į vienintelę lentelę arba visi išrenkamų atributų pavadinimai visose lentelėse yra skirtingi, užtenka nurodyti tik atributo vardą. Vietoje atributo pavadinimo gali būti panaudota ir kokia nors konstanta, tada rezultate bus pateiktas stulpelis, kurio reikšmės bus prilygintos konstantai. Jei vietoje atributų sąrašo nurodyta žvaigždutė (*), tai reiškia, kad užklausos rezultatas turi būti visi joje panaudotų lentelių stulpeliai.

Sąlyga FROM nurodo, iš kokių lentelių išrenkami duomenys. Jos dalis gali būti JOIN sąlyga, aprašanti lentelių jungimo taisykles.

Sąlygoje WHERE visada yra palyginimo predikatas, kuris riboja grąžinamų eilučių skaičių, t.y., atmeta eilutes, kurioms predikato teisingumo reikšmė nėra "tiesa" (TRUE). Šios sąlygos galima nenurodyti, tada išrenkamos visos eilutės. Jei SELECT sakinyje yra sąlyga WHERE, ji visada rašoma po sąlygos FROM.

ORDER BY sąlyga nurodo, kaip reikia rūšiuoja įrašus rezultato lentelėje. Tai daroma pagal nurodytų atributų reikšmes eilės tvarka. Kiekvienam atributui nurodoma rūšiavimo tvarka: ASC (angl. *ascending* – didėjantis) – nuo mažiausios iki didžiausios reikšmės; arba DESC (angl. *descending* – mažėjantis) – nuo didžiausios iki mažiausios reikšmės. Nenurodžius rūšiavimo tvarkos bus automatiškai taikomas modifikatorius ASC. Galima rūšiuoti pagal keletą atributų. Pavyzdžiui, įvykdžius užklausą :

SELECT Kodas, Pavardė, Amžius, Atlyginimas
FROM Darbuotojai
WHERE Skyrius = "Administracija" ORDER BY Atlyginimas, Amžius DESC;

Darbuotojų sąrašas bus išrūšiuotas pagal gaunamą atlyginimą didėjimo tvarka, o vienodą atlyginimą gaunantys darbuotojai – nuo vyriausio iki jauniausio.

Šios sąlygos SELECT sakinyje gali nebūti, tada rezultato lentelėje eilutės pateikiamos atsitiktine tvarka.

GROUP BY – tai loginio grupavimo pagal vienodas reikšmes nurodytame stulpelyje sąlyga. Taip išrenkama mažiau eilučių, negu nurodyta sąlygoje WHERE, kuri visada eina prieš GROUP BY. Kiekvienoje grupėje po instrukcijos HAVING skaičiuojama išraiška privalo įgauti vienintelę reikšmę. Formuojant logines išraiškas gali būti panaudoti IN, BETWEEN, LIKE, IS [NOT] NULL, AND, OR, NOT, EXISTS ir kiti operatoriai.

Užklausos sąlygos išvardinamos po žodžio WHERE ir jungiamos loginiais operatoriais AND (konjunkcija), OR (disjunkcija), NOT (neiginys). Apie logines operacijas plačiau rašoma XII.3 skyriuje. Sąlygose nurodomi reikalavimai išrenkamoms atributų reikšmėms. Jie išreiškiami palyginimo operatoriais, išvardintais lentelėje.

VIII-1 lentelė. SQL palyginimo operatoriai

Operatorius	**Reikšmė**	**Sąlygos pavyzdys**
=	Lygu	Darbuotojai.Vardas = "Jonas"
>	Daugiau	Darbuotojai.GimimoMetai > 1970
<	Mažiau	Darbuotojai.Atlyginimas < 1250
>=	Daugiau arba lygu	Projektai.Trukmė >=6
<=	Mažiau arba lygu	Projektai.Trukmė <=12
<> arba !=	Nelygu	Darbuotojai.Pareigos <> "Direktorius"
LIKE	Panašumo testo operatorius	Darbuotojai.Pareigos LIKE 'Vyr%'

Operatoriuje LIKE naudojami šablonai "%", reiškiantys bet kokius teksto simbolius, atsirandančius prieš arba už nurodytų simbolių. Pagal sąlygą lentelėje pateiktame pavyzdyje bus išrinkti

darbuotojai, kurių pareigos prasideda “Vyr.” – Vyr. kartografas, Vyr. projektų vadovas ir pan. Lyginamas tekstas rašomas viengubose kabutėse.

Operatoriai IN ir BETWEEN yra naudojami nurodyti tinkamoms atributų reikšmių aibėms, kad nereiktų daug kartų kartoti AND ir OR operatorių. Jų sintaksė:

IN (<galimų reikšmių sąrašas, atskirtos kableliais>)
BETWEEN <mažiausia reikšmė> AND <didžiausia reikšmė> (kraštinės reikšmės įtraukiamos).

Pavyzdžiai:
SELECT * FROM Projektai WHERE Projektai.Trukmė IN (6, 8 10);
SELECT * FROM Projektai WHERE Projektai.Trukmė NOT IN (7, 9, 11);
SELECT Kodas, Pavadinimas FROM Prekės WHERE Prekė.Sezonas IN (‘Pavasaris’, ‘Vasara’, ‘Ruduo’);
SELECT * FROM Projektai WHERE Projektai.Trukmė BETWEEN 6 AND 10;
SELECT * FROM Projektai WHERE Projektai.Trukmė NOT BETWEEN 3 AND 6;

VIII-2 lentelė. SQL agreguojančios funkcijos

Funkcija	**Prasmė**	**Sąlygos pavyzdys**
MIN	Mažiausia atributo reikšmė	SELECT MIN (Amžius) FROM Darbuotojai;
MAX	Didžiausia atributo reikšmė	SELECT MAX (Trukmė) FROM Projektai;
SUM	Atributo reikšmių suma	SELECT SUM (Atlyginimas) FROM Darbuotojai;
AVG	Atributo reikšmių vidurkis	SELECT AVG (Trukmė) FROM Projektai;
COUNT	Atributo reikšmių skaičius	SELECT COUNT (Pavardė) FROM Darbuotojai;
COUNT(*)	Įrašų skaičius lentelėje	SELECT COUNT (*) FROM Darbuotojai;

Išrenkant galima naudoti VIII-2 lentelėje išvardintas agreguojančias funkcijas nurodytam atributo stulpeliui. Išimtis – funkcija COUNT(*), kuri nenaudoja konkrečių atributų. Funkcijos SUM ir AVG taikomos tik skaitinėms atributų reikšmėms.

Agreguojančios funkcijos dažniausiai naudojamos su GROUP BY sąlyga. Tarkime, norint gauti kiekvieno skyriaus didžiausių atlyginimų sąrašą, vykdytume užklausą “SELECT Skyrius, MAX (Atlyginimas) FROM Darbuotojai GROUP BY Skyrius;”.

Sąlyga HAVING leidžia nurodyti sąlygas kiekvienos grupės eilutėms, t.y., kurios eilutės turi būti išrinktos pagal nurodytas sąlygas. Tarkime mus domina tik skyriai, kuriuose vidutinis atlyginimas yra daugiau kaip 2000: “SELECT Skyrius, AVG (Atlyginimas) FROM Darbuotojai GROUP BY Skyrius HAVING AVG (Atlyginimas) > 2000;”.

Matematinės funkcijos

SQL palaiko keturias pagrindines aritmetines operacijas:

- sudėtis (+),
- atimtis (-),
- daugyba (*),
- dalyba (/).

Dauguma DBVS palaiko ir nestandartinę sveikosios skaičiaus dalies išskyrimo operaciją (%).

Matematinės funkcijos standarte neapibrėžtos, todėl priklausomai nuo RDBVS, jų gali būti daugiau ar mažiau, taip pat skirtis jų pavadinimai. VIII-3 lentelėje išvardintos pagrindinės, plačiai naudojamos matematinės funkcijos.

VIII-3 lentelė. DBVS dažniausiai naudojamos matematinės funkcijos

Funkcija	**Prasmė**
ABS(x)	absoliuti (be ženklo) x reikšmė
SIGN(x)	x ženklo reikšmė kaip -1, 0 arba 1 (atitinkamai neigiama, nulis arba teigiama)
MOD(x,y)	sveikoji x dalybos iš y dalmens dalis (tas pats, kas x%y)
FLOOR(x)	didžiausia sveikoji reikšmė, mažesnė ar lygi x
CEILING(x)	mažiausia sveikoji reikšmė, didesnė ar lygi x
POWER(x,y)	x pakelta y laipsniu
ROUND(x)	x suapvalinta iki artimiausio sveiko skaičiaus
ROUND(x,d)	x suapvalinta iki d dešimtainių ženklų skaičiaus
SQRT(x)	Kvadratinė šaknis iš x

Pavyzdžiui, užklausa "SELECT Pavardė, ROUND(Atlyginimas) FROM Darbuotojai;" išrenkami darbuotojų atlyginimai, suapvalinti iki sveiko skaičiaus.

Lentelių jungimas

Išrenkant duomenis iš keleto lentelių, būtina atlikti jų reliacinę sąjungą, aprašytą VII.4 skyriuje. Tai gali būti atliekama dviem būdais:

a) nurodant sąlygą, kad išorinio rakto ir atitinkamo pirminio rakto reikšmės turi sutapti;
b) panaudojant operatorių JOIN.

Jungimo operacijoms iliustruoti naudosime projektų ir skyrių duomenų bazės pavyzdį (VII-3 pav.).

Dvi užklausos žemiau atlieka tą pačią dviejų lentelių sujungimo pagal sutampančias SkNr reikšmes operaciją.

```
SELECT Projektas, Pavadinimas
FROM Projektai, Skyriai
WHERE Projektai.SkNr = Skyriai.SkNr;
```

```
SELECT Projektas, Pavadinimas
FROM Projektai INNER JOIN Skyriai
ON Projektai.SkNr = Skyriai.SkNr;
```

Antruoju atveju panaudotas specialus jungimo operatorius JOIN. **Vidinis** jungimo operatorius (INNER JOIN) surenkantis visus įrašus iš abiejų lentelių, kuriuose sutampa atributo SkNr reikšmės. Tai reiškia, kad, jei yra projektų, kuriems nenurodytas atsakingas skyrius, arba skyrių, kurie nėra nurodyti jokiame lentelės ***Projektai*** įraše, jų eilutės į užklausos rezultatą nepateks. Rezultatas atrodys taip.

Projektas	Pavadinimas
Administravimas	Administracija
Planavimas	Planavimo skyrius
Inovacijos	Plėtros skyrius

JOIN operatorius gali būti ir **išorinis**, surenkantis visus įrašus iš pirmosios (kairiosios) lentelės, o iš kitos – tik tuos, kurių reikšmės sutampa su jungiamo atributo reikšmėmis pirmoje lentelėje (LEFT JOIN). Žemiau pateiktos užklausos

```
SELECT Projektas, Pavadinimas
FROM Projektai LEFT OUTER JOIN Skyriai
ON Projektai.SkNr = Skyriai.SkNr;
```

rezultatas yra lentelė, kurioje yra eilutės kiekvienam projektui, tačiau skyriaus numeris gali būti nurodytas ne kiekvienam.

Projektas	Pavadinimas
Administravimas	Administracija
Mokymai	
Planavimas	Planavimo skyrius
Inovacijos	Plėtros skyrius

Galima išrinkti ir atvirkščiai, visus įrašus iš antrosios (dešiniosios) lentelės, o iš pirmosios – tik tuos, kurių reikšmės sutampa su jungiamo atributo reikšmėmis antrojoje lentelėje (RIGHT JOIN). Užklausos

```
SELECT Projektas, Pavadinimas
FROM Projektai RIGHT OUTER JOIN Skyriai
ON Projektai.SkNr = Skyriai.SkNr;
```

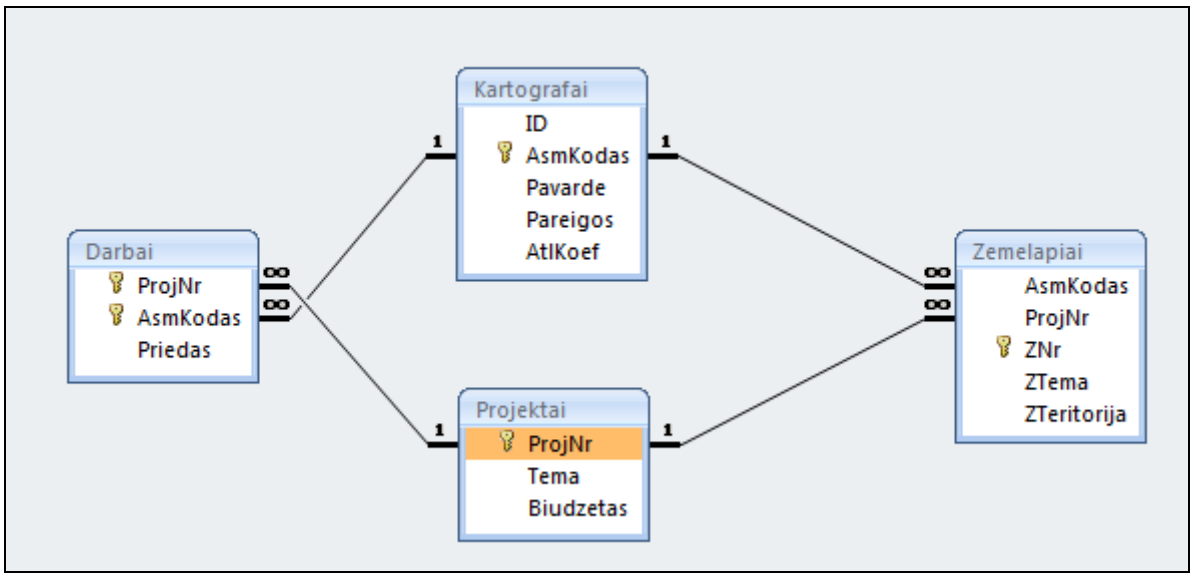
rezultatas yra lentelė, kurioje pateikiama informacija apie visus skyrius, tačiau projekto numeris nurodytas ne kiekvienam skyriui. Pavyzdžiui, Technologijų skyriui šio atributo reikšmė yra NULL.

Projektas	SkNr	Pavadinimas
Administravimas	S1	Administracija
Planavimas	S2	Planavimo skyrius
Mokymai	S3	Rinkodaros skyrius
	S4	Technologijų skyrius
Inovacijos	S5	Plėtros skyrius

VIII.4 DUOMENŲ IŠRINKIMO UŽKLAUSŲ PAVYZDŽIAI

Pavyzdžiuose panagrinėsime, kaip dar gali būti panaudotos SELECT sakinio galimybės. Pateiktos užklausos duomenų bazei *Kartografai.mdb*, kuri pridedama prie šios knygos.

Duomenų bazėje yra keturios lentelės, susietos ryšiais, kaip parodyta diagramoje žemiau.



VIII-1 pav. Duomenų bazės "Kartografai" ryšių diagrama

1.
SELECT Pavardė, Atlyginimas-Atlyginimas*31% FROM Kartografai WHERE GimimoMetai<1950;
Išrenkamos kartografų, gimusių anksčiau, negu 1950 metais, pavardės, prie jų nurodomas atlyginimas, iš kurio išskaičiuotas 31%.

SELECT * FROM Kartografai;
Rezultatas – jau esančios lentelės Kartografai kopija, t.y., išrinkti visi šios lentelės įrašai.

SELECT Kartografai.*, Žemėlapiai.* FROM Kartografai, Žemėlapiai;
Įvykdžius šią užklausą, gaunama dviejų lentelių Dekarto sandauga.

2.
SELECT [Pavardė], [Pareigos], [AtlKoef] AS Koeficientas
FROM Kartografai
WHERE Pareigos<>"Projektų vadovas" And AtlKoef>3;

Taip išrenkamos pavardės, pareigos ir atlyginimo koeficientas kartografų, išskyrus projektų vadovą, kurių atlyginimo koeficientas didesnis už 3. Stulpelis *AtlKoef* rezultate pervadinamas *Koeficientas*. Sąlygoje galimi palyginimo operatoriai: =, <>, <, >, >=, <=.

3.
SELECT DISTINCT [Kartografai].[Pareigos], [Kartografai].[AtlKoef]
FROM Kartografai
WHERE ([Kartografai].[Pareigos])<>"Projektų vadovas")
ORDER BY [Kartografai].[Pareigos] DESC , [AtlKoef] ASC;

Naudojant ORDER BY instrukciją išrinktas eilutes galima išrūšiuoti pagal atributų reikšmes didėjimo (ASC) arba mažėjimo (DESC) tvarka.

4.
```
SELECT [Pavardė], Int([AtlKoef]*14.5) AS Atlyginimas
FROM Kartografai
ORDER BY [Pavardė];
```

Atlyginimo koeficientas padauginamas iš 14.5 ir rezultatas paverčiamas į sveiką skaičių, atmetant trupmeninę dalį (MS Access funkcija Int).

5.
```
SELECT [Kartografai].[Pavardė], Žemėlapiai.*
FROM Kartografai, Žemėlapiai
WHERE [Kartografai].[AsmKodas]=[Žemėlapiai].[AsmKodas]
ORDER BY [ZTema], [Pavardė];
```

Atliekama natūrali reliacinė aibių sąjunga – iš dviejų lentelių išrenkamos kartografų pavardės ir kiekvieno atitinkamo kartografo sudaryto žemėlapio visa informacija. Rezultatas visų pirma išrūšiuojamas pagal žemėlapio temą, o vienos temos eilutės – pagal kartografo pavardę.

6.
```
SELECT DISTINCT [Kartografai].[Pavarde], [Žemėlapiai].[ProjNr], [Žemėlapiai].[ZNr],
[Žemėlapiai].[ZTema]
FROM Kartografai, Darbai, Žemėlapiai
WHERE [Kartografai].[AsmKodas]=[Darbai].[AsmKodas] And
[Žemėlapiai].[ProjNr]=[Darbai].[ProjNr];
```

Išrenkami žemėlapiai, sudaryti tik konkretiems projektams, nurodant pavardes visų kartografų, dirbančių atitinkamame projekte.

7.
```
SELECT K1.Pavardė AS Pirmas, K2.Pavardė AS Antras, D1.ProjNr AS Projektas
FROM Kartografai AS K1, Kartografai AS K2, Darbai AS D1, Darbai AS D2
WHERE D1.ProjNr=D2.ProjNr AND K1.AsmKodas=D1.AsmKodas AND
K2.AsmKodas=D2.AsmKodas AND K1.AsmKodas< K2.AsmKodas
ORDER BY D1.ProjNr;
```

Taip išrenkamos poros kartografų, dirbančių tame pačiame projekte. Naudojant instrukciją AS galima padaryti keletą to paties pagrindinės lentelės stulpelio kopijų.

8.
```
SELECT COUNT(*) AS Skaičius
FROM Kartografai;
```
Rezultatas – lentelė su vienu stulpeliu *Skaičius* ir vienintele eilute, kurioje yra lentelės ***Kartografai*** įrašų skaičius.

```
SELECT MAX([AtlKoef]) AS Didžiausias, MIN([AtlKoef]) AS Mažiausias
FROM Kartografai
WHERE Pareigos="Kartografas";
```
Rezultatas – lentelė su dviem stulpeliais *Didžiausias* ir *Mažiausias* ir vienintele eilute, kurioje yra atitinkamai maksimali ir minimali atlyginimo koeficiento reikšmės.

Apibendrinančios funkcijos COUNT, MAX, MIN, AVG (vidutinė reikšmė), SUM ir kt., yra taikomos eilučių grupėms pagal duotą sąlygą arba visai lentelei.

9.

```
SELECT Darbai.AsmKodas, SUM(Darbai.Priedas) AS BendrasPriedas
FROM Darbai
GROUP BY Darbai.AsmKodas;
```

Susumuojami atlyginimo priedai visiems kartografams pagal lentelę ***Darbai***. Galima šią užklausą parašyti ir kitaip, bet jei nenaudojama GROUP BY funkcija, rezultate atsiras eilutės kartografams, kurie nedalyvauja jokiuose projektuose:

```
SELECT Kartografai.AsmKodas, ( SELECT SUM(Darbai.Priedas) FROM Darbai WHERE
Darbai.AsmKodas= Kartografai.AsmKodas) AS BendrasPriedas
FROM Kartografai;
```

Įdėtos operacijos leistos palyginti neseniai – 1992 metais. Tai vienas svarbiausių SQL patobulinimų

10.

```
SELECT Darbai.ProjNr
FROM Darbai
GROUP BY ProjNr
HAVING COUNT (Darbai.AsmKodas)>3;
```

Išrenkami projektai, kuriuose dirba daugiau kaip trys kartografai.
HAVING instrukcija taikoma grupėms yra analogiška instrukcijai WHERE, taikomai eilutėms – pagal HAVING išrenkamos grupės su norima savybe. Išraiška po HAVING grupės viduje turi būti vienareikšmė.

```
SELECT [Žemėlapiai].[ProjNr], [Projektai].[Biudžetas]
FROM Žemėlapiai, Projektai
WHERE [Projektai].[ProjNr]=[Žemėlapiai].[ProjNr]
GROUP BY Žemėlapiai.[ProjNr], [Projektai].[Biudžetas]
HAVING COUNT (Žemėlapiai.ZNr)>2;
```

Išrenkami žemėlapiai, sudaryti projektuose, kurie iš viso apima daugiau kaip du žemėlapius, nurodant projekto biudžetą kiekvienam žemėlapiui.

11.

```
SELECT DISTINCT [Kartografai].[Pavardė]
FROM Kartografai
WHERE [Kartografai].[AsmKodas] IN
        (SELECT Žemėlapiai.AsmKodas
         FROM Žemėlapiai
         GROUP BY AsmKodas
         HAVING COUNT (ZNr)>1
         );
```

Išrenkamos pavardės kartografų, sudariusių daugiau kaip vieną žemėlapį.
Iš pradžių įvykdoma užklausa žemiausiame lygmenyje. Sąlyga WHERE … IN reiškia, kad imamos tik tos nurodyto atributo reikšmės, kurios atsiranda įdėtos užklausos rezultato lentelėje.

12.
```
SELECT DISTINCT [Kartografai].[Pavardė]
FROM Kartografai
WHERE [Kartografai].[AsmKodas] IN
        (SELECT Žemėlapiai.AsmKodas
         FROM Žemėlapiai
         WHERE AsmKodas IN (SELECT AsmKodas
                            FROM Darbai
                            WHERE Priedas >10
                           )
         GROUP BY AsmKodas
         HAVING COUNT (ZNr)>1
         );
```

Gali būti daugiau negu dvi įdėtos užklausos. Čia išrenkamos pavardės kartografų, gaunančių daugiau kaip 10% priedą ir sudariusių daugiau kaip vieną žemėlapį

13.
```
SELECT [Pavardė], [AtlKoef]
FROM Kartografai
WHERE AtlKoef > (SELECT AVG (AtlKoef) FROM Kartografai );
```

Išrenkamos pavardės kartografų, kurių atlyginimo koeficientas yra didesnis už vidutinį apskaičiuotą.

14.
```
SELECT DISTINCT [Kartografai].[Pavardė]
FROM Kartografai
WHERE EXISTS
      (SELECT *
       FROM Žemėlapiai
       WHERE Žemėlapiai.AsmKodas = Kartografai.AsmKodas AND Žemėlapiai.ProjNr = „P1“);
```

Išrenkamos pavardės kartografų, kurie yra sudarę bent vieną žemėlapį projektui P1. Sąlyga EXISTS yra patenkinta tik tada, kai įdėtos užklausos rezultatas – netuščia lentelė.

15.
```
SELECT DISTINCT [Kartografai].[Pavardė], Darbai.ProjNr
FROM Kartografai, Darbai
WHERE Kartografai.AsmKodas = Darbai.AsmKodas AND NOT EXISTS
        (SELECT *
         FROM Žemėlapiai
         WHERE  Žemėlapiai.ProjNr = Darbai.ProjNr
         );
```

Išrenkamos pavardės kartografų, dirbančių projektuose, kuriems nėra sudarytas nė vienas žemėlapis. Sąlyga NOT EXISTS yra patenkinta tik tada, kai įdėtos užklausos rezultatas yra tuščia lentelė.

16.
```
SELECT DISTINCT [Kartografai].[Pavardė], [Žemėlapiai].[ProjNr]
FROM Kartografai, Žemėlapiai
WHERE Kartografai.AsmKodas=[Žemėlapiai].[AsmKodas] AND NOT EXISTS
        (SELECT *  FROM Darbai
         WHERE Darbai.ProjNr = Žemėlapiai.ProjNr AND Darbai.AsmKodas =
Kartografai.AsmKodas );
```

Išrenkamos pavardės kartografų, kurie nedirba atitinkamuose projektuose, bet sudaro tiems projektams žemėlapius, nurodant projekto numerį .

17.
```
SELECT DISTINCT [Kartografai].[Pavardė]
FROM Kartografai, Žemėlapiai
WHERE NOT EXISTS
        (SELECT *  FROM Projektai
         WHERE NOT EXISTS
                    (SELECT * FROM Darbai
                     WHERE Darbai.ProjNr =  Projektai.ProjNr
                     AND  Darbai.AsmKodas = Kartografai.AsmKodas));
```

Taip išrenkami kartografai, dirbantys absoliučiai visuose projektuose – visuotinumo kvantorių FORALL (žr. priedą) atstoja dviguba NOT EXISTS sąlyga.
Tą pačią užklausą galima parašyti ir aplinkiniu keliu – išrenkami kartografai, kurių projektų skaičius sutampa su visų projektų skaičiumi:

```
SELECT DISTINCT [Kartografai].[Pavardė]
FROM Kartografai
WHERE (SELECT COUNT (Darbai.ProjNr)
        FROM Darbai
        WHERE Darbai.AsmKodas = Kartografai.AsmKodas) =  (SELECT COUNT (ProjNr)
                                                          FROM Projektai);
```

Antruoju atveju padeda vientisumo taisyklė: negali būti darbų projektuose, kurie neaprašyti lentelėje ***Projektai***.

18.
```
SELECT Žemėlapiai.ZNr, Žemėlapiai.ZTema
FROM Žemėlapiai
WHERE AsmKodas = „18“
UNION SELECT Žemėlapiai.ZNr, Žemėlapiai.ZTema
FROM Žemėlapiai
WHERE ZTema = „Istorija“;
```

Atliekama tradicinė aibių sąjunga, išrenkant žemėlapius, kurie tenkina bent vieną iš dviejų nurodytų sąlygų. SQL palaiko aibių operacijas UNION (sąjunga), INTERSECT (sankirta), EXCEPT (skirtumas).

VIII.5 DUOMENŲ MODIFIKAVIMO OPERACIJOS

Trys pagrindinės duomenų modifikavimo operacijos yra įterpimas, naikinimas ir reikšmių keitimas.

Duomenų naikinimo operacija DELETE.
Šio sakinio bendras pavidalas yra

DELETE FROM <lentelės vardas> WHERE <predikatas>;
Jei predikatas praleistas, operacija taikoma visiems lentelės įrašams.

DELETE *
FROM Darbai
WHERE „Projektų Vadovas“ = (SELECT Kartografai.Pareigos FROM Kartografai WHERE Kartografai.AsmKodas = Darbai.AsmKodas);

Iš lentelės ištrinami įrašai apie direktoriaus projektus.

Įterpimo operacija INSERT.

INSERT INTO <lentelės vardas> [atributas, atributas...] VALUES [reikšmė, reikšmė...]
INSERT INTO <lentelės vardas> <dalinė užklausa>

Nurodytų atributų ir reikšmių skaičius bei tipai privalo sutapti.

INSERT INTO Darbai (AsmKodas, ProjNr, Priedas)
VALUES (“17“, „P7“, 24);

Taip yra įterpiama nauja eilutė. Beje, dėl išorinių raktų vientisumo taisyklių jos nebus galima įtraukti, jei nebus kartografo, kurio asmens kodas yra 17 arba projekto, kurio numeris P7.

INSERT INTO Darbai (AsmKodas, ProjNr, Priedas)
SELECT [AsmKodas], „P5“ AS ProjNr, 100 AS Priedas
FROM Kartografai
WHERE AtlKoef>4;

Įterpiama eilučių aibė, sudaryta naudojant lentelės Kartografai informaciją, atrinktą pagal sąlygą. Du stulpeliai yra pridedami kaip konstantos, t.y., jų reikšmės visose naujose eilutėse yra vienodos.

Atnaujinimo operacija UPDATE.

UPDATE <lentelės vardas> SET <atributas>= <išraiška>, {<atributas>= <išraiška>}
WHERE <predikatas>;

Jei predikatas praleistas, atnaujinimo operacija taikoma visiems lentelės atributams.

UPDATE Kartografai SET Pareigos = „specialistas“ WHERE Pareigos = „redaktorius“;
Visi kartografai redaktoriai „pervadinami“ specialistais.

UPDATE Kartografai SET Pareigos = „vyr. kartografas“ WHERE 2000 > (SELECT Biudžetas FROM Darbai WHERE Kartografai.AsmKodas = Darbai.Atsakingas);

Visi kartografai, atsakingi už projektus, kurių biudžetas didesnis negu 2000, „pervadinami“ vyr. kartografais.

UPDATE Kartografai SET AtlKoef = AtlKoef + AtlKoef*20/100
WHERE AtlKoef < 2;

Visų kartografų atlyginimo koeficientas padidinamas 20%, jei tik jis buvo mažesnis už 2.

Problemos, kylančios vykdant duomenų modifikavimo operacijas, dažniausiai susijusios su duomenų vientisumu.

1. Gali būti pažeistas nuorodų vientisumas (nepašalinamos atsiradusios vienodos eilutės).
2. Jei WHERE sąlygoje yra dalinė užklausa, joje negali būti naudojama pati modifikavimo operacijos tikslinė lentelė.

Klausimai diskusijai

Aptarkite skirtingais būdais atliekamo lentelių jungimo privalumus ir trūkumus.

Užduotis savarankiškam darbui

Suprojektuokite duomenų bazę, kurioje būtų saugomi statistiniai duomenys apie Lietuvos savivaldybes, pavyzdžiui, gyventojų skaičius, užimtumas, išsilavinimas, struktūra pagal tautybę. Žinant, kad savivaldybės priklauso regionams, o penki regionai sudaro visą Lietuvos teritoriją, parašykite užklausas, kurios suskaičiuotų rodiklius atskirai kiekvienam regionui ir visai valstybei.

Užduotys praktikos darbams

Patikrinkite, kaip *MS Access* skaičiuoja predikatus, kai naudojamos NULL reikšmės, ar yra tenkinamos trireikšmės logikos taisyklės:

- (TRUE OR NULL) = TRUE,
- (FALSE AND NULL) = FALSE,
- (FALSE OR NULL) IS NULL = TRUE,
- (TRUE AND NULL) IS NULL = TRUE.

Parašykite užklausas, kuriomis būtų sukurta lentelė *Žemėlapiai*, jai pridėtas naujas stulpelis, sunaikintas esamas stulpelis, išmesta visa lentelė. Užpildykite lentelę *Žemėlapiai* duomenimis.

Išbandykite pavyzdžiuose pateiktas išrinkimo užklausas, išsiaiškinkite kaip jos veikia, modifikuokite. Išbandykite pavyzdžiuose pateiktas duomenų apibrėžimo ir modifikavimo užklausas, išsiaiškinkite, kaip jos veikia, pakeiskite jų parametrus.

Sukurkite užklausas, įvykdykite, įsitikinkite, kad gaunamas norimas rezultatas.

Išrinkti žemėlapius, sudarytus konkrečiam projektui, ir jų autorius.

Išrinkti žemėlapį, kurio projekte jis yra vienintelis žemėlapis.

Išrinkti daugiausiai žemėlapių sudariusį kartografą.

Išrinkti kartografus, kurių atlyginimo koeficientas didesnis už pusę maksimalaus kartografų atlyginimų koeficiento.

Išrinkti projektus, kuriuose dirba nelyginis skaičius kartografų.

Išrinkti projektus tokius, kad už juos gaunamų priedų visiems kartografams suma neviršytų 100.

Patikrinkite, ar visose anksčiau sukurtose duomenų bazėse tinkamai taikomos vientisumo taisyklės. Sukurkite situaciją, kai įvesti duomenys neleidžia apibrėžti kurio nors ryšio nuorodų vientisumo, parašykite užklausą SQL kalba, kuria būtų išrinktos išorinio rakto reikšmės, neturinčios atitinkančių pirminio rakto reikšmių tikslinėje lentelėje. Naudodami užklausos rezultatus sutvarkykite duomenis ir iš naujo apibrėžkite ryšį.

IX. NORMINIMAS

Viena strėlė numuša vieną erelį. Dviejų strėlių per daug.
Čanų sentencija, XV a.

Šiame skyriuje panagrinėsime, kaip reikėtų parinkti tinkamą loginę duomenų bazės struktūrą. Apskritai tai yra intuicijos reikalaujantis darbas, taigi, daugiau menas negu mokslas, tačiau egzistuoja tam tikri moksliniai principai, kuriuos šiame skyriuje išdėstysime.

IX.1 FUNKCINĖS PRIKLAUSOMYBĖS

Funkcinės priklausomybės (FP) sąvoka jei ir nėra fundamentali tikrąja to žodžio prasme, tai bent jau labai artima fundamentaliai duomenų bazių teorijoje. FP – tai ryšys „vienas su daug“ tarp atributų aibių viename santykyje. Pavyzdžiui, santykyje ***Darbai*** funkcinė priklausomybė egzistuoja tarp atributų poros (AsmKodas, ProjNr, PasoNr) ir atributo Priedas. Toliau apibrėšime FP savybes. Kol kas kaip pavyzdį nagrinėsime lentelę ***Darbai*** (AsmKodas, ProjNr, PasoNr, Vieta). Reikia pabrėžti, kad niekada neturi būti painiojamos konkrečiu laiko momentu lentelėje esančios reikšmės ir visas įmanomų reikšmių rinkinys.

Abiems atvejams formaliai apibrėšime FP sąvoką.

Tegu R yra santykis, o X ir Y – bet kurios to santykio atributų aibės. Y funkciškai priklauso nuo X (žymėsime X → Y; dar skaitysime „X funkciškai apibrėžia Y“) tada ir tik tada, kai kiekviena X aibės reikšmė yra susieta su lygiai viena Y reikšme.

Kitaip tariant, jei santykyje R du kortežai sutampa pagal X, jie būtinai sutaps ir pagal Y. Pavyzdžiui, egzistuoja FP AsmKodas → PasoNr, nes asmens kodas apibrėžia žmogų, kuris gali turėti tik vieną pasą, todėl, jei sutampa asmens kodai, sutaps ir paso numeriai.

Santykyje ***Darbai*** yra ir daugiau funkcinių priklausomybių:

(ProjNr) → Vieta
(AsmKodas, ProjNr) → AsmKodas
(AsmKodas, ProjNr) → ProjNr
(AsmKodas, ProjNr) → (AsmKodas, ProjNr, Priedas, Vieta) …

Kairioji FP pusė vadinama determinantu, o dešinė – priklausomuoju. Abi pusės yra atributų poaibiai. Jei juos sudaro vienintelis atributas, jie vadinami elementariaisiais. Toks apibrėžimas tinka pirmajam variantui (konkrečioms santykio R reikšmėms). Jį galima praplėsti taip, kad tiktų **kiekvienam galimam R** įtraukus šią frazę į aukščiau pateiktą apibrėžimą. Tokį apibrėžimo variantą toliau ir naudosime.

Reikia pastebėti, kad jei X yra santykio R potencialus raktas, tai visi R atributai **būtinai** (pagal CK apibrėžimą) turi būti funkciškai priklausomi nuo X.

Jei santykyje R galioja FP A → B, o A nėra potencialus raktas, tai galima teigti, kad R yra kažkoks informacijos perteklius, pavyzdžiui, faktas, kad projektas P1 vykdomas Vilniuje, be reikalo kartojamas daugelį kartų.

Kiekviename santykyje galima aptikti daug funkcinių priklausomybių. Projektuotojo tikslas yra jų skaičių sumažinti iki „protingo“. Gali kilti klausimas, kodėl tai taip svarbu.

Kiekviena FP yra **vientisumo taisyklė**, todėl ji turi būti tikrinama kiekvieną kartą atnaujinant duomenų bazę. Kuo mažiau yra taisyklių, tuo paprasčiau, be to, dažniausiai galima keletą taisyklių suvesti į vieną, neprarandant jokios informacijos.

Akivaizdus supaprastinimo būdas – atsisakyti trivialių FP, t.y., tokių FP, kurių tiesiog negali nebūti, pavyzdžiui, (AsmKodas, ProjNr) $\rightarrow$ AsmKodas. Faktiškai FP yra triviali tada ir tik tada, kai dešinioji pusė yra determinanto poaibis. Tokios FP visiškai neįdomios.

Priklausomybių aibės papildymas.

Kaip jau buvo minėta, kai kurios FP reiškia kitas FP, pavyzdžiui, (AsmKodas, ProjNr) $\rightarrow$ (Priedas, Vieta) reiškia (AsmKodas, ProjNr) $\rightarrow$ (Priedas) ir (AsmKodas, ProjNr) $\rightarrow$ (Vieta).

Sudėtingesnis pavyzdys būtų santykis R su atributais A, B ir C tokiais, kad galioja FP A $\rightarrow$ B ir B $\rightarrow$ C. Nesunku pastebėti, kad tada egzistuoja ir priklausomybė A $\rightarrow$ C, kuri vadinama **tranzityvia** priklausomybe, t.y., C tranzityviai priklauso nuo A per B.

Visų FP, kurios nurodomos per duotą FP aibę S, aibė vadinama S papildiniu S'. Pirmą kartą apskaičiuoti S' pabandė Armstrongas, sudaręs FP išvedimo iš duotųjų FP taisykles.

Armstrongo taisyklės. Tegu A, B ir C – bet kokios santykio R atributų aibės.

1. Refleksyvumas. Jei B $\subset$ A, tai A $\rightarrow$ B.
2. Papildymas. Jei A $\rightarrow$ B, tai A$\cup$C $\rightarrow$ B$\cup$C.
3. Tranzityvumas. Jei A $\rightarrow$ B ir B $\rightarrow$ C, tai A $\rightarrow$ C.

Visas Armstrongo taisykles galima įrodyti remiantis FP apibrėžimu, pavyzdžiui refleksyvumas – tai trivialios priklausomybės apibrėžimas. Taisyklės yra išsamios ta prasme, kad iš aibės S, kuri yra minimalus santykio R funkcinių priklausomybių rinkinys, visos kitos FP išvedamos remiantis tik tomis taisyklėmis, be to, jokių kitokių FP jomis remiantis išvesti neįmanoma. Taigi, taisyklės panaudojamos S papildymui iki S'.

Siekiant supaprastinti darbą, įvesta daugiau taisyklių.

4. A $\rightarrow$ A
5. Dekompozicija. A $\rightarrow$ B$\cup$C $\Rightarrow$ A $\rightarrow$ B ir A $\rightarrow$ C
6. Sąjunga. A $\rightarrow$ B, A $\rightarrow$ C $\Rightarrow$ A $\rightarrow$ B$\cup$C
7. Kompozicija. A $\rightarrow$ B ir C $\rightarrow$ D $\Rightarrow$ A$\cup$C $\rightarrow$ B$\cup$D

Darvenas įrodė **visuotinio apjungimo teoremą**, iš kurios galima išvesti praktiškai visas minėtas taisykles.

A $\rightarrow$ B, C $\rightarrow$ D $\Rightarrow$ A$\cup$(C – B) $\rightarrow$ B$\cup$D

Pavyzdys

Tegu turime tokius santykio atributus:
A – asmens kodas
B – skyriaus numeris
C – viršininko kodas
D – viršininko vadovaujamas projektas
E – skyriaus pavadinimas
F – vadovo skiriamas projektui laikas.

Čia yra nurodytos tokios funkcinės priklausomybės:

1. A $\rightarrow$ B$\cup$C
2. B $\rightarrow$ E
3. C$\cup$D $\rightarrow$ E$\cup$F.

Įrodysime, kad A$\cup$D $\rightarrow$ F.

$A \to B \cup C$ (duota, 1) $\Rightarrow A \to C$ (dekompozicija) $\Rightarrow$ [$A \cup D \to C \cup D$ (papildymas) ir $C \cup D \to E \cup F$ (duota, 3)] $\Rightarrow A \cup D \to E \cup F$ (tranzityvumas) $\Rightarrow A \cup D \to F$ (dekompozicija).

Jei S1' = S2', tai S1 ir S2 yra ekvivalenčios FP aibės, t.y., apribojimai, išreikšti per S1, gali būti išreikšti per S2 ir atvirkščiai.
FP aibė S vadinama nesuprastinama, jei

a) jos kairioji pusė yra nesuprastinama,
b) dešiniąją pusę sudaro vienintelis atributas,
c) negalima išmesti nė vienos FP, išsaugant buvusią S'.

Pavyzdžiui, išmetus bent vieną priklausomybę iš rinkinio { (AsmKodas) → Pavardė, (AsmKodas) → Amžius, (AsmKodas) → Miestas }, bus prarasta dalis informacijos, todėl ji yra nesuprastinama.

Kiekvienai FP aibei S egzistuoja bent viena ekvivalenti nesuprastinama aibė.

IX.2 NORMINIMO PROCEDŪRA

Kaip pavyzdį nagrinėsime lentelę ***Darbai*** (AsmKodas, Pavardė, ProjNr, PasoNr, Vieta).
Matyti, kad tokioje lentelėje, yra sąlygos atsirasti perteklinei informacijai, konkrečiai – įvedus realius duomenis daug kartų kartosis derinys (AsmKodas, Pavardė). Dar blogiau yra tai, kad keičiant duomenis įmanoma sukurti situaciją, kai asmens kodą AK1 vienoje vietoje atitinka Pavardė1, o kitoje – Pavardė2. Tai ir reiškia, kad duomenų bazės struktūra yra kažkuo „bloga".

Iš tikrųjų, vienas faktas turi būti saugomas tik vieną kartą. Ši intuityvi samprata formaliai išreiškiama norminimu, kuris remiasi tam tikrais metodais. Pirmoji NF reiškia, kad santykyje yra tik neskaidomos (skaliarinės) reikšmės ir pagal apibrėžimą kiekvienas santykis duomenų bazėje yra pirmosios NF. Tačiau tokio norminimo nepakanka, jis neapsaugo nuo duomenų pertekliaus. Yra ir kitos, griežtesnės NF, kurios leidžia pagerinti duomenų bazės struktūrą.

Norminės formos.
Norminimas pagrįstas NF samprata. Santykis yra kurios nors NF, jei jis tenkina atitinkamą taisyklių rinkinį, pavyzdžiui, 1NF tada ir tik tada, kad santykyje yra tik neskaidomos (skaliarinės) reikšmės. Dažniau terminas „norminis" taikomas kalbant apie aukštesnes NF, ypač apie 3NF.

Norminimo lygmenys yra tokie:
1NF ⊃ **2NF** ⊃ **3NF** ⊃ Boiso-Kodo NF ⊃ 4NF ⊃ 5NF …
Kuo aukštesnė NF, tuo labiau ji pageidautina. Iš vienos NF galima gauti kitą, ir taip nuo pirmosios iki penktosios, norminimo proceso metu neprarandant informacijos. Dažniausiai naudojamos pirmosios trys norminės formos.

Pastaba: nors bendra norminimo idėja yra ta, kad projektuojant DB būtų siekiama 5NF, tai nėra privaloma taisyklė, t.y., yra atvejų, kai tolesnis norminimas pablogina DB struktūrą. Be abejo, norminimo principus būtina žinoti bet kuriuo atveju.

Norminimo procedūra – tai santykio skaidymas į keletą santykių, dekomponavimo metu išsaugant **visą** informaciją. Taigi, dekomponavimas turi būti grįžtamasis procesas. Kaip nustatyti, ar dekomponavimo metu informacija prarandama, ar ne – tai klausimas, glaudžiai susijęs su FP sąvoka.

Pavyzdys

Santykis S (Darbuotojas, Projektas, Miestas) gali būti dekomponuojamas dvejopai:

Darbuotojas	Projektas	Miestas
D1	P1	M1
D2	P1	M2

S1 (Darbuotojas, Projektas) arba S1 (Darbuotojas, Projektas)
S2 (Darbuotojas, Miestas) S2 (Projektas, Miestas)

Pirmuoju atveju visa buvusi informacija išlieka. Antruoju atveju dalis informacijos prarandama, nes keli darbuotojai gali dirbti viename projekte ir išskaidžius nebebus galima pasakyti, kad, pavyzdžiui, D1 dirba mieste M1, o D2 – mieste M2.

Kodėl atsitiko taip, kad pirmoji dekompozicija yra be praradimų, o antroji – ne? Pastebėsime, kad tai, ką vadinome dekompozicija, iš tiesų yra pradinio santykio projektavimas.

Pirmuoju atveju informacijos išsaugojimas reiškia, kad galime du gautus santykius vėl sujungti į pradinį, o antruoju atveju bandant tą padaryti atsiras „klaidingi“ kortežai, pavyzdžiui, (D1, P1, M2), tuo tarpu metodo atskirti teisingiems ir klaidingiems kortežams nėra.

Dekompozicijos grįžtamumas reiškia, kad projekcijų sąjunga yra pradinis santykis. Iš to seka klausimas: jei R1 ir R2 yra santykio R projekcijos, apimančios visus R atributus, kokios sąlygos turi būti patenkintos, kad sujungus R1 ir R2 garantuotai gautume R. Čia ir pritaikoma FP sąvoka.

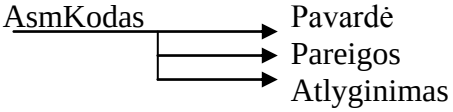
Tarkime, kad pradiniame santykyje yra apibrėžta minimali FP aibė (laikoma, kad ji nesikeičia laike), pavyzdžiui, Darbuotojas $\rightarrow$ Projektas ir Darbuotojas $\rightarrow$ Miestas.

Hezo (Heath) teorema. Jei R (A,B,C,D) yra santykis ir galioja funkcinė priklausomybė A $\rightarrow$ B, tai R = (A,B) $\cup$ (A, C, D) , t.y., santykis lygus jo projekcijų reliacinei sąjungai.

Tačiau teorema nepaaiškina, kodėl antruoju atveju informacija prarandama. Tačiau galima nujausti, kad antruoju atveju buvo prarasta viena iš funkcinių priklausomybių, būtent Darbuotojas $\rightarrow$ Miestas, o išliko tik FP Darbuotojas $\rightarrow$ Projektas.

Iš kairės pusės „minimalios“ priklausomybės – tai tokios FP, kurių kairioji pusė yra „ne per didelė“, kaip, pavyzdžiui, yra (AsmKodas, ProjNr) $\rightarrow$ … santykyje ***Darbai*.** Visi kiti santykio atributai yra minimaliai priklausomi nuo šios poros.

FP diagramos (schemos) – tai nuorodų diagramos, kuriomis dažnai vaizduojamos minimalios santykių FP aibės. Matyti, kad jose rodyklės prasideda potencialaus rakto reikšme (taip turi būti pagal apibrėžimą, nes kiekvienai CK reikšmei egzistuoja bent viena kokio nors kito atributo reikšmė).



Galima atsisakyti kai kurių priklausomybių, kurios kelia problemas, tačiau rodyklės, einančios iš CK, privalo išlikti. Neformaliai norminimas apibrėžiamas kaip procedūra, kurios metu naikinamos rodyklės, neinančios iš CK.

Žinoma, FP – tai semantinė sąvoka. Priklausomybių atpažinimas yra duomenų prasmės išaiškinimo dalis, pavyzdžiui, jei egzistuoja FP AsmKodas → AtlygKoef, tai reiškia, kad kiekvienas darbuotojas gauna tik vieną atlyginimą. Situaciją galima aprašyti taip:

- realiame pasaulyje egzistuojantis apribojimas yra įtrauktas į duomenų bazę;
- kadangi tai prasminiai duomenys, jie turi būti aprašyti formaliai;
- duomenys turi būti sutvarkyti taip, kad būtų galima patikrinti, ar apribojimas galioja;
- metodas tam padaryti ir yra pagrįstas funkcinėmis priklausomybėmis.

Norminimo sąvoka leidžia paprastai ir aiškiai aprašyti funkcines priklausomybes.

IX.3 PIRMOJI, ANTROJI IR TREČIOJI NORMINĖS FORMOS

Iš karto pradėsime nuo trečiosios NF, kad būtų aišku, apie ką kalbama. O toliau panagrinėsime, kaip santykį paversti 3NF santykių rinkiniu. Kol kas dėl paprastumo laikysim, kad nagrinėjami santykiai turi vienintelį CK (aišku, tada jis turi būti ir PK).

Santykis yra 3NF tada ir tik tada, kai visi neraktiniai atributai yra:

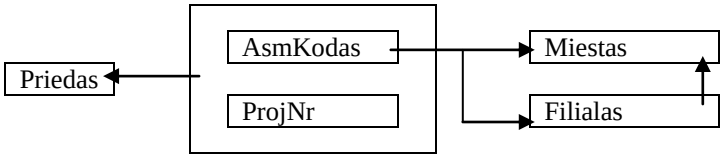
1) tarpusavyje nepriklausomi,
2) yra priklausomi nuo PK.

"Neraktinis" atributas – tai toks atributas, kuris neįeina į jokį potencialų raktą (mūsų atveju į PK). Du ar daugiau atributų vadinami tarpusavyje nepriklausomais, jei nė vienas iš jų nėra funkciškai priklausomas nuo bet kokio likusių atributų derinio, t.y., gali būti atnaujintas nepriklausomai nuo jų. Pavyzdžiui, santykis (AsmKodas, Pavardė, Amžius, Atlyginimas) yra 3NF, nes galima keisti amžių ar pavardę nekeičiant atlyginimo ir atvirkščiai, o visi šie atributai priklauso nuo asmens kodo reikšmės.

Dar labiau neformaliai: santykis yra 3NF tada ir tik tada, kai kiekvieną kortežą sudaro PK, identifikuojantis esybę ir neapibrėžtų arba tarpusavyje nepriklausomų atributų, kaip nors charakterizuojančių tą esybę, reikšmių aibė.

Pirmoji norminė forma (1NF) jau žinoma – ji tiesiog reiškia, kad visos atributų reikšmės yra skaliarai. Santykis gali būti pirmosios NF ir nebūti antrosios arba trečiosios NF. Tada jo struktūra nėra visai gera.

Panagrinėkime dar kitokį santykį S (AsmKodas, Filialas, Miestas, ProjNr, Priedas). Čia pirminis raktas yra (AsmKodas, ProjNr).
Santykio FP diagrama atrodo taip:



Tarkime, kad Filialas vienareikšmiškai nusako Miestą. Matyti, kad FP diagrama gana sudėtinga.

Pagal 3NF reikalavimus visos rodyklės turėtų prasidėti PK. O pavyzdyje yra ir papildomų rodyklių iš PK dalių ir apskritai ne iš CK. Be to, ne visi atributai minimaliai priklauso nuo PK – Miestas ir Filialas priklauso tik nuo PK dalies, t.y., nuo atributo AsmKodas.
Santykio **S** duomenys galėtų atrodyti taip

AsmKodas	ProjNr	Filialas	Miestas	Priedas
A1	P1	F1	Vilnius	10
A1	P2	F1	Vilnius	20
A1	P3	F1	Vilnius	10
A1	P4	F1	Vilnius	30
A1	P5	F1	Vilnius	40
A1	P6	F1	Vilnius	20
A2	P1	F2	Kaunas	40
A2	P2	F2	Kaunas	20
A3	P2	F3	Kaunas	10
A4	P2	F1	Vilnius	30
A4	P4	F1	Vilnius	20
A4	P5	F1	Vilnius	10

Duomenys atitinka FP diagramą. Pastebėsime, kad perteklius sukelia atnaujinimo anomalijas (istorinis pavadinimas, reiškiantis sunkumus atliekant INSERT, DELETE ir UPDATE operacijas).

Panagrinėsime perteklių
AsmKodas → Miestas
ir su juo susijusias problemas, laikant kad galioja visi vientisumo apribojimai.

Atliekant operaciją INSERT – negalima įtraukti duomenų apie tai, kad žmogus dirba konkrečiame mieste, neįtraukiant duomenų apie projektą, kuriame jis dirba (nes kol žmogus nedirba jokiame projekte, neturėsime apibrėžtos kortežo pirminio rakto reikšmės).

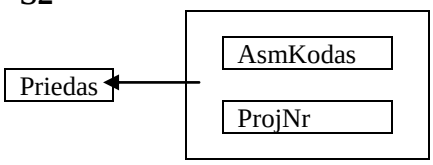
Atliekant operaciją DELETE – jei išmesime kortežą, atitinkantį vieną darbuotoją, išmesime ne tik informaciją apie jo darbą projekte, bet ir informaciją apie buvimą konkrečiame mieste. Iš tiesų, kadangi santykis **S** aprašo dvi esybes (Projektas ir Darbuotojas), keičiant vienos iš jų duomenis, pakenkiama kitai. Matyti, kad reikia atskirti šias esybes kaip logiškai susijusios informacijos atskirus blokus.

Atliekant operaciją UPDATE – pastebime, kad poros (AsmKodas, Miestas) kartojasi daug kartų. Todėl keičiant Miestą, pavyzdžiui, iš Vilniaus į Kauną, arba reikia išrinkti visas poras (A_i, Vilnius), arba dalis jų liks nepakeista (tai dar blogiau).

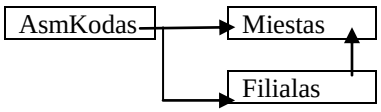
Sprendimas – išskaidyti **S** į du santykius.
S1 (AsmKodas, Miestas, Filialas) ir **S2** (AsmKodas, ProjNr, Priedas).

FP diagramos tada atrodo taip:

S2



S1



Atsikratėme priklausomybės nuo pirminio rakto dalies

Santykių **S1** ir **S2** duomenys dabar tokie (pastebėkime, kad atsirado galimybė įtraukti A5)

S1

AsmKodas	Filialas	Miestas
A1	F1	Vilnius
A2	F2	Kaunas
A3	F3	Kaunas
A4	F1	Vilnius
A5	*F4*	*Klaipėda*

S2

AsmKodas	ProjNr	Priedas
A1	P1	10
A1	P2	20
A1	P3	10
A1	P4	30
A1	P5	40
A1	P6	20
A2	P1	40
A2	P2	20
A3	P2	10
A4	P2	30
A4	P4	20
A4	P5	10

Taip pertvarkius santykį, nebekils problemų atnaujinant informaciją apie miestus ir projektus.

Antroji norminė forma (2NF)
Santykis yra antrosios NF tada ir tik tada, kai jis yra 1NF ir kiekvienas neraktinis atributas minimaliai priklauso nuo PK (t.y., nuo viso PK, bet ne nuo kurios nors jo dalies).

Santykiai **S1** ir **S2** yra 2NF.

Norminimo procesas yra grįžtamasis – sujungus **S1** ir **S2** pagal AsmKodas, vėl gausime **S**, be to, atsirado galimybė saugoti daugiau faktų, negu buvo galima senoje struktūroje (A5, kuris nedirba jokiame projekte). Taigi, 2NF yra ir praktiniu požiūriu geresnė už 1NF.

Norminimo procesą 1NF $\rightarrow$ 2NF galima formaliai užrašyti:
R(A,B,C,D) su PK (A,B) ir FP A $\rightarrow$ D skaidomas:
R1(A,D) su PK (A)
R2(A,B,C) su PK (A,B) ir FK (A), susietu su A santykyje **R1**.

Tačiau 2NF struktūra **S1** ir **S2** dar vis kelia kai kurių problemų.
S2 yra tvarkingas 3NF santykis ir jis toks liks. Tačiau santykio S1 neraktiniai atributai nėra nepriklausomi tarpusavyje. Konkrečiai tai yra „bloga“ funkcinė priklausomybė Filialas $\rightarrow$ Miestas. Nors atributas Miestas yra minimaliai priklausomas nuo AsmKodas, jis yra dar ir tranzityviai priklausomas nuo AsmKodas per atributą Filialas, nes iš tranzityvumo savybės išplaukia, kad jei
A $\rightarrow$ B ir B $\rightarrow$ C, tai A $\rightarrow$ C.

Dėl tranzityvių priklausomybių gali atsirasti atnaujinimo anomalijos. Šiuo atveju trukdo perteklius Filialas $\rightarrow$ Miestas ir su juo susiję duomenys.

Atliekant operaciją INSERT – negalima įtraukti duomenų apie tai, kad filialas yra konkrečiame mieste, tol kol filiale nedirba nė vienas darbuotojas (nes tuo atveju neturėsime apibrėžtos pirminio rakto reikšmės).

Atliekant operaciją DELETE – jei išmesime kortežą, atitinkantį vieną darbuotoją, išmesime ne tik informaciją apie jo darbą filiale, bet ir informaciją apie filialo buvimą konkrečiame mieste. Iš tiesų, santykis **S1** vėl aprašo dvi esybes (Filialas ir Darbuotojas), keičiant vienos iš jų duomenis, pakenkiama kitai. Matyti, kad teks atskirti šias esybes.

Atliekant operaciją UPDATE – poros (Filialas, Miestas) kartojasi daug kartų. Todėl keičiant Miestą, pavyzdžiui, iš Vilniaus į Kauną, arba reikia išrinkti visas poras (F_i, Vilnius), arba dalis jų liks nepakeista (tai dar blogiau).

Sprendimas – išskaidyti **S1** į du santykius.

S11 (AsmKodas, Filialas) ir **S12** (Filialas, Miestas).
Abi FP diagramos dabar elementarios ir santykiai yra 3NF.

Santykių **S11, S12** ir **S2** duomenys dabar yra tokie, kaip parodyta lentelėse žemiau. Pastebėkime, kad atsirado galimybė įtraukti įrašą F5, kuris reiškią faktą, kad Rygoje yra filialas (nors darbuotojų tame filiale kol kas nėra). Naujose lentelėse jau nėra pasikartojančių grupių, bet kokią informaciją apie darbuotojus, filialus ir projektus galima įvesti nepriklausomai.

S11

AsmKodas	Filialas
A1	F1
A2	F2
A3	F3
A4	F1
A5	F4

S12

Filialas	Miestas
F1	Vilnius
F2	Kaunas
F3	Kaunas
F4	Klaipėda
F5	*Ryga*

S2

AsmKodas	ProjNr	Priedas
A1	P1	10
A1	P2	20
A1	P3	10
A1	P4	30
A1	P5	40
A1	P6	20
A2	P1	40
A2	P2	20
A3	P2	10
A4	P2	30
A4	P4	20
A4	P5	10

Trečioji norminė forma (3NF)
Santykis yra 3NF tada ir tik tada, kai jis yra 2NF ir kiekvienas neraktinis atributas netranzityviai priklauso nuo pirminio rakto (t.y., nėra jokių kitokių tarpusavio priklausomybių).

Santykiai **S11, S12** ir **S2** yra 3NF.

Norminimo procesas taip pat yra grįžtamasis – sujungus **S11** ir **S12** pagal Miestas, vėl gausime **S1**, be to, atsirado galimybė saugoti dar daugiau faktų, negu buvo galima **S1** (F5, kuris yra mieste Rygoje, kuriame nėra jokio darbuotojo). Taigi, 3NF yra geresnė už 2NF.

Norminimo procesą 2NF $\rightarrow$ 3NF galima formaliai užrašyti:
R(A,B,C) su PK (A) ir FP B $\rightarrow$ C skaidomas:
R1(A,B) su PK (A) ir FK (B), susietu su B santykyje **R2**
R2(B,C) su PK (B).

Be abejo, norminė forma priklauso ne nuo konkrečių reikšmių, o nuo duomenų prasmės. Iš pirmo žvilgsnio į lentelę neįmanoma pasakyti, ar ji yra 3NF.

Priklausomybės išsaugojimas.
Pasitaiko atvejų, kai dekomponuoti galima įvairiai, pavyzdžiui, santykį S1 buvo galima skaidyti dvejopai:
S11 (AsmKodas, Miestas) ir **S12** (Filialas, Miestas) arba
S11 (AsmKodas, Miestas) ir **S12** (AsmKodas, Filialas).

Tačiau antruoju atveju gautume blogesnį rezultatą – sujungus du santykius nebegautume buvusio **S**, konkrečiai – faktų, kad miestas ir filialas yra susieti tarpusavyje nepriklausomai nuo darbuotojo. Antruoju atveju nėra atitinkamo išorinio rakto, todėl abu gauti santykiai būtų atnaujinami nepriklausomai vienas nuo kito. Pirmuoju atveju visos buvusios vientisumo taisyklės palaikomos automatiškai per raktus.

Kaip parinkti teisingą dekompoziciją?
R skaidomas į R1 ir R2 tada ir tik tada kai:

1. Kiekviena FP santykiuose R1 ir R2 yra santykio R funkcinių priklausomybių pasekmė;
2. R1 ir R2 bendri atributai sudaro nors vieno iš šių santykių potencialų raktą.

IX.4 BOISO-KODO NORMINĖ FORMA

Iki šiol laikėme, kad santykis turi tik vieną CK, t.y., PK. Dabar panagrinėsime bendresnį atvejį. Problema yra ta, kad Kodo apibrėžta 3NF ne visai tinka santykiams, kurie pasižymi žemiau išvardintomis savybėmis:

1) santykis turi daugiau, nei vieną CK;
2) bent du CK jame yra sudėtiniai;
3) jie persidengia (t.y., turi bendrų atributų).

Todėl originalus 3NF apibrėžimas vėliau buvo pakeistas griežtesniu amerikiečio Reimondo Boiso (Raymond Boyce) ir E.Kodo pateiktu apibrėžimu, gavusiu Boiso-Kodo norminės formos (BKNF) vardą. Iš tiesų dar anksčiau 1971 metais BKNF atitinkantį 3NF apibrėžimą pateikė Ijanas Hezas (Ian Heath), todėl ši norminė forma turėtų būti vadinama Hezo NF. Beje, minėtos trys sąlygos kartu pasitaiko gana retai, o kitais atvejais 3NF sutampa su BKNF.

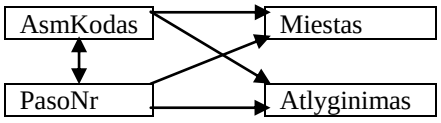
Santykis yra BKNF tada ir tik tada, kai kiekviena netriviali ir iš kairės nesuprastinama FP turi CK kaip determinantą.

Ne taip formaliai išsireiškiant, santykis yra BKNF tada ir tik tada, kai kiekvienas determinantas yra CK. Tai reiškia, kad kaip ir 3NF visos rodyklės diagramose išeis iš CK ir jokių kitokių nebus.

Pastebėsime, kad BKNF apibrėžime nebefigūruoja 1 ir 2NF, be to, jame nėra tranzityvios priklausomybės sąvokos. Taigi, jis konceptualiai paprastesnis už 3NF ir yra griežtesnis. Tačiau kiekvienas santykis, kuris yra 3NF, gali būti neprarandant informacijos dekomponuotas į BKNF santykių rinkinį.

Pavyzdys

Panagrinėkime dar kitokį santykį S (AsmKodas, PasoNr, Miestas, Atlyginimas). Jame potencialūs raktai yra AsmKodas ir PasoNr, o Miestas ir Atlyginimas nepriklauso vienas nuo kito.
FP diagrama atrodo taip:



Šis santykis yra BKNF. Nors diagrama sudėtingesnė už 3NF, visi FP determinantai yra potencialūs raktai. Tai reiškia, kad turėti daugiau nei vieną CK nėra blogai. Lieka tik paskelbti atributą PasoNr potencialiu raktu jau apibrėžiant duomenų bazę, kad būtų galima taikyti unikalumo apribojimą.

Jeigu CK persidengia, turėsime sudėtingesnį atvejį.

Pavyzdys

Panagrinėkime santykį **D** (AsmKodas, PasoNr, ProjNr, Atlyginimas). Čia potencialūs raktai yra (AsmKodas, ProjNr) ir (PasoNr, ProjNr). Toks santykis nėra BKNF, nes du determinantai (AsmKodas ir PasoNr, kurie priklauso vienas nuo kito) nėra CK.
Šiame santykyje yra tam tikras perteklius, taigi, ir atnaujinimo anomalijos. Jis yra 3NF, nes tenkinama taisyklė, kad kiekvienas ne rakto atributas turi minimaliai priklausyti nuo CK, o apie rakto atributus nekalbama. Tuo tarpu būtent potencialaus rakto atributas PasoNr neminimaliai priklauso nuo (AsmKodas, ProjNr).

Sprendimas – išskaidyti D. Galima skaidyti dvejopai:
D1 (AsmKodas, PasoNr) ir **D2** (AsmKodas, ProjNr, Atlyginimas) arba

D1 (AsmKodas, PasoNr) ir **D2** (PasoNr, ProjNr, Atlyginimas).

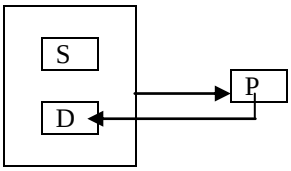
Šiuo atveju abu variantai vienodai geri.
Visi norminimo žingsniai – tai tik formaliai užrašyti sveiku protu pagrįsti sprendimai.

Panagrinėsime dar egzotiškesnę situaciją.

Pavyzdys
Santykyje **X** (S, D, P) aprašyti studentai, disciplinos ir dėstytojai. Laikysime, kad :

a) vieną studentą vieno dalyko moko tik vienas dėstytojas, t.y., (S, D) $\rightarrow$ P
b) vienas dėstytojas dėsto tik vieną discipliną, t.y., P $\rightarrow$ D.

FP diagrama atrodo taip:



Santykyje yra persidengiantys CK: (S, D) ir (S, P). Santykis yra 3NF, bet ne BKNF, nes determinantas P nėra CK. Galėtume dekomponuoti į santykius (S, P) ir (P, D), bet iš jų jau nebeatkurtume buvusios situacijos: būtų pararasta FP (S, D) $\rightarrow$ P.

Taigi, šis santykis neskaidomas, nors ir nėra BKNF.

Elegantišką 3NF ir BKNF apibrėžimą pateikė Karlas Dzaniolas (Carlo Zaniolo).

Santykis R yra 3NF, jei kiekviena FP X $\rightarrow$ A tenkina bent vieną iš trijų sąlygų:

a) A $\subset$ X (triviali FP);
b) CK $\subset$ X (kurio nors tame santykyje);
c) A $\subset$ CK (būtent ši savybė skiria 3NF ir BKNF).

IX.5 AUKŠTESNĖS NORMINĖS FORMOS

Pratęsime norminimo procesą. Pasirodo, kad yra ir aukštesnių už BKNF – ketvirtoji, penktoji norminės formos. Norint apibrėžti aukštesnes NF, prireiks ir kitų, bendresnių, sąvokų – daugiareikšmės priklausomybės (apbrėžiant 4NF) ir sąjungos priklausomybės (apbrėžiant 5NF), kurios yra FP sąvokos apibendrinimai. 5NF tam tikra prasme yra aukščiausia ir galutinė.

Ketvirtoji norminė forma (4NF)
Tarkime, kad turime tokį nenormintą santykį S’ (jis net nėra 1NF).

Kursas	**Dėstytojas**	**Vadovėlis**
Fizika	Jonaitis Petraitis	Mechanika Optika
Matematika	Jonaitis	Mechanika Trigonometrija Algebra

Tarkime, kad dėstytojai ir vadovėliai visai nepriklauso vienas nuo kito. FP apskritai neužduotos. Sunorminę šį santykį, gausime santykį S su septyniais įrašais. S yra visiškai BKNF.

Matyti, kad jei yra (K1, D1,V1) ir (K1, D2,V2), tai turi būti ir įrašai (K1, D2,V1) ir (K1, D1,V2).

Kursas	Dėstytojas	Vadovėlis
Fizika	Jonaitis	Mechanika
Fizika	Jonaitis	Optika
Fizika	Petraitis	Mechanika
Fizika	Petraitis	Optika
Matematika	Jonaitis	Mechanika
Matematika	Jonaitis	Trigonometrija
Matematika	Jonaitis	Algebra

Aišku, ir šiame santykyje yra perteklius ir su juo susijusios atnaujinimo anomalijos. Pavyzdžiui, jei norėsime pasakyti, kad fiziką dar dėsto Antanaitis, reikės įtraukti jam du kortežus (kiekvienam fizikos vadovėliui).

Tokios blogybės pastebėtos gana seniai. Mūsų atveju intuityviai yra aišku, kad problema kyla dėl vadovėlių ir dėstytojų nepriklausomumo. Situacija labai pagerės išskaidžius S: S1(Kursas, Dėstytojas) ir S2 (Kursas, Vadovėlis). S1 ir S2 bus BKNF, o dekompozicija grįžtamoji.

1971 Ronaldas Feiginas (Ronald Fagin) teoriškai apibrėžė tokią situaciją. Mes panagrinėsime klausimą labai neformaliai. Tiesa, dar prieš norminant santykį reikėtų jį suskaidyti pagal pasikartojimo grupes, bet kad tai paaiškėtų, prireikė teorijos.

Pastebėsime, kad pavyzdyje negalėjome dekomponuoti pagal FP, nes jų paprasčiausiai nėra (išskyrus trivialias). Tačiau santykyje S yra dvi *daugiareikšmės priklausomybės*:
K $-\!\!\twoheadrightarrow$ D ir K $-\!\!\twoheadrightarrow$ V.

Daugiareikšmės priklausomybės (DP) apibrėžimas.
K $-\!\!\twoheadrightarrow$ D reiškia, kad, nors kiekvieno kurso neatitinka vienintelis D, t.y., nėra FP K $\rightarrow$ D, bet kursą atitinka *apibrėžtas* dėstytojų ar vadovėlių rinkinys, tuo tarpu dėstytojai ir vadovėliai vienas nuo kito nepriklauso. Analogiškai K $-\!\!\twoheadrightarrow$ V.

Tegu A, B ir C – santykio R atributų aibės. A $-\!\!\twoheadrightarrow$ B tada ir tik tada, kai B reikšmių aibė porai (A,C) priklauso tik nuo A, bet ne nuo C.

Beje, DP visada sukuria susijusias poras tokio tipo A $-\!\!\twoheadrightarrow$ B|C, pavyzdžiui, K $-\!\!\twoheadrightarrow$ D|V.
DP apibendrina FP ta prasme, kad FP yra DP dalinis atvejis kai atributų aibės sudarytos tik iš vieno elemento. Santykio S problema buvo ta, kad jame DP nebuvo FP.

Feigino (Fagin) teorema. Tegu A, B ir C – santykio R (A,B,C) atributų poaibiai. R lygus jo projekcijų (A, C) ir (A, B) sąjungai tada ir tik tada, kai santykyje R yra DP A $-\!\!\twoheadrightarrow$ B|C. tai Hezo teoremos griežtesnė versija.

Santykis yra 4NF (būtent ketvirtoji norminė forma, nes, jai atsiradus, dar nebuvo apibrėžta BKNF) tada ir tik tada, kai bet kurioms A ir B, kurioms galioja netriviali DB A $-\!\!\twoheadrightarrow$ B,visi R atributai yra *funkciškai priklausomi* nuo A.

Tai reiškia, kad santykyje gali būti tik FP arba DP tipo CK $\rightarrow$ X – visos daugiareikšmės priklausomybės yra funkcinės priklausomybės nuo CK.

4NF visada įmanoma ir gali būti gaunama be informacijos praradimų.

Penktoji norminė forma (5NF)

Iki šiol laikėme, kad vienintele naudojama ir leistina operacija norminimo procese yra santykio pakeitimas dviem jo projekcijomis neprarandant informacijos. Tai galiojo iki 4NF. Tačiau gali egzistuoti santykiai, kurių negalima be praradimų suskaidyti į dvi projekcijas, bet galima į tris ar daugiau. Tokie santykiai vadinami n-dekomponuojamais.

Pavyzdys

Santykyje **KPZ** (K, Z, P) aprašyti kartografai, žemėlapiai ir projektai. Jis visiškai raktinis ir yra 4NF.

Jį galima suskaidyti į tris santykius KZ, PZ ir KP

K	**P**	**Z**
K1	P1	Z1
K1	P2	Z1
K2	P1	Z1
K1	P1	Z1

K	**Z**
K1	Z2
K1	Z1
K2	Z1

P	**Z**
P1	Z2
P2	Z1
P1	Z1

K	**P**
K1	P1
K1	P2
K2	P1

Šiuos santykius jungiant, pastebėsime įdomų dalyką. Sujungus PZ ir KP pagal P, atsiras įrašas (K2, P1, Z2), kurio nebuvo pradiniame santykyje, tačiau sujungus rezultatą su KZ pagal (K, Z), jis pasinaikins ir gausime korektišką rezultatą – pradinį santykį KPZ.

Kokias savybes turi tenkinti santykis, kad jis būtų 3-dekomponuojamas?
Pasirodo, sąlyga yra ciklinė: jei

a) K1 sudarė žemėlapį Z1;
b) K1 dirba P1;
c) Žemėlapis Z1 sudarytas projektui P1,

tai

d) K1 sudarė žemėlapį Z1 projektui P1.

Beje, taip yra nebūtinai, kaip matėme iš ankstesnių pavyzdžių.

Toks apribojimas vadinamas sąjungos priklausomybe (SP). Ji apibendrina DP, ir, aišku, FP.

Tegu A, B … Z – santykio R atributų aibės. R tenkina sąjungos priklausomybės sąlygą *(A, B, … Z) tada ir tik tada, kai R yra lygus šių jo projekcijų sąjungai.

Gali kilti klausimas, ar toks dekomponavimas tikslingas. Atsakymas yra taip, nes tokiu būdu pašalinamos anomalijos.

Santykis yra 5NF tada ir tik tada, kai kiekviena sąjungos priklausomybė išplaukia iš CK, t.y., CK turi būti jos determinantas (įrodė Feiginas).

Matyti, kad KPZ *(KP, PZ, KZ), tačiau KPZ nėra 5NF, nes sąjungos priklausomybė neišplaukia iš rakto, kuris yra pats KPZ. Paėmę pavyzdį ***Kartografai*** (AsmKodas, Pavardė, Pareigos, Miestas), rasime jame SP *((AsmKodas, Pavardė, Pareigos),(AsmKodas, Miestas)). Bet ši priklausomybė išplaukia iš to, kad AsmKodas yra pirminis raktas, todėl šis santykis yra 5NF.

Teoriškai 5NF visada įmanoma ir gaunama be informacijos praradimų, tačiau procesas, kaip nustatyti, ar santykis yra 5NF ar nėra, ne visai aiškus. Vis dėlto, santykiai, kuriuose sąjungos priklausomybė nepriklauso nuo CK, praktiškai pasitaiko gana retai.

Norminimo procedūros schema.
Priminsime, kad nagrinėjome dekompoziciją be praradimų.
Tegu duotas santykis R, kuris yra 1NF arba į ją suvedamas, bei tam santykiui nurodytos FP, DP ir SP. Norminimo technologijos idėja – sistemingai paversti R mažesnių santykių rinkiniu, kuris informacijos prasme yra ekvivalentus R, bet dėl įvairių priežasčių labiau pageidautinas. Kiekvienas norminimo etapas – tai santykių, gautų ankstesniame etape, skaidymas į projekcijas. Apribojimai panaudojami parenkant tinkamas projekcijas.

Norminimo procedūros taisyklės (visas procesas gali ir turi vykti neprarandant informacijos).

1. 1NF santykis skaidomas į projekcijas taip, kad būtų panaikintos visos FP, kurios nėra nesuprastinamos. Gausime 2NF santykių rinkinį.
2. 2NF santykis skaidomas į projekcijas taip, kad nebeliktų tranzityvių FP. Gausime 3NF santykių rinkinį.
3. 3NF santykis skaidomas į projekcijas taip, kad neliktų FP, kurių determinantai nėra potencialūs raktai. Gausime BKNF santykių rinkinį.

Pirmaisiais trimis žingsniais išnaikinamos visos FP, kurių determinantas nėra CK.

4. BKNF santykis skaidomas į projekcijas taip, kad nebeliktų daugiareikšmių priklausomybių, kurios nėra FP (praktiškai tai dažnai padaroma anksčiau). Gausime 4NF santykių rinkinį.
5. 4NF santykis skaidomas į projekcijas taip, kad nebeliktų sąjungos priklausomybių, jei tik įmanoma jas aptikti, kurios nebūtų apibrėžtos potencialaus rakto. Gausime 5NF santykių rinkinį.

Norminimo procedūros tikslai.

1. Sumažinti informacijos perteklių.
2. Pašalinti atnaujinimo anomalijas.
3. Parengti maketą, kuris būtų „geras“ realybės modelis, t.y., intuityviai suprantamas ir tinkamas vystyti toliau.
4. Supaprastinti vientisumo apribojimų išraišką (iš vienų išplaukia kiti ir jie turi būti palaikomi automatiškai, kai tik sutvarkomi apribojimai potencialiems raktams).

Norminimo procedūra teikia tik rekomendacijas. Pasitaiko atvejų, kai nėra prasmės vykdyti ją iki galo.

Pavyzdys
Santykį **S** (Pavardė, Gatvė, Miestas, Rajonas, PaštoKodas), kuris yra 2NF, bet ne 3NF, galima būtų norminti išskaidant į du: S1 (PaštoKodas, Miestas, Rajonas) ir S2 (Pavardė, Gatvė, PaštoKodas), nes pašto indeksas nurodo miestą ir rajoną vienareikšmiškai. Tačiau pašto indeksai keičiami palyginti labai retai, tuo tarpu derinys Gatvė, Miestas, Rajonas dažniausiai naudojamas kartu formuojant adresą. Taigi, santykio S skaidymas apsaugotų nuo kai kurių atnaujinimo anomalijų, tačiau pailgintų paieškos operacijų laiką. Šiuo atveju skaidyti greičiausiai neverta.

Reliacinis skaičiavimas yra susijęs su duomenų reikšmėmis, o norminimas – su jų prasme. Programavimo kalbos ir procedūros formaliai nereikalauja norminimo, jis tik padeda teisingai interpretuoti duomenis. Dažnai galimos kelios vienodai geros dekompozicijos alternatyvos.

Norminimas nėra universali procedūra – ji padeda įvykdyti vientisumo apribojimus, bet negali padėti tais atvejais, kai apribojimai yra kitokie, neišreiškiami per FP.

Nors 5NF užbaigia norminimo procedūrą, galimos ir dar aukštesnės NF. Tokia, pavyzdžiui, yra domenų-raktų NF, kurioje kievienas apribojimas yra domenų ir raktų apribojimų loginė pasekmė. Dar yra apibrėžta išrinkimo-sąjungos NF, kuri numato santykių skaidymą pagal dažnai atliekamas išrinkimo operacijų grupes, pavyzdžiui gyventojai – į Vilniaus, Kauno ir kitus. Tačiau ši NF praktiškai visada yra blogesnė už 5NF.

🕮 Klausimai diskusijai

Pateikite įvairių santykių, kurie nėra ketvirtosios norminės formos, pavyzdžių. Patikrinkite, ar anksčiau sukurtose DB nėra daugiareikšmių priklausomybių ir sąjungos priklausomybių

✍ Užduotis savarankiškam darbui

Duoti tokie duomenys: ***Klientai*** (AsmKodas, Adresas, Balansas, Maksimalus kreditas, Nuolaida); ***Užsakymai*** (Klientas, Pristatymo adresas, Data), sudarantys užsakymo antraštę ir (Prekės Nr, Kiekis), sudarantys užsakymo eilutes; ***Prekės*** (Nr, Tiekėjas, Kiekis pas kiekvieną tiekėją, Aprašymas). Sukurkite duomenų bazės schemą, nurodykite pagrindines funkcines priklausomybes, sunorminkite.

Tarkime, kad labai nedaug (mažiau kaip vienas procentas) klientų turi daugiau, negu vieną adresą. Kaip išspręsti šią situaciją? Atitinkamai modifikuokite DB loginį modelį.

💻 Užduotys praktikos darbams

Patikrinkite, kokios norminės formos yra visos duomenų bazės, sudarytos atliekant ankstesnės užduotis. Jei reikia, atlikite norminimą iki 3NF.

Sukurkite užklausas *Giminės medžio* duomenų bazėje.

Išrinkti visiems žmonėms jų tėvų vardus bei pavardes.
Išrinkti susituokusiems vyrams jų uošvių vardus ir pavardes.
Išrinkti žmonėms, kurie turi vaikų, jų vardus (poromis *tėvas/motina—vaikas*).
Išrinkti žmonėms, kurie turi brolių ar seserų, jų vardus bei pavardes (poromis).
Išrinkti poras *senelis—anūkas, močiutė—anūkė*.
Išrinkti žmonėms, kurie turi sūnėnų, jų vardus bei pavardes (poromis *dėdė/teta–sūnėnas*).
Išrinkti pusbrolių/pusseserių poras.
Išrinkti žmones, kurie buvo sudarę daugiau kaip vieną santuoką ir jų santuokų skaičių..

X. LYGIAGRETUS DUOMENŲ NAUDOJIMAS

Vienu metu nuspaudus du klavišus, bus įvesta ta raidė, kurios klavišą užkliudėte netyčia.
Merfio dėsnis

Atkūrimas (angl. *recovery*) – tai duomenų bazės grąžinimas į korektišką būseną. Jis turi būti atliekamas, kai susidaro situacijos, galinčios sukelti problemas. Atkūrimo principas paprastas – informacijos perteklius (dubliavimas) fiziniame lygmenyje. Apskritai, atkūrimo ir transakcijos sąvokos nepriklauso nuo to, ar duomenų bazių valdymo sistema yra reliacinė, ar ne.

X.1 TRANSAKCIJOS

Transakcijos sąvoka yra viena svarbiausių duomenų bazių valdymo teorijoje. Tai – loginis sistemos darbo vienetas.
Pavyzdžiui, užklausa, kuri išrenka kartografus, sudariusius istorinių žemėlapių ir padvigubinanti jiems atlyginimo koeficientą, iš tikrųjų yra sudaryta iš dviejų operacijų: išrinkti ir pakeisti. Išrinkimas ar pakeitimas, atlikti vienas be kito, neturi prasmės, be to, tarp jų vykdymo duomenų bazės būsena gali laikinai tapti prieštaringa, kaip pamatysime vėliau.

Šios transakcijos vykdymas pseudokalba galėtų būti užrašytas taip.

```
BEGIN TRANSACTION
      SELECT Pavarde, AtlKoef FROM Darbuotojai WHERE Pareigos<>“Direktorius“;
      IF <klaida> …. GOTO undo
      UPDATE Darbuotojai SET AtlKoef = AtlKoef*2 WHERE Pareigos<>“Direktorius“;
      IF <klaida> …. GOTO undo
COMMIT TRANSACTION
      GOTO finish
undo: ROLLBACK TRANSACTION
finish: RETURN
```

Normaliomis sąlygomis yra įvykdomos abi operacijos. Tačiau negalima tikėtis, kad taip bus absoliučiai visada. Sistema gali sutrikti, bet ir tuo atveju būtina garantija, kad atsitikus klaidai viduryje transakcijos pakeitimai bus anuliuoti, t.y., įvykdoma viskas arba nieko.

Tuo rūpinasi transakcijų modulis, naudojantis instrukcijas COMMIT (patvirtinti) ir ROLLBACK (atšaukti). COMMIT signalizuoja, kad transakcija įvykdyta ir duomenų bazės būsena vėl yra korektiška, taigi, galima fiksuoti operacijos rezultatą. ROLLBACK signalizuoja apie klaidą ir nekorektišką DB būseną, t.y., kad reikia atšaukti visus jau atliktus pakeitimus. Tai galioja ir, pavyzdžiui, kaskadiniam atnaujinimui. Beje, ne visada galima aptikti klaidą kokiu nors paprastu testu, todėl turi būti atsižvelgta ir į nenumatyto atvejo galimybę. Pavyzdyje neįvykdžius COMMIT, pakeitimai bus atšaukti bet kuriuo atveju.

Reliacinėse sistemose, kurios leidžia daugelio lygių įdėtas transakcijas bei operuoja aibėmis, šios instrukcijos ir apskritai transakcijų valdymas yra labai svarbus.

COMMIT – tai taip vadinamas fiksacijos taškas (angl. *syncpoint*), rodantis, kad loginis darbo vienetas yra užbaigtas. Transakcija galioja tik tai DB daliai, kurią ji apima.

COMMIT ypatybės.

1. Visi rezultatai tampa nuolatiniais (iki tol jie būna tarsi „bandomieji“).
2. Visi kortežai tampa prieinamais kitoms transakcijoms.

Transakcijos turi pasižymėti taip vadinamomis ASIS savybėmis.

1. Atomiškumas (transakcijos yra neskaidomas).
2. Suderinamumas (transakcijos rezultatas yra korektiška DB būsena).
3. Izoliuotumas (transakcijos yra atskirtos viena nuo kitos)
4. Stabilumas (įvykdžius transakciją, rezultatai lieka išsaugoti net ir sistemai išsijungus sekančiu momentu).

Išoriniai įvykiai, nutraukiantys sistemos darbą gali būti

1. Sistemos (programiniai), kai prarandamas operatyviosios atminties turinys, pavyzdžiui, srovės dingimas. Tuo atveju panaudojamas OA turinys, kuris rašomas į specialų žurnalą. Jei reikia, atkūrus sistemą, transakcija pakartojama.
2. Diskų (aparatūriniai), kai prarandamas nuolatinės atminties turinys, pavyzdžiui, disko galvutės gedimas. Tada prarastą informaciją galima perrašyti nebent iš rezervinės kopijos.

SQL palaiko COMMIT ir ROLLBACK instrukcijas.

X.2 LYGIAGRETUMAS

Lygiagretus transakcijų vykdymas yra glaudžiai susijęs su saugumo problema. Lygiagretumas – tai DBVS galimybė vykdyti daug transakcijų vienu metu ir dirbant su tais pačiais duomenimis. Jei nebūtų atitinkamo valdymo, būtinai kiltų konfliktinės situacijos. Pagrindinis transakcijų valdymo metodas yra blokavimas. Tai labai plati tema, kuria paliesime tik paviršutiniškai.

Lygiagretumo idėja nepriklauso nuo to, ar DBVS reliacinė, ar ne, tačiau vystant jo teoriją daugiausiai orientuojamasi būtent į specializuotą reliacinį kontekstą.

Vykdant transakcijas, kartais galima gauti klaidingą rezultatą todėl, kad jos pakeičia viena kitos tuo pačiu metu naudojamus duomenis.

Trys lygiagretumo problemos.

1. Atnaujinimo rezultatų praradimas.
2. Neužfiksuota priklausomybė.
3. Nesuderinta analizė.

1. Atnaujinimo rezultatų praradimas.

Transakcija A	**Laikas**	**Transakcija B**
Skaityti kortežą k	t_1	
	t_2	Skaityti kortežą k
Atnaujinti kortežą k	t_3	
	t_4	Atnaujinti kortežą k
	…	

Transakcijos A rezultatas bus prarastas, nes laiko momentu t_4 bus įvykdytas atnaujinimas pagal transakciją B

2. Neužfiksuota priklausomybė.

Tai atsitinka, kai yra naudojamas, ar (dar blogiau) keičiamas kortežas, kurį tuo pat metu keičia kita transakcija. Kol tas keitimas nebaigtas, visada egzistuoja tikimybė, kad jis niekada nebus baigtas ir bus grįžta prie pradinės būsenos. Tuo atveju pirmoji transakcija naudos duomenis, kuriuos ji nuskaitė pakeistus antrosios transakcijos kažkokiu tarpiniu momentu, taigi, kurių nėra ir nebuvo duomenų bazėje (nes antroji transakcija atšaukta).

Transakcija A	**Laikas**	**Transakcija B**
	t_1	Atnaujinti kortežą k
Skaityti kortežą k	t_2	
	t_3	Atšaukti B
	…	

Transakcijos A rezultatas momentu t_2 tampa priklausomu nuo transakcijos B įvykdymo.

Dar blogiau:

Transakcija A	**Laikas**	**Transakcija B**
	t_1	Atnaujinti kortežą k
Atnaujinti kortežą k	t_2	
	t_3	Atšaukti B
	…	

Transakcijos A rezultatas momentu t_2 tampa priklausomu nuo transakcijos B įvykdymo, be to, laiko momentu t_3 dar ir prarandami transakcijos A rezultatai.

3. Nesuderinta analizė.

Sąskaita 1: 40 LT	**Sąskaita 2: 50 LT**	**Sąskaita 3: 30 LT**

Transakcija A	**Laikas**	**Transakcija B**
Skaityti S1, SUM =40	t_1	
Skaityti S1, SUM =90	t_2	
	t_3	Skaityti S3
	t_4	Keisti S3: 30 →20
	t_5	Skaityti S1
	t_6	Keisti S1: 40 →50
	t_7	Baigti B
Skaityti S3, SUM =110	t_8	
(Turėjo būti 120!)		

Gautas klaidingas transakcijos A rezultatas dėl įsiterpusios transakcijos B. Todėl duomenų bazės būsena tampa nekorektiška.

X.3 BLOKAVIMAS

Lygiagretumo problemas galima išspręsti valdant lygiagretų procesų vykdymą. Blokavimo metodikos idėja tokia: jei kokiai nors transakcijai reikia, kad kortežas nebūtų neprognozuojamai pakeistas jos darbo metu, tokį kortežą (ar kitą objektą) reikia blokuoti.

Blokuotas įrašas tampa neprieinamas kitoms transakcijoms tol, kol blokavusi transakcija pasibaigs. Objektas lieka pirmosios transakcijos dispozicijoje tiek laiko, kiek reiks. Blokas gali būti dviejų tipų.

1. Išskirtinis (angl. *eXclusive lock*) x-blokas. Jis naudojamas, kai įrašas keičiamas ir uždraudžia kitoms transakcijoms bet kokį to įrašo naudojimą.
2. Skaitymo (angl. *Shared lock*) s-blokas. Jis naudojamas, kai įrašas skaitomas ir uždraudžia kitoms transakcijoms to įrašo keitimą.

Jeigu transakcija A naudoja kortežą blokuodama jį X tipo bloku, tai kitos transakcijos B reikalavimas blokuoti tą patį kortežą bus atmestas ir B pereis į laukimo būseną.

Jeigu transakcija A naudoja kortežą blokuodama jį S tipo bloku, tai kitos transakcijos B reikalavimas blokuoti tą patį kortežą bus:

Jei B reikia s-blokuoti įrašą , reikalavimas bus patenkintas;

Jei B reikia x-blokuoti įrašą , reikalavimas bus atmestas ir B pereis į laukimo būseną.

Duomenų naudojimo protokolas yra toks.

1. Transakcija, skaitanti kortežą, turi prieš tai jį s-blokuoti.
2. Transakcija, keičianti kortežą, turi prieš tai jį x-blokuoti

Paprastai tai atliekama automatiškai: pradedant transakciją blokas uždedamas, įvykdžius – nuimamas.

Jeigu transakcijos reikalavimas blokuoti kortežą atmetamas (taip atsitinka, kai tą kortežą jau yra blokavusi kita transakcija), ji pereina į laukimo būseną, kol ankstesnės transakcijos blokas nebus nuimtas jai pasibaigus arba ją atšaukus. Yra būdai išvengti be galo ilgo laukimo, pavyzdžiui, sudaromos transakcijų FIFO (angl. *first in, first out*) eilės.

Transakcija A	**t**	**Transakcija B**
Skaityti kortežą k (s-blokas)	t1	
	t2	Skaityti kortežą k (s-blokas)
Atnaujinti kortežą k (x-blokas)	t3	
	t4	Atnaujinti kortežą k (x-blokas)
Laukti	t5	Laukti
Laukti	t6	Laukti
	…	

Transakcija A laukia nuo laiko momento t3, nes yra transakcijos B uždėtas blokas. Analogiškai transakcija B laukia nuo momento t4. Rezultatai neprarandami, bet susidariusi situacija yra nepageidautina.

Taigi, nors blokavimas padeda spręsti tris lygiagretumo problemas, jis gali sukelti kito tipo problemą. Probleminė (“aklavietės“) situacija susidaro, kai tą patį objektą blokuoja dvi ar daugiau transakcijų ir kiekviena laukia, kol baigsis kita.

Tokios situacijos aptinkamos naudojant laukimo būsenų diagramą. Viena iš laukiančių transakcijų turi būti „paaukojama“, t.y., atšaukiama tam, kad galėtų įvykti kita. Kartais vietoje laukimo būsenų diagramos naudojamas paprastas chronometražas, t.y., transakcija nutraukiama, jei ji per ilgai neįvyksta. Pageidautina, kad „paaukota“ transakcija būtų pakartota, užtikrinant, kad nesusidarytų sąlygos, privedusios prie aklavietės.

Blokai gali būti ir kitų tipų, jei jie taikomi ne vienam kortežui, o kitiems duomenų bazės objektams, pavyzdžiui, santykiams.

Transakcijų aibės tvarkymas.

1. Transakcija laikoma teisinga, jei ji perveda duomenų bazę iš vienos neprieštaringos būsenos į kitą neprieštaringą būseną.
2. Iš to išplaukia, kad transakcijos gali vykti viena po kitos bet kokia tvarka, jei tik jos nepriklausomos.
3. Transakcijų grupė laikoma teisinga, jei ji ekvivalenti nuosekliai transakcijų sekai, t.y., yra sutvarkoma. Jei ji tokia nėra, tenka tvarkyti per prievartą, verčiant transakcijas laukti.

Beje, dviejų nuosekliai įvykdytų transakcijų rezutatas priklauso nuo jų tvarkos, pavyzdžiui, jei transakcija A reiškia x+1, o transakcija B – x*2, tai seka A,B duos rezultatą (x+1)*2, o seka B,A – 2x+1.

Transakcijų tvarkos sampratą pasiūlė K. Esvaranas (Kapali Eswaran), išreiškęs ją dviejų fazių blokavimo teorema:
Jei visos transakcijos yra pavaldžios dviejų fazių blokavimo protokolui, jas galima nuosekliai sutvarkyti.
Dviejų fazių blokavimo protokolas:

1. Prieš pradėdama darbą su objektu, transakcija jį blokuoja (blokavimo fazė).
2. Pabaigus darbą su objektu, transakcija neuždeda jokių kitų blokų (blokavimo nuėmimo fazė).

Transakcijos tvarkomos logiškai, t.y., jei transakcija B naudoja A rezultatus, tai pirma turi įvykti *visa* transakcija A.

Izoliacijos lygmenys.
Izoliacijos lygmuo rodo, kiek kitos transakcijos gali įsiterpti į duotosios transakcijos darbą. Jei transakcijų aibė sutvarkyta, jokio įsiterpimo nebus, t.y., izoliacijos lygmuo yra maksimalus.

Realiose sistemose paprastai leidžiamos transakcijos, kurių izoliacijos lygmuo žemesnis. Iš principo ta pati transakcija gali dirbti įvairiais izoliacijos lygmenimis skirtingose DB dalyse, bet dėl paprastumo laikysime, kad izoliacijos lygmuo yra nekintama transakcijos savybė. SQL standarte yra numatyti keturi izoliacijos lygmenys, DB2 sistemoje – tik du. Kuo aukštesnis izoliacijos lygmuo, tuo mažesnė transakcijų lygiagretumo galimybė.

X.4 PASKIRSTYTOS DUOMENŲ BAZĖS

Duomenys gali būti laikomi skirtinguose kompiuteriuose, sujungtuose į tinklą. Tą temą jau palietėme, kai kalbėjome apie kliento-serverio architektūrą. Buvo minėta skaidrumo sąvoka, reiškianti, kad turi būti galima dirbti su skirtingomis DBVS, operacine sistema, technine įranga, nekreipiant į tai dėmesio.

Dabar laikysime, kad paskirstyta duomenų bazė (PDB) – tai rinkinys *mazgų*, sujungtų komunikacijų tinklais. Kiekvienas mazgas turi savo atskirą DBVS. Laikysime, kad mazgai dirba sinchroniškai. Taigi, PDBVS yra tik virtualus objektas, loginis vienetas, bet ne fiziškai egzistuojanti sistema. Yra kelios PDB paskirstymo strategijos:

2. Horizontalus paskirstymas (eilučių grupės, pvz., gyventojų registras – vieno miesto gyventojai – tame mieste saugomi)
3. Vertikalus paskirstymas (stulpelių grupės)
4. Mišrus

Kiekviename PDBVS mazge turi būti komponentas, koordinuojantis bendrą mazgų darbą. Paprastumo dėlei laikysime, kad kiekviename mazge yra ta pati DBVS (“griežto homogeniškumo“ principas). PDBVS, kaip taisyklė, yra reliacinės, nes reliacinis modelis gerai atitinka PDB modelį.

Kodėl reikia PDB:

- Dauguma stambių įmonių turi geografiškai paskirstytus filialus.
- PDB leidžia pasiekti geriausią efektyvumo ir duomenų pasiekiamumo pusiausvyrą.

Fundamentalus PDB principas teigia, kad naudotojui paskirstyta DBS turi atrodyti taip pat, kaip **ne**paskirstyta.
Iš šio principo išplaukia PDB tikslai, savybės ir reikalavimai.

1. **Lokalus autonomiškumas**. Naudotojo informacija stotyje lieka įslaptinta net ir jam dirbant tinkle, vietiniais duomenimis operuojama vietoje, darbas mazge nepriklauso nuo kitų mazgų. Absoliutaus vietinio autonomiškumo pasiekti neįmanoma, bet siekiama, kad jis būtų maksimalus.
2. **Nepriklausymas nuo centrinio mazgo**. Visi mazgai laikomi lygiareikšmiais, niekas nėra sutelkta viename centre. Esant vienam „centriniam“ mazgui, jis būtų silpna sistemos grandis, be to, jam apkrova būtų neproporcingai didelė.
3. **Nepertraukiamas veikimas**. Tai reiškia patikimumą (bet kuriuo momentu viskas vyksta teisingai) ir prieinamumą (sistemos mazgai neišjungiami, net, pavyzdžiui, keičiant sistemos konfigūraciją).
4. **Nepriklausymas nuo vietos**. Vietos „permatomumas“ reiškia, kad naudotojui neturi rūpėti, kur fiziškai yra kuri DB dalis.
5. **Nepriklausymas nuo fragmentacijos**. Fragmentacija gali būti vertikali ir horizontali. Naudotojas neturi jos pastebėti – jo naudojami fragmentai laikinai sujungiami (tuo rūpinasi optimizatorius).
6. **Nepriklausymas nuo replikavimo**. Jei yra kelios duomenų bazės objekto kopijos keliuose mazguose, naudotojas neturi to matyti. Kopijos daromos dėl dviejų priežasčių, pirma, vietinės kopijos dirba greičiau, antra, naudotojai turi daugiau galimybių prieiti prie duomenų.
7. **Paskirstytų užklausų palaikymas**. DBVS turi surinkti duomenis iš skirtingų mazgų, be to, naudotojas to nejaučia. Paskirstytos užklausos turi būti optimizuojamos, nes, esant daug skirtingų galimybių joms įvykyti, tai gali trukti, pavyzdžiui, nuo 0.1 sekundės iki 6 valandų.
8. **Paskirstytų transakcijų valdymas**. PDB ypač svarbios problemos yra lygiagretumas ir duomenų atkūrimas (vienu metu galima keisti duomenis keliuose mazguose). Transakcijas PDB sudaro agentai (procesai, transakcijos nurodymu vykdomi įvairiuose mazguose). Atkūrimas paremtas dvifaze fiksacija.
9. **Nepriklausymas nuo aparatūros** (Macintosh, IBM PC, RISC platformų).
10. **Nepriklausymas nuo OS** (Windows, MacOS, Linux, Unix, MVS).
11. **Nepriklausymas nuo DBVS** (Access, Oracle, Informix – duomenų bazės tik turi būti reliacinės ir palaikyti SQL). Jos turi turėti vienodą naudotojo sąsają, kuri pasiekiama naudojant šliuzų technologiją. Apskritai, tai nėra privaloma, tačiau pageidautina PDB savybė.
12. **Nepriklausymas nuo tinklo architektūros.**

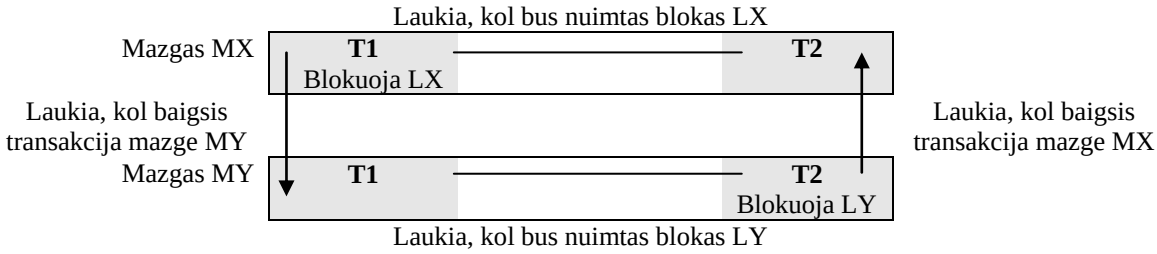
PDBVS problemos.
Siekiant minimizuoti tinklo apkrovą, kyla papildomų problemų.

1. **Užklausų vykdymas** (jis reikalauja vietinės ir globalios optimizacijos – ką kur perkelti, kur vykdyti, kokia tvarka, ar įmanoma tą daryti lygiagrečiai ir t.t.)
2. **Katalogo valdymas ir paskirstymas**. Galimi keli katalogo paskirstymo variantai:
    a) centrinis (visas vienoje vietoje) – tuo atveju pažeidžiamas antrasis reikalavimas;
    b) visiškai replikuotas (kiekviename mazge saugoma viso katalogo kopija) – pažeidžiamas pirmasis reikalavimas;
    c) sekcijinis (kiekviename mazge saugomas jo objektų katalogas) – neoptimalus variantas;
    d) a) ir c) derinys.

3. **Paskirstytas atnaujinimas**. Problema čia yra ta, kad reikia nuolat atnaujinti visas egzistuojančias kopijas visuose mazguose, tuo tarpu kuris nors mazgas atnaujinimo momentu gali būti laikinai neprieinamas. Sprendimo schema yra tokia.
   - Pirminės skirtingų objektų kopijos yra skirtinguose mazguose.
   - Atnaujinimo operacija laikoma baigta, kai yra atnaujintos visos pirminės kopijos.

   Tada mazgas, kuriame yra pirminė kopija, tampa atsakingu už antrinių kopijų atnaujinimą. Iš tiesų, ši operacija panaši į transakciją – turi būti ASIS savybės:
   1. Atomiškumas (atnaujinimas yra neskaidomas).
   2. Suderinamumas (atnaujinimo rezultatas yra korektiška DB būsena).
   3. Izoliuotumas (atnaujinimo operacijos yra atskirtos viena nuo kitos)
   4. Stabilumas (įvykdžius atnaujinimo operaciją, rezultatai lieka išsaugoti net ir sistemai išsijungus sekančiu momentu).

   Deja, taip yra pažeidžiamas vietinio autonomiškimo principas.
4. **Atkūrimo valdymas.** Naudojant dvifazės fiksacijos protokolą, nelabai aišku, kuris mazgas turi koordinuoti visą transakciją. Paprastai tai būna mazgas, kuris ją inicijuoja. Taip vėl pažeidžiamas vietinio autonomiškimo principas. Be to, tinklas gali gesti ir neįmanoma išvengti su tuo susijusių problemų.
5. **Lygiagretumo valdymas.** Dirbant keliuose mazguose, atitinkamai padaugėja blokavimo ir blokavimo atsisakymo operacijų. Vėl kyla ir kopijų problema. Pirminė ir visos antrinės kopijos turi būti blokuojamos kaip vientisas objektas, taip sumažinant bendrą blokavimo operacijų skaičių. Gali susidaryti ir globali aklavietė, t.y., laukimo ciklas, einantis per keletą mazgų. Tuo atveju lokaliose transakcijų laukimo diagramose ciklų nėra ir konkrečiame mazge aptikti paveiksle pavaizduotos situacijos neįmanoma. Reikia atlikti papildomą analizę.



PDBVS atvejis yra kliento-serverio sistemos, kai visi duomenys saugomi tik mazguose-serveriuose, programos vykdomos tik mazguose-klientuose, be to, klientai atskirti nuo serverių ir naudotojas tą mato.

🕮 Klausimai diskusijai

Aptarkite blokavimo metodo privalumus ir trūkumus, galimas alternatyvas. Paieškokite Internete informacijos apie paskirstytų duomenų bazių reikšmę, tokių duomenų bazių raidą ir perspektyvas. Kaip paskirstytos DB susiję su debesų kompiuterija?

✍ Užduotis savarankiškam darbui

Sudarykite paskirstytos pagrindinio 1:1000 mastelio Lietuvos geografinių duomenų bazės projektą. Numatykite, kaip turi būti naudojami šie duomenys ir kaip užtikrinamas jų saugumas.

XI. DUOMENŲ SAUGA

Jei yra galimybė padaryti klaidą, anksčiau ar vėliau klaida bus padaryta.
Berklio taisyklė

XI.1 DUOMENŲ SAUGOS PROBLEMA

Duomenų saugos tikslas – užtikrinti, kad duomenų bazėje (ar apskritai organizacijoje) kaupiama ir naudojama informacija būtų patikima ir apsaugota nuo atsitiktinio ar neteisėto sunaikinimo, pakeitimo, atskleidimo, kokio nors kitokio neteisėto jos tvarkymo.

Svarbiausi duomenų saugos rizikos veiksniai yra:

- netyčiniai įvykiai (duomenų tvarkymo klaidos ir apsirikimai, duomenų ištrynimas, klaidingas duomenų teikimas, fiziniai informacijos technologijų sutrikimai, duomenų perdavimo tinklais sutrikimai, programinės įrangos klaidos, netinkamas veikimas ir pan.);
- tyčiniai žmogaus sukelti įvykiai (nesankcionuotas naudojimasis informacine sistema duomenims gauti, duomenų pakeitimas ar sunaikinimas, duomenų perdavimo tinklais sutrikdymai, sabotažas, vagystės, įsilaužimai, virusai, kiti saugumo pažeidimai);
- gamtinės nelaimės, pavyzdžiui, gaisras, potvynis, žemės drebėjimas ir pan.; nenugalima jėga (*force majeure*).

Labiausiai tikėtina, kad saugos problemos atsiras dėl programų klaidų, sistemos sutrikimų, įsibrovėlių atakos bei žmonių klaidos. Tačiau yra ir kitų pavojų. Pavyzdžiui, teikiant paslaugas debesų kompiuterijos technologijomis naudotojų duomenys tvarkomi naudojant bendrą programinę įrangą, bendrus elektroninių ryšių išteklius, o informacija yra saugoma nuotoliniuose serveriuose (tinkle). Šių technologijų naudotojai nežinokokiuose serveriuose, kuriose šalyse saugomi jų duomenys, taip pat neaišku, kas juos administruoja ir kokiais tikslais tvarko. Taip atsiranda pavojus duomenų saugumui.

Duomenų saugos rizikos veiksniai gali būti sugrupuoti pagal duomenų svarbos kategorijas (duomenys priskiriami atitinkamai kategorijai atsižvelgiant į jų konfidencialumą ir svarbą infromacinės sistemos veiklai) ir pagal galimo poveikio apimtį.

- Mažas poveikis. Duomenų pažeidimo padariniai nepavojingi – pavyzdžiui, informacija pasiųsta kitam adresatui, įvesti netikslūs duomenys, dingo dalis informacijos, prarasta informacija po paskutinio kopijavimo, sugadinta operacinė sistema.
- Vidutinis poveikis. Duomenų pažeidimo padariniai rimti – duomenys netikslūs ar visiškai sugadinti, duomenų bazių įrašai neteisingi, sunku aptikti klaidas ar prieštaringus faktus, nebefunkcionuoja DBVS.
- Didelis poveikis. Duomenų pažeidimo padariniai kritiški – duomenys visiškai sugadinti, dėl vagystės, gaisro ar užliejimo prarasti ne tik duomenys iš duomenų bazių, bet ir atsarginės kopijos, dėl to neveikia visa informacinė sistema.

Svarbiausi duomenų saugumo problemos aspektai ir saugumo užtikrinimo būdai.

- Teisiniai, visuomeniniai ir etiniai (pavyzdžiui, ar konkretus asmuo turi teisę gauti norimą informaciją, tarkime, apie banko sąskaitas).
- Fizinės sąlygos (pavyzdžiui, ar apsaugotas kompiuteris, ar rakinamas kambarys, kuriame jis yra).
- Organizacinės sąlygos (kaip organizuotas priėjimas prie duomenų).
- Valdymas (kokiu lygmeniu palaikomos saugumo priemonės).

- Operacinės sistemos (pavyzdžiui, ar sistema ištrina duomenis, kai darbas su jais baigtas).
- Naudojimo teisių suteikimo ir skirstymo strategija.

Bendruosiuose elektroninės informacijos saugos valstybės institucijų ir įstaigų informacinėse sistemose reikalavimuose[18] numatyta, kad IS saugaus elektroninės informacijos tvarkymo ir naudotojų administravimo taisyklėse turi būti:

- informacinėje sistemoje esančios informacijos kategorijų sąrašas ir kiekvienai kategorijai priskirtini duomenys;
- techninių ir kitų saugos priemonių aprašymas, apimantis kompiuterinės įrangos, sisteminės ir taikomosios programinės įrangos, duomenų perdavimo tinklais saugumo užtikrinimo, patalpų ir aplinkos (įėjimo kontrolė, elektros tiekimas, aplinkos drėgnumas, darbo vietos temperatūra, priešgaisrinė sauga), informacinės sistemos darbo apskaitos ir kitas priemones informacijos saugai užtikrinti;
- informacinės sistemos saugą užtikrinantys reikalavimai, keliami informacinių sistemų funkcionavimui reikalingoms paslaugoms (patalpos, įrangos ir sistemų priežiūra);
- informacinės sistemos ir duomenų vientisumo pažeidimų fiksavimo ir pažeistų duomenų atkūrimo tvarka (naudotojų veiksmų registravimas, atsarginės duomenų kopijos, jų saugojimas, saugojimo kontrolė ir kita);
- saugaus duomenų perkėlimo ar perdavimo tvarka;
- duomenų perdavimo tinklais reikalavimai;
- saugaus duomenų teikimo informacinės sistemos naudotojams kontrolės tvarka.

Yra dvi pagrindinės **duomenų saugos sritys**, kurias aptarsime išsamiau:

- duomenų konfidencialumo užtikrinimas (apsauga nuo neteisėto panaudojimo, taip pat asmens duomenų apsauga);
- duomenų išsaugojimas ir pasiekiamumo užtikrinimas nenumatytų įvykių atveju.

Lietuvoje rekomendacijas, susijusias su duomenų sauga nuolat skelbia Valstybinė duomenų apsaugos inspekcija (http://www.ada.lt).

XI.2 DUOMENŲ APSAUGA NUO NETEISĖTO PANAUDOJIMO

XI.2.1 Naudotojų teisių valdymas

Dabartinėse DBVS naudojamas vienas iš dviejų paplitusių metodų saugumui palaikyti.

1. Atrankos metodas. Naudotojas turi skirtingas privilegijas dirbti su skirtingais DB objektais (santykiais, objektais, atributais). Skirtingi naudotojai turi skirtingas teises. Tokios sistemos yra lanksčios.
2. Klasifikacinis valdymas. Kiekvienas duomenų objektas turi klasifikacinį lygmenį o kiekvienam naudotojui nurodytas priėjimo lygmuo. Prie konkretaus objekto gali prieiti tik atitinkamą lygmenį turintis naudotojas. Tai statiškos, nelanksčios sistemos.

Nepriklausomai nuo to, kokia saugumo schema naudojama, visi sprendimai priimami ne techniniu, o strateginiu lygmeniu. DBVS tik palaiko priimtus sprendimus. Iš to išplaukia kelios išvados.

1. Strateginių sprendimų rezultatai turi būti išsaugoti sistemoje saugumo taisyklių pavidalu.
2. Užklausos turi būti atitinkamai analizuojamos, t.y., nagrinėjami deriniai objektas-operacija-naudotojas. Tą darbą atlieka DBVS saugumo posistemė.

[18] http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=398479

3. Kad sistema galėtų taikyti taisykles, turi būti atpažintas užklausos šaltinis, t.y., įvestas naudotojo identifikatorius ir slaptažodis, pateisinantis kreipimąsi į DBVS.

Sistemos paprastai palaiko ne tik atskirus naudotojus, bet ir jų grupes.

Sistema, prieš leisdama naudotojui atlikti kokius nors veiksmus, privalo patikrinti, ar tie veiksmai yra teisėti. Duomenų saugaus naudojimo taisykles nurodo duomenų bazės administratorius programavimo kalba. Jos saugomos atskirame faile. Aprašant saugumo taisykles naudojama tokio tipo sintaksė.

```
CREATE SECURITY RULE <taisyklės vardas>
        GRANT RETRIEVE (AsmKodas, Pavardė), DELETE
        ON Darbuotojai WHERE Darbuotojai.Pareigos<>“Direktorius“
        TO Buhalterė
        ON ATTEMPTED VIOLATION REJECT;

DELETE SECURITY RULE <taisyklės vardas>.
```

Naudojimo taisyklių pažeidimas turi sukelti atitinkamus veiksmus. Veiksmai gali būti užprogramuoti bet kokie, bet dažniausiai tiesiog uždraudžiama įvykdyti užklausą.

Privilegijos gali būti teikiamos skirtingiems veiksmams RETRIEVE, INSERT, UPDATE, DELETE, ALL.

Neapeinamų saugumo sistemų negali būti iš principo. Jei duomenys labai svarbūs, galima ir reikia sekti sistemos su jais vykdomas operacijas. Fiksuojama užklausa, terminalas, naudotojas, data ir laikas, objektai, paliesti operacijos, senos reikšmės, naujos reikšmės. Tai – atrankos metodo darbas.

Didelės ir nelanksčios struktūros, pavyzdžiui, karinės ir vyriausybinės, naudoja klasifikacinį valdymą. Kiekvienam objektui nustatomas klasifikacinis lygmuo: „tarnybiniam naudojimui“, „slaptai“, „visiškai slaptai“ ir pan.

Klasifikacinio valdymo taisyklės yra paprastos.

- Naudotojas X gali prieiti prie objekto Y tik jei jo lygmuo yra ne žemesnis už objekto klasifikacinį lygmenį.
- Naudotojas X gali keisti objektą Y tik jei jo lygmuo yra *lygus* objekto klasifikaciniam lygmeniui (kitaip jo įrašyta informacija automatiškai būtų klasifikuojama kaip X lygmens, taigi, galėtų atsitikti taip, kad X įrašytų „slaptus“ duomenis į „mažiau slaptą“ objektą).

Pagal klasifikacinio valdymo reikalavimus ir instrukcijas, kurias paprastai rengia šalių saugumo insitucijos, atliekamas auditas, ieškoma blogai apsaugotų duomenų – tuo rūpinasi duomenų saugos administratorius.

XI.2.2 Duomenų perdavimas tinklais ir šifravimas

Iki šiol kalbėjome apie situaciją, kai naudotojas bando nelegaliai patekti į duomenų bazę per DBVS. Tačiau sistemą galima nesunkiai apeiti, pavyzdžiui, fiziškai kopijuojant diskų informaciją, perimant tinklu siunčiamus duomenis ir pan.

Pagrindinė tinklo apsaugos strategijos taisyklė – kuo labiau apriboti išorinio nuotolinio vartotojo teises. Paprasčiausia priemonė – **užkarda,** dar vadinama ugniasiene (angl. *firewall*) yra programa arba techninės įrangos dalis, kuri padeda apsisaugoti nuo įsilaužėlių per tinklą. Programinė užkarda

analizuoja tinklo srautą, identifikuoja ir praleidžia ar blokuoja informaciją, siunčiamą iš pavojingų ar įtartinų šaltinių. Galimos užkardos konfigūracijos:

- filtrų naudojimas (informacija filtruojama ir kompiuteris pagal taisykles arba atmeta, arba priima gaunamus duomenis),
- dvipusiai vartai (programinė įranga – vienintelis dviejų (saugomo ir nesaugaus) tinklų tarpininkas),
- stebimas potinklis (sukuriamas tinklas su užkardos komponentais tarp vidinio ir išorinio tinklų).

Užkardos negali visiškai apsaugoti nuo neteisėto įsiterpimo į duomenų perdavimo linijas. Pažeidėjas gali pasinaudoti duomenimis juos pasisavindamas ar sugadindamas. Jeigu informacija yra siunčiama tinklu nekoduota, ji gali būti lengvai perskaitoma, naudojant tinklo analizės priemones. Efektyviausias apsaugos būdas šiuo atveju yra **duomenų šifravimas**. Šifravimo ir dešifravimo procesas apsunkina darbą, todėl jis atliekamas tik su svarbiais duomenimis.

Naudojami įvairūs šifravimo algoritmai, bet visi jie priklauso nuo šifro rakto. Bendra schema yra tokia: perduodamas tekstas skaidomas į rakto ilgio gabalus, raidė keičiama jos kodu, kiekvienas gabalas sudedamas su taip pat užkoduotu raktu bei paimama sumos dalybos iš 7 liekana. Tada vėl atkeičiama į raides. Jei raktas žinomas, tą atlikti yra paprasta, jei ne – belieka bandyti atspėti perrenkant variantus. Problema ta, kad norint iššifruoti, taip pat reikia žinoti raktą.

Skaičiuojama, kad sąnaudos dokumentui iššifruoti būtų didesnės už įsilaužimo atneštą naudą. Tuo pagrįsta naujesnė „atviro rakto“ arba **asimetrinio šifravimo** metodika. Jos esmė tokia, kad yra du raktai: vienas, skirtas šifravimui, yra visiems prieinamas (viešas); kitas – skirtas iššifravimui – slaptas (privatus). Užšifruoti informaciją gali bet kas, turintis viešąjį raktą, o iššifruoti – tik privatų raktą turintis asmuo (be jo iššifruoti negali net pats informaciją užšifravęs asmuo).

Algoritmas pagrįstas tuo, kad atpažinti pirminį skaičių sąlygiškai lengva (pavyzdžiui, jei skaičius iš 130 skaitmenų, nustatyti, ar jis pirminis, naudojant kompiuterį prireiks apie septynių minučių). Tuo tarpu nėra efektyvaus algoritmo išskaidyti skaičių į pirminius daugiklius (jei skaičius iš 60 skaitmenų, jam išskaidyti prireiks 10^{17} metų!). Raktai yra tiesiog dideli pirminiai skaičiai ir šifruojant operuojama jų sandauga. Taigi, žinant viešąjį raktą nėra praktinės galimybės nustatyti su juo susijusį privatų raktą.Toks šifravimas naudojamas ir elektroninio parašo sistemose.

Plačiausiai paplitęs būdas perduodamų duomenų saugumui užtikrinti yra saugaus duomenų perdavimo protokolo *HTTPS* (angl. *Hypertext Transfer Protocol Secure*) naudojimas. *HTTPS* jungia *HTTP* (angl. *Hypertext Transfer Protocol*) duomenų perdavimo protokolą ir SSL/TLS (angl. *Secure Sockets Layer / Transport Layer Security*) protokolą, kuriuos naudojant identifikuojamas Interneto serveris ir perduodama šifruota informacija, todėl užkertamas kelias jos neteisėtam naudojimui. SSL sertifikatą galima susikurti patiems arba užsisakyti SSL sertifikatus išduodančioje įmonėje. Abiem atvejais jį turi patvirtinti įgaliota trečioji šalis (angl. *Certificate Authority*). Tai yra mokama paslauga, kurią galima užsisakyti beveik visose prieglobos paslaugas teikiančiose įmonėse. *HTTPS* protokolas duomenų perdavimo metu taip pat naudoja asimetrinį šifravimą.

HTTPS protokolo veikimas.

Tarnybinėje stotyje įdiegiamas viešasis ir privatusis raktas. Viešasis raktas, skirtas informacijai užšifruoti, yra platinamas viešai, o privatusis, skirtas informacijai iššifruoti, lieka slaptas.

Atvertus tinklalapį per *HTTPS* protokolą, tarnybinė stotis, kurioje yra tinklalapis, atsiunčia į naršyklę to tinklalapio viešąjį raktą, skirtą informacijai šifruoti.

Interneto naršyklė susisiekia su trečiąja šalimi ir palygina viešąjį raktą, gautą iš tarnybinės stoties, su viešuoju raktu, kurį turi trečioji šalis. Jei viešasis raktas registruotas, Interneto naršyklė visada gauna patvirtinimą apie rakto galiojimą ir apie tai, kam jis priklauso. Jei raktas yra neregistruotas, naršyklė pateikia pranešimą, rekomenduojantį nutraukti darbą šiame tinklalapyje. Interneto naršyklė leidžia peržiūrėti viešojo sertifikato informaciją, tačiau sužinoti, ar sertifikatas yra tikras, ar padirbtas, be trečiosios šalies patvirtinimo nėra galimybės.

Jeigu raktai sutampa, visa internetu perduodama informacija yra užšifruojama šiuo raktu. Informaciją iššifruoja privatusis raktas, kuris yra tarnybinėje stotyje.

2001 m. gruodžio 22 d. Lietuvos Respublikos Vyriausybė priėmė nutarimą Nr.1625 „Dėl informacijos technologijų saugos valstybinės strategijos ir jos įgyvendinimo plano“, kuriame numatyta valstybės institucijų kompiuterinio tinklo pagrindu sukurti **saugų valstybinį duomenų perdavimo tinklą** (SVDPT, http://www.svdpt.gov.lt/). 2004 metais SVDPT buvo sujungtas su Europos administracijų duomenų perdavimo tinklu TESTA (angl. *Trans-European Telematics Networks for Administrations*) ir Lietuvos valdžios institucijos pradėjo keistis duomenimis su ES institucijomis.

SVDPT tinklas yra atskirtas nuo bendrojo naudojimo tinklų (Interneto). Jame funkcionuoja atskira domenų vardų sistema, atskiras tarpžinybinis elektroninis paštas ir kitos sistemos. Duomenų perdavimo saugumas SVDPT užtikrinamas naudojant atskiras tinklo sritis skirtingo saugumo lygio informacijai perduoti, duomenų šifravimą ir duomenų mainų dalyvių identifikavimą. Duomenų perdavimo iš vienos tinklo srities į kitą taisyklės aprašomos užkardose, jungiančiose atskiras tinklo sritis. Svarbiems duomenims galima suteikti prioritetus ir apsaugoti duomenų srautus, perduodamus konkrečioje tinklo srityje bei informacines sistemas, prijungtas prie tam tikros tinklo srities. Todėl SVDPT leidžia saugiai ir efektyviai keistis informacija tarp Lietuvos ir Europos institucijų bei jų struktūrinių padalinių ir sumažinti išlaidas duomenų saugos priemonėms.

XI.2.3 Asmens duomenų apsauga

Asmens duomenys – tai informacija, susijusi su gyventojais, duomenų subjektais, kurių tapatybė yra žinoma arba gali būti tiesiogiai ar netiesiogiai nustatyta pasinaudojant tokiais duomenimis kaip asmens kodas, vienas arba keli asmeniui būdingi fizinio, fiziologinio, psichologinio, ekonominio, kultūrinio ar socialinio pobūdžio požymiai.

Europos Sąjungos pagrindinių teisių chartija skelbia, kad kiekvienas žmogus turi teisę būti informuotas apie savo asmens duomenų naudojimą, taip pat turi turėti galimybę juos ištaisyti, ištrinti, sustabdyti jų tvarkymą. Asmens sutikimas tvarkyti jo asmens duomenis turi būti savanoriškas.

Asmens duomenų teisinės apsaugos įstatymo[19] įgyvendinimą Lietuvoje prižiūri Valstybinė duomenų apsaugos inspekcija, kuri gina žmogaus privataus gyvenimo neliečiamumo teisę, kai yra tvarkomi jo asmens duomenys.

[19] http://www3.lrs.lt/pls/inter3/dokpaieska.showdoc_l?p_id=231799

Asmens duomenis naudojančios ir tvarkančios įstaigos privalo turėti duomenų subjektų sutikimą arba būti įstatymo įpareigotos tvarkyti asmens duomenis. Jos turi užtikrinti, kad asmens duomenys būtų:

- renkami ir tvarkomi apibrėžtais ir teisėtais tikslais,
- tikslūs duomenys, jei reikia, nuolat atnaujinami,
- netikslūs ar neišsamūs duomenys turi būti ištaisyti, papildyti, sunaikinti arba sustabdytas jų tvarkymas;
- tapatūs, tinkami ir tik tokios apimties, kuri būtina jiems rinkti ir toliau tvarkyti;
- saugomi tokia forma, kad duomenų subjektų tapatybę būtų galima nustatyti ne ilgiau, negu to reikia tiems tikslams, dėl kurių asmens duomenys buvo surinkti ir tvarkomi.

XI.3 ATSARGINIS KOPIJAVIMAS IR DUOMENŲ ATKŪRIMAS

Skaitmeninėse sistemose didesnė rizika įvykių, dėl kurių duomenys, o ir visa duomenų bazė gali būti nepataisomai sugadinti. Galimybė atkurti duomenis, kurie dėl kokios nors priežasties buvo prarasti arba pažeisti, dažniausiai siejama su atsarginių duomenų bazės kopijų darymu ir duomenų atkūrimu iš atsarginių kopijų. Jei tokios kopijos egzistuoja, Sistema leidžia vartotojui atkurti duomenis tokius, kokie jie buvo kuriuo nors konkrečiu laiku momentu.

Atsarginės duomenų kopijos saugomos diskuose arba juostose, arba prieš atiduodant į ilgalaikę juostinę saugyklą kitoje vietoje jos laikinai įrašomos į diskus. Atsargines kopijas reikia laikyti atskirai nuo pagrindinės sistemos vietos, kad būtų kuo mažesnė jų pažeidimo kartu su pagrindine sistema tikimybė. Atsarginio duomenų kopijavimo bei atkūrimo sistemoms dažniausiai reikalinga speciali programinė įranga, kurios paskirtis – susieti dažniausiai naudojamas programas ir duomenų bazes.

Rengiant atsarginio kopijavimo bei atkūrimo planą ir pasirenkant svarbu nustatyti duomenų atkūrimo taško ir atkūrimo laiko tikslus. **Atkūrimo taško** tikslas yra laiko (iki incidento, dėl kurio prarasti duomenys) momentas, iki kurio buvę duomenys privalo būti atkurti. Organizacijos strategijoje turi būti numatyta, koks didžiausias dar priimtinas laikotarpis, kurio metu atlikti duomenų pakeitimai po atkūrimo gali būti prarasti. Tada atkuriant duomenis, blogiausiu atveju bus išsaugoti bent jau ankstesni už tą laikotarpį iki gedimo ar avarijos duomenų pakeitimai. Šis laikotarpis gali būti diena, savaitė, retai keičiamiems duomenimas – mėnesiai, o kritiškai svarbiose sistemose – labai mažas, praktiškai nulinis. **Atkūrimo laiko** tikslas nurodo, koks ilgiausias dar priimtinas laikotarpis, per kurį turi būti atkurti prarasti duomenys, t.y., kiek laiko duomenys gali būti nepasiekiami be rimtų pasekmių.

Paplitę keli atsarginių kopijų kūrimo būdai.

- Atsarginis kopijavimas vietiniu tinklu – tai dažniausiai pasitaikantis atsarginio kopijavimo būdas, palaikomas visų rinkoje esančių komercinių operacinių sistemų. Jį nesudėtinga įdiegti ir valdyti. Kiekviename serveryje įdiegiamas atsarginio kopijavimo programinės priemonės, o duomenys į atsarginio kopijavimo serverį siunčiami vietiniu tinklu. Rekomenduojama naudoti atskirą atsarginio kopijavimo tinklą. Atsarginio kopijavimo ir atkūrimo našumą daugeliu atveju riboja tinklo pralaidumas.
- Atsarginis kopijavimas nenaudojant vietinio tinklo. Šiuo atveju serveriuose vietoje atsarginio kopijavimo priemonių įdiegiamos serverio, valdančio duomenų saugojimo įrenginius, programos. Tai leidžia serveriams persiųsti duomenis tiesiogiai į atsarginio kopijavimo įrenginį (diską ar juostą) nenaudojant vietinio tinklo. Šio būdo pranašumas yra didesnis

atsarginio kopijavimo ir atkūrimo našumas. Jis naudojamas serveriuose, kuriuose saugomi dideli duomenų kiekiai ir (arba) kritiškose veiklai sistemose, kurioms keliami greito duomenų atkūrimo reikalavimai.

- Momentinių kopijų metodas. Taikant momentinių kopijų atsarginio kopijavimo sprendimą galima apsieiti be įprastos atsarginio kopijavimo programinės įrangos. Vienas iš didžiausių šio būdo pranašumų - galimybė atsargines kopijas daryti daug dažniau, pavyzdžiui, kas valandą, o ne kas naktį. Siekiant išvengti duomenų praradimų avarijos atveju, rekomenduotina replikuoti duomenų momentines kopijas į antrinius (rezervinius) duomenų centrus. Naudojant šį metodą reikalinga duomenų saugykla, galinti sukurti ir saugoti daugybę duomenų momentinių kopijų nemažinant sistemos našumo.

Duomenų ir duomenų bazių valdymo sistemos **atkūrimas** paremtas procesų, taisyklių ir procedūrų visuma, kuri turi būti numatyta iš anksto tam, kad atsitikus avarijai, dėl kurios sutrinka organizacijos duomenų bazių valdymo sistemos veikla, būtų galima atkurti technologinę infrastruktūrą ir toliau tęsti darbą. Svarbu žinoti, kurie duomenys organizacijai yra vertingiausi – jie visada turi būti prieinami saugiai atskirtoje nuo galimo saugos incidento srities vietoje.

🕮 Klausimai diskusijai

Kaip manote, ar įmanoma išspręsti prieštarą tarp vis tikslesnės geografinės informacijos poreikio ir asmens bei kitų duomenų apsaugos reikalavimų. Kokios grėsmės visuomenei susijusios su masiniu GPS sekimo įrangos, antžeminės ir oro lazerinio skenavimo įrangos naudojimu? Ar visada jų galima išvengti apribojus šių technologijų naudojimą?

✍ Užduotis savarankiškam darbui

Susipažinkite su Lietuvos Respublikos Asmens duomenų teisinės apsaugos įstatymu, Bendraisiais elektroninės informacijos saugos valstybės institucijų ir įstaigų informacinėse sistemose reikalavimais, kitais pasirinktais su duomenų sauga susijusiais teisės aktais. Pagalvokite, ar esate susidūrę su šių teisės aktų pažeidimais.

💻 Užduotys praktikos darbams

Susipažinkite su duomenų bazės lentelių ir užklausų pagrindu kuriamomis formomis, formų objektais, jų savybėmis bei galimais veiksmais. Sukurkite keletą skirtingos paskirties formų pasirinktoje anksčiau sukurtoje duomenų bazėje. Įsitikinkite, kad per formą matomi ir redaguojami lentelėse saugomi duomenys. Apribokite rašymo teises pasirinktiems formos laukams. Pabandykite sukurti savo formą slaptažodžiui įvesti ir *Visual Basic .NET* aprašykite algoritmą jam patikrinti. Formų pagalba sukurkite grafinę pasirinktos duomenų bazės naudotojo sąsają.

XII. PAPILDOMI SKYRIAI

XII.1 DUOMENŲ KLASIFIKAVIMAS

Duomenų bazė yra ne kas kita, kaip realaus pasaulio dalies modelis, sudaromas pagal tam tikras taisykles. Pirmasis jos projektuotojo uždavinys yra išskirti konkrečiame kontekste svarbias esybes, dominančias jų savybes, ir nustatyti ryšius tarp jų. Taip gaunamas *koncepcinis modelis*, kuriame visa svarbi informacija yra atrinkta, apibendrinta ir sutvarkyta bei aprašyta žmogui suprantamomis *sąvokomis*. Koncepcinio modelio pavyzdys gali būti aprašyti geografiniai objektai, kurie bus saugomi duomenų bazėje (pavyzdžiui, upė ir ežeras), su jų neerdviniais atributais (pavyzdžiui, upės vidutinis metinis nuotėkis ir ežero tipas bei vandens druskingumas) ir tarpusavio ryšiais (pavyzdžiui, ežero priklausymas upės baseinui arba upės įtekėjimas į ežerą). Analogiškas modelis turi būti sudarytas grafiniams objektams (sutartiniams ženklams), kuriais vaizduojami išskirti geografiniai objektai ir jų savybės. Koncepcinio informacijos modelio sudarymo metodai išsamiai aptariami ketvirtajame šios knygos skyriuje.

Duomenų klasifikavimas (skirstymas) yra vienas žinomiausių informacijos sutvarkymo būdų. Nuo jo labai priklauso informacijos vertė.

Jei apibrėžimas atskleidžia sąvokos turinį, tai skirstymas nurodo visas rūšis, įeinančias į sąvokos apimtį.

Skirstydami giminę į rūšis, kreipiame dėmesį į požymius, kuriuos turi vienos rūšys ir neturi kitos. Tų požymių rūšis vadinama skirstymo pagrindu. Pavyzdžiui, trikampio kampų dydis yra pagrindas skirstyti juos į stačiakampius, bukakampius ir lygiašonius; kraštinių tarpusavio santykis – pagrindas skirstyti į lygiakraščius, lygiašonius ir įvairiakraščius. Sudėtingiau, kai porūšiai skirstomi dar kartą. Vienas iš galimų skirstymo metodų yra **dichotomija**:

Žmogus	Rusas			
	Ne rusas	Vokietis		
		Ne vokietis	Prancūzas	
			Ne prancūzas	
				Ir t.t.

Tai metodas skirstyti ne iki galo pažintai aibei. Jis išsamus kiekviename etape.

Skirstymo taisyklės.

- **Skirstymas turi būti suderintas**, t.y., rūšių suma turi būti lygi visumai. Tik tokia klasifikacija bus išsami ir teisinga.
- **Skirstymo nariai turi vienas kitą šalinti**, t. y., skirstoma į nesikertančias rūšis[20]. Pavyzdžiui, knygos pagal savybes negali būti skirstomos į naudingas, suprantamas, įdomias.
- **Skirstymo pagrindas turi būti tas pats.** Pavyzdžiui, gyventojai negali būti skirstomi į krikščionis, musulmonus, indus – arba pagal religiją, arba pagal tautybę.
- **Skirstymas turi būti nuoseklus,** t.y., pereinama į artimiausią žemesnę giminę nedarant „šuolių“. Pavyzdžiui, gamta – gyvūnai, augalai, uolienos keičiama į gamta – organinis pasaulis ir neorganinis pasaulis, kurie skirstomi toliau.

[20] Apskritai reikalaujama, kad skirstymas būtų vienareikšmis, t.y., objektas vienu metu priklausytų vienai ir tik vienai klasei. Tačiau ne visada įmanoma taip suskirstyti sudėtingus objektus. Tada įvedamos priklausomybės klasei su tam tikra tikimybe ar laipsniu sąvokos (angl.: *fuzzy dependences*).

Klasifikacija – tai objektų skirstymas pagal jų panašumą (požymius) į atskiras apibrėžtas aibes - klases.

Požymiai turi būti praktiškai reikšmingi, be to gera klasifikacija leidžia formuluoti maksimumą teiginių apie skirstomą aibę. Klasifikacija remiasi indukcija, kurios dėka ir nustatomi skirstymo požymiai. Jie gali būti ir kiekybiniai, ir kokybiniai. Svarbu, kad būtų gimininis (jungiantis) ir rūšinis (skiriantis) požymiai.
Klasifikacijos tikslas – sisteminti informaciją, padėti orientuotis objektų ir sąvokų sistemose.

Natūralioji ir dirbtinė klasifikacija.
Jei objektus į klases jungiame pagal esminius požymius, kurie susiję su dauguma objekto savybių, išreiškia jo prigimtį – tai natūralioji klasifikacija (pvz., augalų sistematika, periodinė cheminių elementų sistema). Dirbtinė klasifikacija sudaroma pagal neesminius objektų požymius, siekiant juos lengviau rasti tarp kitų objektų (pvz., abėcėlinis katalogas).

Enciklopedinė klasifikacija siekia apimti visą pažinimą. Specialioji klasifikacija apima vieną siaurą sritį, kurioje turi būti išsami. Klasifikacijos principus, metodus ir taisykles tiria taksonomija. Tikslas – sukurti klasifikacijas, kurios būtų informatyvios, neprieštaringos, adekvačios natūralioms sistemoms.

Klasifikacija gali būti vykdoma *induktyviai* (atskiri objektai jungiami į poklasius ir t.t.) arba *deduktyviai* (bendriausios klasės skaidomos į poklasius ir t.t.)

XII.2 SĄVOKOS IR JŲ SANTYKIAI

Sąvokomis išreiškiame viską, ką suvokiame. Sąvokos mąstyme fiksuojamos, įgauna apibrėžtumo terminų dėka, t.y., operuojame tik tomis sąvokomis, kurios gali būti išreikštos kalba. Susipažinsime su sąvokų klasifikacijomis, kurios naudojamos koncepciniame modeliavime.

Individualiosios ir bendrosios sąvokos
Individualiosios sąvokos apibrėžia vienetinius konkrečius objektus, pavyzdžiui, „aukščiausias Amerikos kalnas“, „Japonijos ambasadorius Lietuvoje“. Joms priskiriami ir tikriniai vardai.
Sąvokos, taikomos grupei (klasei) susijusių objektų arba reiškinių, vadinamos bendrosiomis, pavyzdžiui, „augalas“, „ambasadorius“, „grožis“.
Koncepciniame modelyje bendrosios sąvokos tampa objektais (esybėmis). Individualios sąvokos atitinka objekto (esybės) realizacijas, egzempliorius (angl. *instance*), t.y., konkrečius objektus.

Bendrosios ir kuopinės sąvokos
Sąvokos gali būti vartojamos kuopine prasme, pavyzdžiui, „miškas išskiria deguonį“ – „miškas“ vartojama bendrąja prasme, kaip vienas iš daugybės vientisų objektų. Tačiau „miškas“ gali būti suprantamas kaip medžių visuma – tada jis tampa kuopine sąvoka. Kuopinė sąvoka žymi visumą, susidedančią iš vientisų vienetų, pavyzdžiui, „žvaigždynas“, „minia“. Tačiau, jei ta visuma suvokiama kaip tam tikros klasės atstovas, kuopinė sąvoka virsta bendrąja, pavyzdžiui, „LN biblioteka“ (kuopinis) – „Lietuvos *bibliotekos*“ (bendrasis). Taigi, kuopinės sąvokos yra savita individualiųjų sąvokų forma.

Negalima painioti kuopinės ir bendrosios sąvokų. Teiginys, teisingas kuopinei sąvokai, visai nebūtinai tinka jos apimamiems objektams ir atvirkščiai. Pavyzdžiui, „parlamentas leidžia įstatymus“, tačiau ne kiekvienas parlamento narys tą daro.

Teiginys teisingas bendrąjai sąvokai, būtinai bus teisingas ir jos apimamiems objektams. Pavyzdžiui, „miškas išskiria deguonį“ ir kiekvienas konkretus miškas išskiria deguonį.

Koncepciniame modelyje kuopinis terminas nurodomas įdėtumo ryšiu tarp esybių „miškas ***sudarytas iš*** medžių“. Bendrosios sąvokos modeliuojamos paveldimumo ryšiu – kiekvienas konkretus „miškas“, pavyzdžiui, šilas, beržynas, giria ***yra*** „MIŠKAS“. Apibendrinanti sąvoka dar vadinama objekto klase. Klasės gali priklausyti dar bendresnėms klasėms – superklasėms. Taip pereinant nuo konkrečių prie vis bendresnių sąvokų sudaroma klasių hierarchija.

Abstrakčios ir konkrečios sąvokos

Abstrakčios (lot. *abstrahere* – abstrahuoti) sąvokos vartojamos objektų savybėms, būsenoms, veiksmams žymėti, kalbant apie juos atsietai nuo daiktų, t.y., šios savybės, būsenos ar veiksmai neegzistuoja apibrėžtoje erdvėje ar laike, pavyzdžiui, „svoris“, „malonumas“, „tiesa“. Dėl jų neapibrėžtumo ir daugiareikšmiškumo tokių sąvokų reikia vengti sudarant koncepcinį modelį. Be to, kartais abstrakčiomis dar laikomos sąvokos tokių daiktų, kurių negalima įsivaizduoti kaip apibrėžtų objektų, pavyzdžiui, „visata“, „žmonija“.

Konkrečios yra objektų, faktų, būsenų ir kt. sąvokos, jei jie laikomi tam tikru būdu egzistuojančiais, pavyzdžiui, „kvadratas“, „liepsna“.

Koncepciniame modelyje esybės visada yra tik konkrečios sąvokos, tuo tarpu jų savybės, kurios dar vadinamos atributais, gali būti nusakomos abstrakčiais terminais, pavyzdžiui, žmogaus ūgis arba svoris.

Teigiamosios ir neigiamosios sąvokos

Teigiamosios sąvokos naudojamos vienai ar kitai esamai kokybei žymėti, pavyzdžiui, „gražus“, „baigtinis“. Neigiamosios sąvokos žymi kokybės nebuvimą, pavyzdžiui, „negražus“, „begalinis“. Modeliuojant jų reikėtų vengti, nes pačios savaime tokios sąvokos neturi turinio.

Reliatyvios ir absoliučios sąvokos

Absoliuti sąvoka – tai tokia sąvoka, kuri žymį nepriklausomą objektą, neturintį santykio su jokiu kitu, pavyzdžiui, „namas“. Santykinė sąvoka be žymimo objekto suponuoja dar ir kito objekto buvimą, pavyzdžiui, „tėvas“, „partneris“.

Koncepciniame modelyje santykinė sąvoka dažnai reiškia taip vadinamą silpnąją esybę, kuri visada yra ryšiu sujungta su kita esybe. Dar dažniau silpnosios esybės pakeičiamos ryšiais.

Kiekviena sąvoka gali turėti požymių aibę, kuriais ji skiriasi nuo kitų sąvokų. Ne visi požymiai yra vienodai reikšmingi. Nuo Aristotelio laikų sąvokų požymiai skirstomi į penkias klases.

1. Gimininis požymis.

Jei sakysime, kad geografija yra mokslas, tai mokslas yra gimininis sąvokos „geografija“ požymis. Tuo ji skiriasi nuo viso to, kas nėra mokslas.

Apibrėžimas. Giminė – tai sąvoka klasės, į kurią įtraukiama kita nagrinėjama sąvoka.

Koncepciniame modelyje gimininis požymis visada yra paveldimas iš superklasės.

2. Rūšinis skirtumas (specifika)

Jei sakysime, kad geografija yra mokslas, *tiriantis teritorinį objektų pasiskirstymą*, tai pažymėsime, kuo šis mokslas skiriasi nuo kitų mokslų.

Apibrėžimas. Rūšinis skirtumas – tai žymė, padedanti atskirti sąvoką nuo kitų tos pačios giminės sąvokų.

Rūšinis skirtumas – tai požymis, kuriuo skiriasi skirtingi tos pačios klasės objektai, t.y., atributas, kuris *nėra* paveldimas iš superklasės.

3. Rūšis

Jei prie gimininio požymio prijungsime rūšinį skirtumą, gausime rūšį. Pavyzdžiui, „pastatas *ginklams saugoti*“ – arsenalas; „pastatas *grūdams laikyti*“ – svirnas. Taip sudaromi apibrėžimai.

4. Savybinis požymis (savybė)

Apibrėžimas. Savybinis požymis – tai žymė, priklausanti visiems nagrinėjamos klasės objektams, kuri nėra jiems esminė, bet išvedama iš esminių požymių.

Pavyzdžiui, trikampio esminiai požymiai yra „tiesialinijinė dvimatė uždara figūra su trimis kampais“. Požymis, kad trikampio kampų suma lygi 180°, yra savybinis požymis. Beje, jei trikampis egzistuos ne Euklido erdvėje, šis savybinis požymis gali būti kitoks, pavyzdžiui, sferinio trikampio kampų suma visada bus didesnė už 180° ir priklausys nuo sferos spindulio. Tuo tarpu trys kampai yra neatsiejamas, esminis požymis.

5. Nesavybinis požymis

Apibrėžimas. Nesavybinis požymis – tai žymė, galinti priklausyti visiems nagrinėjamos klasės objektams, kuri negali būti išvedama iš esminių požymių.

Pavyzdžiui, varnos juoda spalva (nežinome kodėl taip yra). Nesavybiniai požymiai skirstomi į *neatskiriamus* (pavyzdžiui, „aš *gimiau Lietuvoje*“) ir *atskiriamuosius* (t.y., tokius, kurie vienu metu gali būti, o kitu – nebūti, pavyzdžiui, „Bušas *yra Amerikos prezidentas*“). Koncepciniame modelyje leistini tik neatskiriamieji nesavybiniai požymiai. Savybinių požymių, jei jie yra saugomi, reikšmės paprastai apskaičiuojamos, kaip, pavyzdžiui, linijinių objektų ilgiai.

Sąvokos turinys ir apimtis

Apibrėžimai. Sąvokos turinys – tai jos požymių visuma. Sąvokos apimtis – suma tų klasių, grupių, giminių, rūšių ir kt., kurioms ta sąvoka gali būti taikoma.

Sąvokos turinys gali kisti priklausomai nuo požiūrio, žinių ir pan., pavyzdžiui, „cukrus“ konditerio, chemiko, mediko požiūriu – suvokiamos skirtingos savybės. Sąvokos „keturkampis“ apimtis – kvadratas, stačiakampis, rombas, trapecija ir kt.

Koncepciniame modelyje sąvokos turinys – tai ją atitinkančios esybės atributų aibė. Sąvokos apimtis – tai jos žymimos klasės poklasių aibė.

Apibrėžimas. Didesnės apimties sąvoka vadinama į jos apimtį įeinančios sąvokos gimine; įeinanti sąvoka šiuo atveju vadinama rūšimi.

Bet kuri rūšis gali virsti gimine ir atvirkščiai, pavyzdžiui, medis–palmė–kokoso palmė. Taip skaidant galų gale prieinama sąvoka, kurios apimtyje nebegali būti rūšių, tik atskiri individai.

Susiaurinimas ir apibendrinimas. Norėdami sudaryti siauresnę pagal apimtį sąvoką iš bendresnių, turime pridėti papildomų žymių prie bendrosios sąvokos, pavyzdžiui, iš sąvokos „elementas“ sąvoka „metalas“ gaunama papildomai nurodžius valentinių elektronų skaičių. Atvirkštinis procesas (požymių atėmimas) vadinamas apibendrinimu. Apibendrinant pereinama nuo klasės prie superklasės, paliekant tik bendriausius požymius.

Didėjant sąvokos turiniui, mažėja jos apimtis ir atvirkščiai. Pavyzdžiui, „žmogus“ ir „negras“ – pirmosios sąvokos platesnė apimtis (visi žmonės, tam tarpe ir negrai), o antrosios – turinys (visos žmogui būdingos savybės + negrams specifinės: tamsi oda ir pan.).

Susiaurinimas ir apibendrinimas yra loginis bet kokio klasifikavimo pagrindas.

Sąvokų subordinacija
Viena sąvoka įeina į kitą kaip jos apimties dalis, pavyzdžiui, „beržas“ < „medis“. Koncepciniame modelyje subordinuotas sąvokas atitinkančios esybės sudaro klasių hierarchiją, susietą paveldimumo ryšiais.

Sąvokų koordinacija
Į bendresnės sąvokos apimtį įeina dvi ar daugiau vienodai jai subordinuotų siauresnių sąvokų. Siauresnės sąvokos vadinamos koordinuotomis, pavyzdžiui, „ežeras“ ir „tvenkinys“ koordinuoti „vandens telkinio“ sąvokos kontekste. Koncepciniame modelyje koordinuotas sąvokas atitinkančios esybės turi bendrą superklasę.

Lygiareikšmės sąvokos
Tai skirtingo turinio, bet vienodos apimties sąvokos, pavyzdžiui, „gyvas“ yra lygiareikšmė sąvoka sąvokai „mirtingas“. Koncepciniame modelyje gali būti tik viena esybė, žyminti visas lygiareikšmes sąvokas.

Priešingos sąvokos
Jei sąvokos apimtyje rūšis sutvarkysime pagal kokio nors požymio intensyvumą, gausime seką, kurios pirmas ir paskutinis narys bus priešingos sąvokos. Pavyzdžiui, *Grayscale* spalvas suskirstę pagal intensyvumą nuo 0 iki maksimalaus, gausime priešingas juodą ir baltą spalvas. Ne visos sąvokos turi sau priešingas, pavyzdžiui, „raudona“. Žemėlapių reljefo aukščių skalėse priešingos spalvos yra žalia ir ruda.

Prieštaraujančios sąvokos
Sąvoka A ir kita sąvoka B, apie kurią žinoma, kad ji *nėra* A. Prieštaraujančios sąvokos terminas gaunamas pridėjus neigimo dalelytę, pavyzdžiui, baltas-nebaltas. Prieštaraujanti sąvoka neturi savo apibrėžto turinio. Koncepciniame modelyje prieštaraujančios sąvokos neleistinos.

Susikertančios sąvokos
Tai sąvokos, kurių turinys skirtingas, bet apimtis iš dalies sutampa, pavyzdžiui, „geografai“ ir „dėstytojai“. Sankirtoje esančios apimčių dalys yra lygiareikšmės (*geografijos* dėstytojas ir geografas *dėstytojas*). Koncepciniame modelyje susikertančios sąvokos nepageidaujamos.

XII.3 PREDIKATŲ LOGIKOS PAGRINDAI

Teiginiai ir predikatai

Teiginys – tai sakinys, apie kurį galima pasakyti, teisingas jis, ar ne.

Teiginys gali būti užrašytas raidėmis ar kitais simboliais, pvz., p, r, q. Teiginio teisingumo reikšmė yra funkcija T(p).

Jei teiginys p išreiškia tiesą, T(p)=1, kitaip T(p)=0. Matematinė logika tiria, kaip nustatyti sudėtingų teiginių teisingumo reikšmes.

Yra sakinių, kurie nėra teiginiai, pavyzdžiui, klausiamasis sakinys. Išnagrinėkime dar keletą pavyzdžių, kurie skamba labai panašiai į teiginius.

x>0
"*Ežeras yra į šiaurę nuo Vilniaus*."
"*Upė įteka į ežerą*."

Neįmanoma nustatyti šių teiginių teisingumo, nežinant kintamųjų x, „ežeras", „upė" reikšmių – priklausomai nuo jų atitinkami teiginiai gali tapti ir klaidingais, ir teisingais.

Predikatas – tai sakinys su kintamaisiais, kuris tampa teiginiu vietoje kintamųjų įrašius konkrečias jų reikšmes.

Predikatus žymėsime kaip jų kintamųjų funkcijas didžiosiomis raidėmis, pvz., P(a, b). Predikatų skaičiavimas taikomas daugelyje sričių – įrodymuose, programavimo kalbose, taip pat ir duomenų bazių valdymo sistemose.

Loginės operacijos

Teiginiai yra skirstomi į paprastuosius ir sudėtinius. Paprastuoju laikomas teiginys, kurio negalima suskaidyti į kitus teiginius. Sudėtiniai teiginiai sudaromi iš kitų paprastų ar sudėtinių teiginių sujungiant juos *loginėmis jungtimis*. Yra penkios loginės jungtys:

a) netiesa, kad;
b) arba;
c) ir;
d) tada ir tik tada, kai;
e) jeigu ..., tai.

Sudėtinių teiginių sudarymas naudojantis šiomis jungtimis vadinamas *loginėmis operacijomis*. Atitinkamai, yra penkios loginės operacijos.

1. **Neigimas**. Jei p – teiginys, tai teiginys „netiesa, kad p" (dažnai sakoma „ne p") vadinamas teiginio p neiginiu ir žymimas ¬p; Jo teisingumo reikšmė T(¬p) = 1-T(p). Programavimo kalbose neigimas paprastai žymimas angliškų žodžiu NOT.
2. **Disjunkcija (arba loginė sudėtis)**. Jei p ir q – teiginiai, tai teiginys „p arba q" vadinamas teiginių p ir q disjunkcija arba logine suma ir žymimas p ∨ q; Teiginių p ir q disjunkcija neteisinga vieninteliu atveju – kai p ir q abu yra neteisingi. Programavimo kalbose disjunkcija dažnai žymima angliškų žodžiu OR.
3. **Konjunkcija (arba loginė daugyba)**. Jei p ir q – teiginiai, tai teiginys „p arba q" vadinamas teiginių p ir q disjunkcija arba logine sandauga ir žymimas p & q. Teiginių p ir q disjunkcija teisinga vieninteliu atveju – kai p ir q abu kartu yra teisingi. Programavimo kalbose konjunkciją reiškia žodis AND.
4. **Implikacija.** Jei p ir q – teiginiai, tai teiginys „jeigu p, tai q" vadinamas teiginių p ir q implikacija ir žymimas p ⇒ q. Implikacija p ⇒ q yra neteisingas teiginys tada ir tik tada, kai p teisingas, o q – neteisingas. Tai įdomi savybė, praktiškai reiškianti, kad iš neteisingos prielaidos bet kokia išvada yra formaliai korektiška. Implikacija dar gali būti skaitoma: „iš p išplaukia q"; „p yra teiginio q pakankamoji sąlyga"; „q yra teiginio p būtinoji sąlyga".
5. **Ekvivalentumas**. Jei p ir q – teiginiai, tai teiginys „p tada ir tik tada, kai q" vadinamas teiginių p ir q ekvivalentumu (tapatumu). Ekvivalentumas žymimas p ⇔ q arba p ≡ q. Jis yra teisingas, kai p ir q teisingumo reikšmės sutampa, o klaidingas – kai jos skiriasi. Dar skaitoma „p būtina ir pakankama, kad būtų q" arba „q būtina ir pakankama, kad būtų p".

Pavyzdžiai

p – „*metai turi 12 mėnesių*“, q – „*Nemunas teka per Vilnių*“, r – „*sniegas yra baltas*“.
¬p, ¬q, ¬r - ?
T (p ∨ q) = 1; T(p ∨ q ∨ r) =1; T(p & r) = 1; T(q & r) = 0; T(p & q & r) = 0.
T(p ⇒ q) = 0. T(q ⇒ p) = 1. T (p ≡ ¬q) = 1;

Galima sudaryti ir elementarių loginių operacijų teisingumo reikšmių lenteles.

p	**q**	**p ∨ q**	**p & q**	**p ⇒ q**	**p ≡ q**
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	0
1	1	1	1	1	1

Sudėtingesni teiginiai sudaromi taikant keletą loginių operacijų, pavyzdžiui, (p ⇒ q) & (q ∨ r).

Logikos algebra

Apibrėžiant logines operacijas jungiamiems teiginiams nekeliami jokie reikalavimai, o domina tik jų teisingumo reikšmės. Taigi, teiginius galima žymėti tiesiog raidėmis. Pačios raidės, taip pat elementarios operacijos vadinamos pagrindinėmis loginėmis formomis. Jei juose vietoje raidžių įrašysime kokius nors kitus teiginius, paprastus ar sudėtinius, vėl gausime teiginius.

Loginė forma – tai reiškinys, gautas baigtinį skaičių kartų panaudojus loginių operacijų ženklus ir skliaustus raidėms sujungti.

Pavyzdžiui, loginės formos yra (p ⇒ q) ∨ (¬p & r), ((p ⇒ q) ∨ ¬p & r)) ⇒ r.

Dvi loginės formos, kurių teisingumo reikšmių lentelės sutampa, vadinamos *logiškai ekvivalenčiomis*.

Loginė forma, kurios teisingumo reikšmė yra 1 nepriklausomai nuo ją sudarančių teiginių teisingumo reikšmių, vadinama *tautologija* ir žymima **I**.
Loginė forma, kurios teisingumo reikšmė yra 0 nepriklausomai nuo ją sudarančių teiginių teisingumo reikšmių, vadinama *tapačiai klaidinga* (loginiu nuliu) ir žymima **O**. Akivaizdu, kad
¬I ≡ O, **¬O ≡ I**.

Matome, kad loginės formos panašios į algebros reiškinius (sudėtis, daugyba, nulis, vienetas, lygybė). Aptarsime paprasčiausias loginių operacijų savybes.

Dvigubo neigimo dėsnis.

¬¬p ≡ p

Tai pagrindinė neigimo savybė. Dėsnio teisingumu įsitikiname iš loginių operacijų teisingumo reikšmių lentelės.

Disjunkcijos savybės

1) p ∨ p ≡ p (disjunkcijos idempotencijos dėsnis);
2) p ∨ ¬p ≡ **I** (negalimo trečiojo dėsnis);

3) $p \vee \mathbf{I} \equiv \mathbf{I}$
4) $p \vee q \equiv q \vee p$ (komutatyvumo dėsnis);
5) $(p \vee q) \vee r \equiv p \vee q \vee r)$ (asociatyvumo dėsnis)
6) $p \vee \mathbf{O} \equiv p$.

Konjunkcijos savybės

1) $p \& p \equiv p$ (konjunkcijos idempotencijos dėsnis);
2) $p \& \neg p \equiv \mathbf{O}$ (prieštaravimo dėsnis);
3) $p \& \mathbf{I} \equiv p$
4) $p \& q \equiv q \& p$ (komutatyvumo dėsnis);
5) $(p \& q) \& r \equiv p \& q \& r)$ (asociatyvumo dėsnis)
6) $p \& \mathbf{O} \equiv \mathbf{O}$

Kai kurios implikacijos savybės

$(p \Rightarrow q) \equiv \neg p \vee q$
$(p \equiv q) \equiv ((p \Rightarrow q) \& (q \Rightarrow p))$

Logikos dėsniai

Kiekviena tautologija vadinama logikos dėsniu. Kai kurie dėsniai buvo minėti nagrinėjant loginių operacijų savybes. Visus dėsnius galima įrodyti remiantis jų teisingumo reikšmių lentelėmis.

Kontrapozicijos dėsnis

$(p \Rightarrow q) \equiv (\neg q \Rightarrow \neg p)$

Silogizmo dėsnis.

$[((p \Rightarrow p_1) \square \square \& (p_1 \Rightarrow q)) \Rightarrow (p \Rightarrow q)] \equiv \mathbf{I}$

De Morgano dėsniai.

$\neg (p \vee q) \equiv \neg p \& \neg q$
$\neg (p \& q) \equiv \neg p \vee \neg q$

Prieštaros (suvedimo į prieštaravimą) dėsnis

$(p \& \neg q) \Rightarrow (r \& \neg r) \equiv (p \Rightarrow q)$

Teisingos išvados taisyklė (modus ponens)

Jei teiginiai p ir $p \Rightarrow q$ yra teisingi, tai ir q teisingas.

Neteisingos išvados taisyklė (modus tollens)

Jei teiginys $p \Rightarrow q$ yra teisingas, o q – neteisingas, tai ir p – neteisingas.

XII.4 AIBIŲ ALGEBROS PAGRINDAI

Aibė yra viena iš pagrindinių matematikos sąvokų. Kaip sąvoka visumos elementų, turinčių juos jungiantį požymį, ji atsiranda praktiškai kiekvienoje teorijoje, nors žodis „aibė“ ir nevartojamas. Todėl svarbu mokėti iš turimų aibių konstruoti naujas aibes, t.y., mokėti aibių veiksmus ir pagrindines jų savybes.

Jei aibė apibrėžta ne išvardijant elementus, o kokia nors taisykle, gali būti taip, kad ji neturės nė vieno elemento. Tokią (apibrėžtą) aibę vadinsime *tuščiaja aibe* ir žymėsime Ø . Pavyzdžiui, tokia yra aibė Lietuvos susituokusių gyventojų, priklausančių amžiaus grupei nuo 0 iki 10 metų.

Jei kiekvienas aibės A elementas yra ir aibės B elementas, sakoma, kad A yra B *poaibis*, o B – aibės A *viršaibis*. Tai žymima $A \subset B$ arba $B \subset A$. Pavyzdžiui, $\{a, b, c\} \subset \{a, b, c, d\}$.
Remiantis poaibio apibrėžimu, kiekviena aibė yra savo pačios poaibis. Tuščiąją aibę galima laikyti bet kurios kitos aibės poaibiu. $A \subset A$; $Ø \subset A$. Šie du aibės A poaibiai vadinami *netiesioginiais*, o kiti (jei jų yra) – *tiesioginiais*.

Laikysime, kad visi vienos aibės elementai yra skirtingi. Jei dvi aibės A ir B turi tuos pačius elementus (tuo atveju jos yra viena kitos poaibiai), jas vadinsime lygiomis ir rašysime A = B.

Nagrinėdami konkretų tam tikros teorijos klausimą, niekada nesusiduriame su visomis galimomis aibėmis, o tik su tomis, kurios tiesiogiai siejasi su sprendžiamu uždaviniu. Todėl patogu apibrėžti *universaliąją aibę* **I**, kuri būtų visų toje teorijoje nagrinėjamų aibių viršaibis. Tada bet kuri tos visumos aibė A yra **I** poaibis. Pavyzdžiui, įvairios amžiaus grupės, dirbančiųjų tam tikrose ūkio šakose, moterų, alkoholikų, imigrantų aibės yra **Gyventojų** universaliosios aibės poaibiai.

Aibių sudėtis.

Aibių A ir B *suma* arba junginiu vadinama aibė, sudaryta iš visų elementų, priklausančių bent vienai iš duotųjų aibių. A ir B suma žymima $A \cup B$ arba A + B.

Bendrieji sudedamų aibių elementai įeina į sumą tik vieną kartą (pagal susitarimą, kad aibės elementai yra skirtingi). Pavyzdžiui, $\{1,2,3\} \cup \{1,3,4,7\} = \{1,2,3,4,7\}$; „mergaitės“ ∪ „berniukai“=“vaikai“. Aibių sudėtis atitinka teiginių disjunkciją:

$A + B = \{a \mid a \in A \vee a \in B\}$ arba

$a \in A \cup B \equiv a \in A \vee a \in B$

Toks formalus apibrėžimas patogus, kai aibės apibrėžtos tik nurodant jų požymius. Pavyzdžiui, kai $A = \{x \mid P(x)\}$ ir $B = \{x \mid Q(x)\}$, $A \cup B = \{x \mid P(x) \vee Q(x)\}$.

Aibių sudėties savybės.

1. Jeigu $A \subset B$, tai $A \cup B = B$.
2. $A \cup B = B \cup A$ (komutatyvumas)
3. $(A \cup B) \cup C = A \cup (B \cup C)$ (asociatyvumas)
4. $A \cup A = A$
5. $A \cup Ø = A$
6. $A \cup \mathbf{I} = \mathbf{I}$

Pirmosios trys savybės akivaizdžios; likusios trys yra pirmosios savybės išvados. Daugumą savybių galima išplėsti bet kokiam skaičiui aibių.

Aibių daugyba.

Aibių A ir B *sankirta* arba sandauga vadinama aibė, sudaryta iš visų elementų, priklausančių abiems iš duotųjų aibių. A ir B sankirta žymima $A \cap B$ arba AB.

Pavyzdžiui, {1,2,3} ∩ {1,3,5,7} = {1,3}; „moterys“ ∩ „vaikai“=“mergaitės“; „moterys“ ∩ „berniukai“= Ø. Aibių daugyba atitinka teiginių konjunkciją:

$A \cap B = \{a| a \in A \& a \in B\}$ arba

$a \in A \cap B \equiv a \in A \& a \in B$

Pavyzdžiui, kai $A = \{x| P(x)\}$ ir $B = \{x| Q(x)\}$, $A \cap B = \{x| P(x) \& Q(x)\}$.

Aibių daugybos savybės.

1. Jeigu $A \subset B$, tai $A \cap B = A$.
2. $A \cap B = B \cap A$ (komutatyvumas)
3. $(A \cap B) \cap C = A \cap (B \cap C)$ (asociatyvumas)
4. $(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$ (daugybos distributyvumas sudėties atžvilgiu)
5. $A \cap A = A$
6. $A \cap \emptyset = \emptyset$
7. $A \cap \mathbf{I} = A$

Kaip ir sudėties atveju, daugumą savybių galima praplėsti bet kokiam skaičiui aibių.

Aibių atimtis.

Aibių A ir B *skirtumu* vadinama aibė tų aibės A elementų, kurie nepriklauso aibei B. A ir B skirtumas žymimas A\B.

Pavyzdžiui,{1,2,3,4,7}\{1,2,3}={4,7};“vaikai“\“vyrai“=“mergaitės“; „moterys“\“berniukai“=“moterys“.

$A \setminus B = \{a| (a \in A) \& (a \notin B)\}$ arba

$a \in A\B \equiv (a \in A) \& (a \notin B)$

Pavyzdžiui, kai $A = \{x| P(x)\}$ ir $B = \{x| Q(x)\}$, $A\B = \{x| P(x) \& \neg Q(x)\}$.

Skirtumas **I**\A vadinamas aibės A papildiniu ir žymimas A* .

$A^* = \{a| a \notin A\}$

Aibių skirtumo savybės.

1. $A\B = A \cap B^*$
2. Jeigu $A \subset B$, tai $A\B = \emptyset$
3. $A\setminus \emptyset = A$
4. Jeigu $A \cap B = \emptyset$, tai $A\B = A$
5. $(A\B) \cap C = (A \cap C) \setminus (B \cap C)$
6. $(A\B) \cup B = A \cup B$

Aibių papildinio savybės.

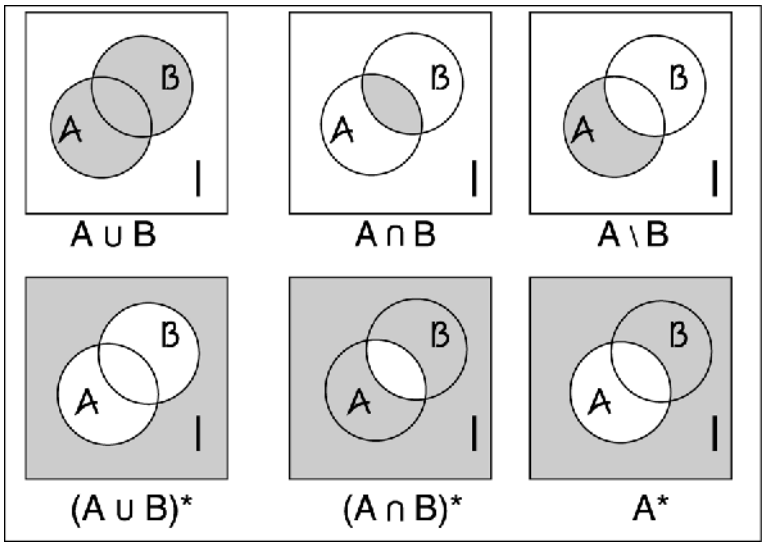
1. $A \cup A^* = \mathbf{I}$
2. $A \cup A^* = \emptyset$
3. $\mathbf{I}^* = \emptyset$
4. $\emptyset^* = \mathbf{I}$
5. $A^{**} = A$

Oilerio ir Veno diagramos

Dažnai patogu aibes vaizduoti grafiškai. Paveiksle parodytos figūros, reiškiančios aibių operacijų rezultatus.

Remiantis diagramomis, lengvai galima patikrinti dvi lygybes, vadinamas *de Morgano dėsniais* (jie visiškai analogiški logikos de Morgano dėsniams):

1. $(A \cap B)^* = A^* \cup B^*$
2. $(A \cup B)^* = A^* \cap B^*$



XII-1 pav. Oilerio ir Veno diagramomis pavaizduoti aibių operacijų rezultatai

Kvantoriai.

Iš predikato P(x) teiginį gausime tik įrašę vietoje x kokią nors konkrečią reikšmę. Pavyzdžiui, „Vidutinės *gyventojų grupės* pajamos yra 1000 Lt/mėn“, jei **I** = **Gyventojai**, bus teisingas, jei *gyventojai* = „gyventojai su aukštuoju išsilavinimu“. Teiginius iš predikatų galima sudaryti ir naudojant *kvantorius*.

Sakinys „(aibėje A) egzistuoja toks elementas, kuriam P(x) – teisingas teiginys“ sutrumpintai rašomas: $(\exists x \in A)$: P(x); $\exists x$: P(x) – jei A universali. Pavyzdžiui, „egzistuoja gyventojas, kurio pajamos lygios 1000 Lt/mėn.“ Ženklas $\exists$ vadinamas egzistavimo kvantoriumi. (angl. *exists*).
Sakinys „Visiems x (iš aibės A)“ teiginys P(x) yra teisingas sutrumpintai užrašomas $(\forall x \in A)$: P(x); $\forall x$: P(x) – jei aibė A yra universali. Ženklas $\forall$ vadinamas visuotinumo kvantoriumi. (angl. *all*). Pavyzdžiui, „Kiekvieno gyventojo (x) su aukštuoju išsilavinimu (A) pajamos ne mažesnės 1000 Lt/mėn.“ SQL kalboje kvantoriai užrašomi žodžiais: EXISTS ir FORALL.

Teiginiai, sudaryti su kvantoriais, jungiami pagal tas pačias taisykles, kaip ir paprasti teiginiai. Pavyzdžiui, $\neg(\forall x \in A)$: P(x); $(\forall x \in A)$: $\neg$P(x); $\neg(\exists x \in A)$: P(x); $(\exists x \in A)$: $\neg$P(x).
“Egzistuoja gyventojas (x) su aukštuoju išsilavinimu (A), kurio pajamos yra mažesnės kaip 1000 Lt/mėn.“ ≡ „Netiesa, kad kiekvieno gyventojo (x) su aukštuoju išsilavinimu (A) pajamos ne mažesnės 1000 Lt/mėn.“

Labai svarbu tiksliai suformuluoti teiginius ir atskirti, kurie iš jų yra logiškai ekvivalentūs, o kurie – ne.

Aibių Dekarto daugyba.

Vienodiems objektams atskirti paprastai vartojame numerius, pavyzdžiui, telefono abonentų. Tačiau kartais vieno numerio nepakanka objektui žymėti ir tenka jam priskirti du skaičius ar kokių kitokių ženklų porą. Porų sudarymas atitinka aibių Dekarto daugybos operaciją.

Aibių A ir B *Dekarto sandauga* (kombinatorine sandauga) vadinama aibė porų (a,b), kuriuose a yra bet kuris aibės A elementas, o b – bet kuris B elementas. Ji žymima $A \otimes B$

Pavyzdžiui, {1,2} x {3,4} = {(1,3), (2,3), (1,4), (2,4)}. $A \otimes A$ (Dekarto kvadratas).
Dekarto daugyba nėra nei komutatyvi, nei asociatyvi, bet pasižymi distributyvumu aibių sąjungos ir sankirtos atžvilgiu.
Dekarto daugybą galima praplėsti bet kokiam skaičiui aibių: A_1 x A_2 x... x A_n. Nesunku pastebėti, kad jei visos aibės A_i yra baigtinės ir kiekviena turi po m_i elementų, tai Dekarto sandaugoje A_1 x A_2 x... x A_n yra $m_1m_2m_3...m_n$ elementų. Jei bent viena aibė begalinė – sandauga taip pat begalinė, pavyzdžiui, jei A = {1} o B = **R**, tai geometriškai sandaugą galima pavaizduoti kaip tiesę, lygiagrečią y ašiai einančią per tašką (1,0).

LITERATŪRA

Lietuvių kalba

Baronas R. (2005). Duomenų bazių sistemos. Vilnius, TEV.

Paradauskas B.,Nemuraitė L. (2008). Duomenų bazių semantiniai modeliai. Kauno technologijos universitetas.

Paršeliūnas E. (1997). Geoinformacinės sistemos: duomenų bazės. Mokomoji knyga. V.: Technika.

Sekliuckis V., Gudas S., Garšva G. (2008). Informacijos sistemos ir duomenų bazės: informacijos sistemų ir reliacinių duomenų bazių kūrimo pagrindai. Kaunas: Technologija.

Anglų kalba

Booch G., Rumbaugh J. and Jacobson I. (2005). The Unified Modeling Language User Guide. Second ed. Addison-Wesley

Castelli V. and Bergman D. (Eds) (2002) Image Databases: Search and Retrieval of Digital Imagery. Wiley-Interscience, John Wiley & Sons Inc., New York.

Cattell R. G. (1994). Object Data Management. Object-Oriented and Extended Relational Database Systems. Addison-Wesley

Chen P. (1976): The entity-relationship model: toward a unified view of data. ACM Transactions and database Systems 1(1), p. 9–35.

Churcher C. (2012). Beginning Database Design: From Novice to Professional. Apress.

Clarke K. (1990) Analytical and Computer Cartography. Englewood Cliffs, New Jersey, USA: Prentice Hall.

Codd, E.F. (1990). The Relational Model for Database Management (Version 2 ed.). Addison

Darwen H., Date C.J. (1998). Foundation for Object/Relational Databases : The Third Manifesto. Addison-Wesley Pub Co.

Date C.J. (2003). An Introduction to Database Systems. Eigth edition. Addison-Wesley.

Date, C.J. (2011). SQL and Relational Theory: How to Write Accurate SQL Code. O'Reilly Media; Second Edition.

Halpin T. (2001) Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design. Morgan Kaufmann Publishers, San Francisco, CA.

Laurini R., Thompson D. (1992). Fundamentals of Spatial Information systems. Academic Press.

Moellering H. (2005).World spatial metadata standard. International Cartographic Association.

Rahimi S. K. and Haug F.S. (2010). Distributed Database Management Systems: A Practical Approach. John Wiley & Sons Inc., New York.

Teorey T.J., Lightstone S. S, Nadeau T. and Jagadish H. V. (2011). Database Modeling and Design, Fifth Edition: Logical Design. The Morgan Kaufmann Series in Data Management Systems.

Whitehorn M.and Marklyn B. (2006). Inside Relational Databases with Examples in Access. Springer.

Yan L. and Ma Z. (2011). Intelligent Multimedia Databases and Information Retrieval: Advancing Applications and Technologies. IGI Global.

Rusų kalba

Дейт К. (2001). Введение в системы баз данных (седьмое издание). Вильямс.

Грофф Д., Вайнберг П. (2003). Энциклопедия SQL. Питер

Interneto ištekliai

„Access“ pagrindai. URL: http://office.microsoft.com/lt-lt/access-help/CH010355021.aspx (2012-06-01)

2007 m. kovo 14 d. Europos Parlamento ir Tarybos direktyva 2007/2/EB, sukurianti Europos bendrijos erdvinės informacijos infrastruktūrą (INSPIRE). URL: http://inspire.jrc.ec.europa.eu/ (2011-12-20)

AulaClic. Access2003 Tutorial. URL: http://www.teacherclick.com/access2003/index.htm (2011-09-01)

Etheridge D. Microsoft Access 2007 Tutorial—Free & Online. URL: http://www.baycongroup.com/access2007/index.html (2011-09-01)

Geografinių duomenų bazių valdymo sistemos. Mokomoji knyga. http://www.geoportal.lt/wps/poc?uri=page:RUBRIC.1291 (2012-06-01)

Geografinių duomenų standartai ir infrastruktūra OpenGIS konsorciumas. URL: http://www.opengis.org (2012-06-01)

Geografinių duomenų standartai. URL: http://www.fgdc.gov/publications/documents/standards/geospatial_standards_part1.html (2011-09-01)

JAV Federalinis geografinių duomenų komitetas. URL: http://www.fgdc.gov/ (2011-09-01)

Oracle Spatial objektinė RDBVS. URL: http://otn.oracle.com/products/spatial/content.html (2012-06-01)

Pasaulinis erdvinių duomenų infrastruktūros sekretoriatas. URL: http://www.gsdi.org/ (2012-06-01)

SQL Course.com. Interactive Online SQL Training. URL: http://sqlcourse.com/intro.html (2011-09-01)

Tarptautiniai erdvinių duomenų standartai. URL: http://www.statkart.no/isotc211 (2012-06-01)

ILIUSTRACIJŲ SĄRAŠAS

LENTELIŲ SĄRAŠAS