3. Duomenų bazių projektavimas

3.1. Įvadas

Šiame skyriuje aptarsime reliacinių duomenų bazių projektavimą. Duomenų bazės projektavimo rezultatas yra loginė DB struktūra. Sprendžiant šį uždavinį ieškoma ne bet kokios DB struktūros, o tokios, kuri turėtų kiek galima mažiau blogų ir kiek galima daugiau gerų savybių. Duomenų bazės kokybei įvertinti taikomi formalūs metodai. Kita vertus, formaliai aptikus nepageidautinas DB struktūros savybes, gana dažnai, galima jas formaliai ir pašalinti. Šiame skyriuje aptarsime, kaip įvertinti DB struktūros kokybę ir kaip projektuoti DB, kad ji neturėtų žinomų nepageidautinų savybių.

Kai kurios reliacinio modelio sąvokos jau buvo pateiktos ankstesniuose skyriuose. Šiame skyriuje reliacinį modelį aptarsime detaliau ir formaliau.

3.2. Pagrindinės reliacinio modelio sąvokos

Reliaciniame modelyje duomenys pateikiami lentelėmis. Matematikoje lentelę atitinka santykis. **Santykis** (angl. *relation*) šiuo atveju išreiškiamas dvimate lentele, susidedančia iš eilučių ir stulpelių. Lentelės stulpelis reliaciniame modelyje dažnai vadinamas **atributu**. Stulpelis turi pavadinimą - tai atributo vardas. Reliacinėje teorijoje laikomasi nuostatos, kad tiek stulpelių, tiek ir eilučių tvarka lentelėje yra neapibrėžta. Visų leistinų atributo reikšmių aibė vadinama **domenu**. **Tuščioji reikšmė** (žymima `NULL`) – tai reikšmė, kuri priskiriama atributui eilutėje, kai atributo reikšmė nežinoma arba atributas yra neprasmingas. Tarkime, duomenų bazėje *Darbai* kai kurie asmenys (projektų vykdytojai) yra be aukštojo išsilavinimo. Tada atributas "baigta aukštoji mokykla" tokiems asmenims netenka prasmės.

Atributų rinkinys (aibė), vienareikšmiškai apibrėžiantis (nusakantis) kiekvieną lentelės eilutę, vadinamas **viršrakčiu**. Lentelės **raktu** vadinamas viršraktis, iš kurio pašalinus bent vieną atributą jis nustoja būti viršrakčiu. Taigi raktas – tai minimalus viršraktis. Raktą taip pat galima apibrėžti kaip minimalią atributų aibę, vienareikšmiškai apibrėžiančią (kitaip dar sakoma funkciškai apibrėžiančią) kiekvieno atributo reikšmę eilutėje. Kitaip tariant, **lentelės** *L* **raktu** vadinamas toks jos atributų aibės poaibis *K*, kad:

1) rakto atributų reikšmės vienareikšmiškai apibrėžia visų lentelės *L* atributų reikšmes, t.y. dvi skirtingos lentelės eilutės negali sutapti pagal visų aibės *K* atributų reikšmes (vienareikšmiška identifikacija),
2) joks aibės *K* poaibis neturi vienareikšmiškos identifikacijos savybės (pertekliaus nebuvimas).

Duomenų bazės *Darbai* lentelėje *Vykdytojai* atributų rinkinys {*Nr*, *Pavardė*} yra lentelės viršraktis, kadangi pagal bet kokias šių atributų reikšmes lentelėje galima surasti ne daugiau kaip vieną eilutę. Tačiau, ši aibė nėra raktas, nes jos poaibis {*Nr*} irgi yra viršraktis. Kadangi pastarąją aibę sudaro tik vienas atributas, tai ji yra lentelės raktas. Užrašant aibę, riestinius skliaustelius praleisime, kai aibę sudarys tik vienas elementas.

Lentelėje gali būti keli atributų rinkiniai, turintys rakto savybę. Visi jie vadinami **galimais raktais**, arba tiesiog raktais. Pavyzdžiui, stulpelis *Pavardė* yra galimas lentelės raktas. Tačiau, taip bus tik tuomet, kai pavardės nesikartos, t.y. nebus dviejų asmenų vienodomis pavardėmis. Paprastai tokios prielaidos daryti negalima, todėl *Pavardė* neturėtų būti raktu. Raktas, kurį sudaro du ar daugiau atributų vadinamas **sudėtiniu raktu**.

Paprastai vienas iš raktų paskelbiamas **pirminiu raktu**. Jei lentelėje yra keli raktai, tai pirminiu parenkamas tas, kuriuo paprasčiausia naudotis, pavyzdžiui, trumpiausias. Dažniausiai, raktu vadinsime pirminį raktą. Pavyzdžiui, lentelė *Vykdymas* turi vieną sudėtinį raktą {*Vykdytojas*, *Projektas*}, kuris yra ir vienintelis galimas lentelės pirminis raktas.

Lentelės reliacinė schema (struktūra, aprašas) - tai lentelės ir visų jos stulpelių pavadinimai, su pažymėtu (išskirtu, pvz., pabrauktu) pirminiu raktu. Pavyzdžiui, DB *Darbai* lentelių reliacines schemas galima pavaizduoti taip:

Vykdytojai (*Nr, Pavardė, Kvalifikacija, Kategorija, Išsilavinimas*);
Projektai (*Nr, Pavadinimas, Svarba, Pradžia, Trukmė*);
Vykdymas (*Projektas, Vykdytojas, Statusas, Valandos*).

Atkreipsime dėmesį į tai, kad lentelės *Projektai* raktą sudaro du stulpeliai, todėl ir pabraukti. Tai jokiu būdu nereiškia, kad kiekvienas iš šių stulpelių, atskirai paimtas, irgi yra raktas. Lentelė turi tik vieną raktą, sudarytą iš dviejų stulpelių.

Dažnai dviejose DB lentelėse yra vienas ar net keli stulpeliai, turintys tą pačią prasmę abiejose lentelėse. Pavyzdžiui, lentelės *Vykdytojas* stulpelio *Nr* ir lentelės *Vykdymas* stulpelio *Vykdytojas* galimų reikšmių aibės pagal stulpelių prasmę sutampa. Abiejų šių stulpelių reikšmės yra vykdytojų tapatumo numeriai. Gana dažnai atitinkami stulpeliai turi panašius, ar net vienodus, pavadinimus. Tokia atributų pora yra išorinio rakto pavyzdys. **Išoriniu raktu** vadinamas vienos lentelės atributų rinkinys, kuris kitoje lentelėje (ar net toje pačioje) yra pirminis raktas. Išorinio rakto atributų pavadinimai nebūtinai turi sutapti su pirminio rakto atributų pavadinimais. Kadangi išorinis raktas paprastai siejamas su lentelės pirminiu raktu, tai jam apibrėžti visiškai pakanka nurodyti išorinį raktą sudarančius atributus ir pavadinimą lentelės, į kurios pirminį raktą išorinis raktas rodo.

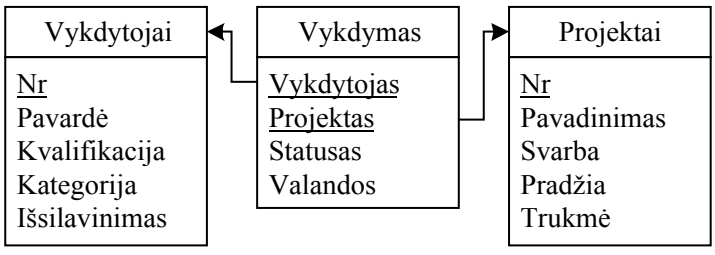
Išorinis raktas vartojamas loginiams ryšiams tarp lentelių apibrėžti. Informacija apie išorinius raktus yra svarbi DB loginei struktūrai. Todėl **duomenų bazės reliacine schema** vadinsime visų jos lentelių reliacinių schemų rinkinį kartu su lentelių išoriniais raktais. Pavyzdžiui, DB *Darbai* reliacinę schemą galima pavaizduoti taip:

Vykdytojai (*Nr, Pavardė, Kvalifikacija, Kategorija, Išsilavinimas*);
Projektai (*Nr, Pavadinimas, Svarba, Pradžia, Trukmė*);
Vykdymas (*Projektas, Vykdytojas, Statusas,Valandos*),
išoriniai raktai: *Projektas* nurodo į *Projektai*,
Vykdytojas nurodo į *Vykdytojai*.

Pastebėsime, kad duomenų bazėje *Darbai* tik viena lentelė *Vykdymas* turi du išorinius raktus, kitos dvi lentelės išorinių raktų neturi.

DB reliacinę schemą galima pavaizduoti ir grafiškai. Grafiškai pavaizduota DB reliacinė schema tampa aiškesnė, ypač kai duomenų bazę sudaro daug išoriniais raktais tarpusavyje susijusių lentelių. Kiekvienos lentelės schema vaizduojama stačiakampiu, kurio viršuje užrašomas lentelės pavadinimas. Žemiau išvardijami visi lentelės atributai (stulpelių pavadinimai). Pirminį raktą sudarantys atributai pabraukiami. Pagrindinis grafinio vaizdavimo privalumas - galimybė grafiškai pavaizduoti ryšius tarp lentelių. Lentelės išoriniai raktai vaizduojami rodykle iš išorinį raktą sudarančių atributų į lentelės, į kurią išorinis raktas nurodo, pavadinimą.

DB *Darbai* reliacinę schemą galima pavaizduoti taip:



3.1 pav. Duomenų bazės *Darbai* reliacinė schema.

3.3. Duomenų vientisumo sąlygos

Reliacinėje teorijoje yra keletas reikalavimų, kuriuos turi atitikti duomenų bazės duomenys. Viena reikalavimų rūšis vadinama **duomenų vientisumo sąlygomis**. Aptarsime tris iš jų:

- kategorijų vientisumas;
- nuorodų vientisumas;

- funkciniai sąryšiai.

Reliacinės lentelės eilutės yra konkrečių realaus pasaulio objektų atvaizdai duomenų bazėje. Pavyzdžiui, kiekviena lentelės *Vykdytojai* eilutė atitinka konkretų firmos darbuotoją. Realaus pasaulio objektai, kurie vaizduojami lentelės eilutėmis, reliacinėje teorijoje vadinami **kategorijomis**. Lentelės raktas vienareikšmiškai nusako lentelės eilutę, tuo pačiu ir kategoriją. Norėdamas rasti duomenis apie konkrečią kategoriją (objektą), vartotojas privalo žinoti rakto reikšmes. Tai reškia, kad kategorija neturi prasmės duomenų bazėje, jei bent vieno rakto atributo reikšmė yra nežinoma. Todėl pagal **kategorijų vientisumo reikalavimą** joks lentelės rakto atributas nė vienoje eilutėje negali turėti `NULL` reikšmės.

Jau minėjome, kad reliacinėje duomenų bazėje vienos lentelės eilučių ryšiui su kitos lentelės eilutėmis vartojami išoriniai raktai. Pavyzdžiui, duomenų bazės *Darbai* lentelės *Vykdymas* kiekvienoje eilutėje yra duomenys apie tai, kiek konkretus vykdytojas konkrečiam projektui vykdyti skiria valandų. Stulpelis *Vykdytojas* šioje lentelėje vartojamas konkrečiam vykdytojui nurodyti. Pagal jo reikšmę lentelėje *Vykdytojai* galima rasti (naudojantis šios lentelės raktu) visus duomenis apie tą vykdytoją. Todėl labai svarbu, kad kiekvienos eilutės atributo *Vykdytojas* reikšmė lentelėje *Vykdymas* atitiktų kurią nors atributo *Nr* reikšmę lentelėje *Vykdytojai*. Kitaip lentelėje *Vykdymas* turėsime nuorodą į nežinomą (neapibrėžtą) vykdytoją. Duomenų bazė, kurioje visi netušti išoriniai raktai nurodo į egzituojančią pirminio rakto reikšmę, tenkina nuorodų vientisumo reikalavimą. Pastebėsime, kad lentelėje išorinis raktas gali būti tuščias (jo atributų reikšmės yra `NULL`), jei tik jo atributai neįeina į pirminį raktą. **Nuorodų vientisumo reikalavimas** gali būti suformuluotas taip: kiekvieno išorinio rakto reikšmė duomenų bazėje turi būti arba tuščia, arba sutapti su viena pirminio rakto reikšme lentelėje, į kurią išorinis raktas nurodo.

Trečiąjį reikalavimą duomenims - funkcinius sąryšius - detaliau aptarsime kituose skyreliuose.

3.4. Duomenų anomalijos

Tarkime, vietoje dviejų DB *Darbai* lentelių *Vykdymas* ir *Projektai* turime vieną lentelę *Projektai_Vykdymas*. Tuomet DB reliacinė schema yra tokia:

Vykdytojai (*Nr, Pavardė, Kvalifikacija, Kategorija, Išsilavinimas*),
Projektai_Vykdymas (*Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojas, Statusas, Valandos*),
išorinis raktas: *Vykdytojas* nurodo į *Vykdytojai*.

Užpildžius lentelę *Projektai_Vykdymas* duomenimis, kurie buvo dviejose lentelėse *Projektai* ir *Vykdymas*, naujoji lentelė bus tokia:

Projektai_Vykdymas

Projektas	*Pavadinimas*	*Svarba*	*Trukmė*	*Vykdytojas*	…
1	Studentų apskaita	Maža	12	1	
1	Studentų apskaita	Maža	12	2	
1	Studentų apskaita	Maža	12	3	
1	Studentų apskaita	Maža	12	4	
2	Buhalterinė apskaita	Vidutinė	10	1	
2	Buhalterinė apskaita	Vidutinė	10	2	
2	Buhalterinė apskaita	Vidutinė	10	4	
3	WWW svetainė	Didelė	6	1	
3	WWW svetainė	Didelė	6	2	
3	WWW svetainė	Didelė	6	3	

Lentelę supaprastinome pavaizduodami ne visus jos duomenis. Dešiniojo stulpelio, kuris pažymėtas daugtaškiu, vietoje turėtų būti pavaizduotos atributų *Pradžia, Statusas* ir *Valandos* reikšmės.

Nesunku pastebėti, kad lentelė *Projektai_Vykdymas* sudaryta nesėkmingai. Pavyzdžiui, keturiose eilutėse, atitinkančiose projektą Nr. 1, kartojasi tas pats projekto pavadinimas, jo

svarba ir trukmė. Tokie **pertekliniai duomenys** ne tik užima papildomą vietą kompiuterio atmintyje – dėl jų gali būti prarastas duomenų vientisumas, arba, kitaip tariant, atsirasti duomenų prieštaringumas. Mat tada vieną projektą gali vykdyti keli vykdytojai. Tarkime, projekto Nr. 1 trukmė pasikeitė, ir naujoji trukmė buvo pakeista tik vienoje eilutėje. Tuomet tarp eilučių, kuriose yra informacija apie tą patį projektą Nr. 1, atsiranda neatitikimas, kuris vadinamas atnaujinimo anomalija. **Atnaujinimo anomalija** – tai duomenų prieštaringumas, atsirandantis dėl duomenų pertekliaus, atnaujinus tik dalį jų.

Tarkime, vykdytojai, kurių numeriai yra 1, 2 ir 3 išeina iš darbo. Tada iš lentelės pašalinamos visos eilutės, atitinkančios šiuos vykdytojus. Todėl iš lentelės *Projektai_Vykdymas*, o kartu ir iš duomenų bazės, pašalinami visi duomenys apie projektą Nr. 3, nes jį vykdė tik tie iš darbo išėję darbuotojai. Tai nėra gerai, nes projektą vis tiek reikės tęsti, paskyrus dirbti kitus darbuotojus. Tai **pašalinimo anomalija** – nenumatytas duomenų praradimas, susijęs su kitų duomenų pašalinimu.

Kita vertus, firma gali gauti užsakymą naujam projektui vykdyti, kai dar nėra paskirtas nė vienas projekto vykdytojas. Kadangi vykdytojo numeris, tiksliau, atributas *Vykdytojas*, yra lentelės *Projektai_Vykdymas* rakto dalis, tai, kol nebus paskirtas bent vienas vykdytojas, mes negalėsime įvesti į duomenų bazę duomenų apie naująjį projektą. Tai **įvedimo anomalija** – nebuvimas galimybės įvesti duomenis dėl to, kad trūksta kitų duomenų.

Duomenų bazė turi būti sudaryta taip, kad atnaujinimo, pašalinimo ir įvedimo anomalijų nebūtų. Todėl duomenų bazėje *Darbai* vietoj lentelės *Projektai_Vykdymas* turi būti dvi lentelės *Projektai* ir *Vykdymas,* kaip tai buvo padaryta anksčiau. Tačiau tai tik intuityvus problemos sprendimas, kuris, esant duomenų bazėje daug lentelių ir atributų, gali nebūti toks akivaizdus. Tolimesniuose skyreliuose mes apibrėšime formalesnį metodą, vadinamą lentelių skaidymu, kuris duos tokį patį rezultatą. **Lentelių skaidymu** vadinamas lentelės padalijimas į kelias lenteles, siekiant išvengti anomalijų ir neprarasti duomenų vientisumo. Tam naudojamos normalinės formos ir lentelių struktūrizavimo taisyklės.

3.5. Pirmoji normalinė forma

Normaline forma (sutrumpintai, **NF**) vadinami apribojimai duomenų bazės reliacinei schemai, kuriuos taikant DB išvengia tam tikrų nepageidaujamų savybių.

Lentelė yra **pirmos normalinės formos** (1NF), jei visų jos atributų reikšmės yra atomai. **Atomu** laikoma reikšmė, kuri nėra nei aibė, nei sąrašas. Paprastai apibėžiant reliacinę lentelę reikalaujama, kad visos visų atributų reikšmės būtų atomai. Todėl toliau visas nagrinėjamas lenteles laikysime attinkančiomis 1NF. 1NF sąvoką paaiškinsime, nagrinėdami pavyzdį lentelės, kuri neatitinka 1NF.

Imkime ankstesniame skyrelyje įvestą lentelę *Projektai_Vykdymas.* Nustatėme, kad šiai lentelei būdingos duomenų anomalijos. Visos įvardytos duomenų anomalijos gali būti pašalintos, išskaidžius lentelę į dvi lenteles. Pabandykime problemą spręsti kitaip. Anomalijų priežastis yra ta, kad vieną projektą gali vykdyti keli vykdytojai, ir dėl to lentelėje *Projektai_Vykdymas* atsiranda duomenų perteklius. Šią lentelę pertvarkykime į lentelę *Projektai_Vykdytojai*:

Projektai_Vykdytojai(*Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojai, Statusas, Valandos*).

Pastarojoje lentelėje atributo *Vykdytojai* reikšmės yra visų projekto vykdytojų numerių aibė. Naująją lentelę galima pavaizduoti taip:

Projektai_Vykdytojai

Projektas	*Pavadinimas*	*Svarba*	*Trukmė*	*Vykdytojai*	…
1	Studentų apskaita	Maža	12	{1,2,3,4}	
2	Buhalterinė apskaita	Vidutinė	10	{1,2,4}	
3	WWW svetainė	Didelė	6	{1,2,3}	

Iš pirmo žvilgsnio gali pasirodyti, kad ši lentelė sudaryta gerai, nes joje nėra duomenų pertekliaus. Tikriausiai tai vienintelis jos privalumas. Visų pirma, šiai lentelei negalima

apibrėžti išorinio rakto, nes lentelės atributas *Vykdytojai* negali būti susietas su lentelės *Vykdytojai* pirminiu raktu, kaip buvo anksčiau, nors ryšys tarp jų yra. Mat vykdytojų numerių aibė nėra palyginama su vienu numeriu. Nesant išorinio rakto, stulpelyje *Vykdytojai* gali atsirasti numeriai, neatitinkantys nė vieno vykdytojo, užregistruoto lentelėje *Vykdytojai*.

Lentelė *Projektai_Vykdytojai* nėra pirmos normalinės formos, nes jos stulpelyje *Vykdytojai* yra aibės, kurių elementai kitoje lentelėje yra atributo reikšmės. Jei lentelę *Projektai_Vykdytojai* pakeistume lentele *Projektai_Vykdymas*, turėtume lentelę, atitinkančią 1NF. Tačiau jau anksčiau padarėme išvadą, kad lentelė *Projektai_Vykdymas* nėra gerai sudaryta, nors ji ir yra 1NF. Taigi reikalavimas lentelei atitikti 1NF yra per silpnas, kad lentelės schemą būtų galima vadinti pakankamai gera. Todėl nagrinėjamos ir aukštesnės eilės normalinės formos. Kuo aukštesnė NF, tuo aukštesni reikalavimai lentelės schemai. Taigi kuo aukštesnės NF reikalavimus tenkina lentelės schema, tuo mažiau blogų savybių ji turi. Duomenų bazė yra tam tikros NF, jei visos jos lentelės yra tos NF. Aukštesnėms NF apibrėžti vartojama funkcinio sąryšio sąvoka.

3.6. Funkciniai sąryšiai

Jau aptarėme kategorijų ir nuorodų vientisumo sąlygas. Dar griežtesnius reikalavimus reliacinei schemai apibrėžia funkciniai sąryšiai. Tai bene svarbiausias apribojimas. Jo esmė - vienų atributų reikšmės eilutėje gali vienareikšmiškai apibrėžti kitų atributų reikšmes. Pavyzdžiui, kiekvienoje lentelės *Projektai_Vykdymas* eilutėje *Projektas* vienareikšmiškai apibrėžia atributų *Pavadinimas*, *Svarba, Trukmė* ir *Pradžia* reikšmes.

Tokiu būdu, jei atributų aibės A reikšmės eilutėje vienareikšmiškai apibrėžia atributų aibės B reikšmes eilutėje, tai sąryšį tarp A ir B vadiname funkciniu. Formaliau, **funkcinis sąryšis** (**f-sąryšis**) gali būti apibrėžtas taip: jei A ir B yra lentelės L atributų aibės, tai užrašas $\boldsymbol{A \rightarrow B}$ reiškia, jog jei dvi lentelės L eilutės turi vienodas atributų A reikšmes, tai atributų B reikšmės taip pat sutampa. Užrašą $A \rightarrow B$ galima skaityti dvejopai: B **funkciškai priklauso** nuo A, taip pat A **funkciškai apibrėžia** B. F-sąryšio kairiosios dalies atributai vadinami **determinantu**. Determinantas – tai atributai, kurių reikšmės apibrėžia kitų atributų reikšmes.

Jau minėti lentelės *Projektai_Vykdymas* atributų tarpusavio sąryšiai yra f-sąryšių pavyzdžiai, kuriuos galima užrašyti taip:

$Projektas \rightarrow Pavadinimas,$
$Projektas \rightarrow Svarba,$
$Projektas \rightarrow Trukmė,$
$Projektas \rightarrow Pradžia.$

Šį keturių funkcinių sąryšių užrašą galima pakeisti vienu trumpesniu:

$Projektas \rightarrow \{Pavadinimas, Svarba, Trukmė, Pradžia\}.$

Be šių f-sąryšių, lentelėje *Projektai_Vykdymas* galioja dar ir toks:

$\{Projektas, Vykdytojas\} \rightarrow \{Statusas, Valandos\}.$

Funkcinio sąryšio sąvoka yra sąvokos "lentelės viršraktis" apibendrinimas. Lentelės L viršraktis – tai toks jos atributų aibės R poaibis S ($S \subseteq R$), kad $S \rightarrow R$. Faktą, kad lentelės L atributų aibė yra R, žymesime $L(R)$, kuris atitinka lentelės L schemos užrašą be lentelės pirminio rakto.

Atkreipsime dėmesį į tai, kad f-sąryšis, kaip ir kiti apribojimai duomenims, galioja reliacinei schemai, o ne apie konkretiems lentelės duomenims. Pagal einamąjį lentelės turinį neįmanoma nustatyti, ar koks nors f-sąryšis teisingas (būdingas) schemai. Pavyzdžiui, iš turimų duomenų lentelėje *Projektai_Vykdymas* būtų galima daryti išvadą, kad šios lentelės schemai būdingas f-sąryšis: $Svarba \rightarrow Trukmė$, kadangi jis tinka visoms lentelės eilutėms. Tačiau taip tvirtinti negalima, nes kitu metu gali būti vykdomi du (ar daugiau) projektai, kurių svarba vienoda, tačiau trukmė skirtinga. Taigi kai sakoma, kad lentelei būdingas konkretus f-sąryšis, tai turima omenyje, kad tas sąryšis galioja bet kuriuo momentu.

F-sąryšis $A_1, A_2,\ldots, A_n \to B$ vadinamas **trivialiu**, jei B sutampa su vienu iš aibės $\{A_1, A_2,\ldots, A_n\}$ atributų. Pastebėsime, kad bet kuris trivialus f-sąryšis yra teisingas bet kuriam santykiui. Bendru atveju f-sąryšis $A_1, A_2,\ldots, A_n \to B_1, B_2,\ldots, B_m$:

- **trivialus**, jei $\{B_1, B_2,\ldots, B_m\}$ yra aibės $\{A_1, A_2,\ldots, A_n\}$ poaibis;
- **netrivialus**, jei egzistuoja bent vienas $B_i \in \{B_1, B_2,\ldots, B_m\}$, toks, kad $B_i \notin \{A_1, A_2,\ldots, A_m\}$;
- **visiškai netrivialus**, jei $\forall i : 1,\ldots,m : B_i \notin \{A_1, A_2,\ldots, A_m\}$.

Labai svarbi *f*-sąryšių savybė yra ta, kad nesudėtinga sistema gali nuolat automatiškai sekti, ar apibrėžti (žinomi sistemai) sąryšiai yra tenkinami (nėra pažeisti). Pvz., kiekvieną kartą įterpiant naujus duomenis į lentelę bei keičiant esamus, galima patikrinti, ar įvykdžius operaciją nebus pažeistas kuris nors f-sąryšis. Pateiksime paprastą algoritmą, sprendžiantį šį uždavinį vienam atskirai paimtam sąryšiui.

Algoritmas patikrinti, ar lentelė L konkrečiu momentu tenkina *f*-sąryšį $A \to B$.

1) lentelės L eilutes sugrupuojame pagal atributų A reikšmes taip, kad eilutės su vienodomis A reikšmėmis priklausytų tai pačiai grupei.
2) jei kiekvienai eilučių grupei su vienodomis A reikšmėmis atributų B reikšmės taip pat sutampa, tai L tenkina f-sąryšį $A \to B$, kitaip - netenkina.

Visiškai nesunku pastarąjį algoritmą apibendrinti, kad vietoj vieno sąryšio būtų tikrinama, ar lentelė tenkina pasirinktą *f*-sąryšių aibę.

3.7. Funkcinių sąryšių uždarinys

Jau minėjome, kad vieni f-sąryšiai gali būti pakeisti kitais. Tarkime, turime lentelę su trimis atributais *A, B* ir *C*, tarp kurių galioja sąryšiai: $A \to B$ ir $B \to C$. Nesunku pastebėti, kad tuomet toje lentelėje galios ir sąryšis $A \to C$, vadinamas **tranzityviuoju** f-sąryšiu, t.y. C tranzityviai priklauso nuo A. Aibė visų galimų f-sąryšių, kurie apibrėžiami duotąja f-sąryšių aibe F, vadinama **aibės *F* uždariniu** ir žymima F^+. Armstrongas (W.W. Armstrong) pirmasis suformulavo išvedimo taisykles, kuriomis galima gauti visus aibės F^+ f-sąryšius. Suformuluosime šias taisykles, kurios dar vadinamos Armstrongo aksiomomis.

Armstrongo aksiomos. Tarkime *A, B* ir *C* yra reliacinės lentelės (santykio) *L(R)* atributų aibės R poaibiai, o užrašas AB reiškia aibių A ir B sąjungą.

1. Refleksyvumas: jei $B \subseteq A$, tai $A \to B$.
2. Papildymas: jei $A \to B$, tai $AC \to BC$.
3. Tranzityvumas: jei $A \to B$ ir $B \to C$, tai $A \to C$.

Kiekviena iš šių taisyklių gali būti įrodyta remiantis f-sąryšio apibrėžimu. Be to, ši aibė yra pilnoji, t.y. duotai f-sąryšių aibei F bet koks f-sąryšis, priklausantis F^+, gali būti išvestas naudojant tik šias taisykles. Tam, kad turimai aibei F būtų paprasčiau išvesti aibę F^+, šios trys taisykės yra papildomos dar keliomis.

4. Savęs apibrėžimas: $A \to A$.
5. Dekompozicija: jei $A \to BC$, tai $A \to B$ ir $A \to C$.
6. Apjungimas: jei $A \to B$ ir $A \to C$, tai $A \to BC$.
7. Kompozicija: jei $A \to B$ ir $C \to D$, tai $AC \to BD$, čia $D \subset R$.

Pastarosios keturios taisyklės gali būti išvestos iš pirmųjų trijų, visos jos įvestos tik patogumo dėlei.

Turint duotai reliacinei lentelei $L(R)$ galiojančių f-sąryšių aibę F, galima ieškoti lentelės raktų. Išspręsime šiek tiek paprastesnį uždavinį - sudarysime procedūrą, kuria patikrinsime, ar pasirinktas lentelės atributų aibės poaibis yra viršraktis. Priminsime, kad lentelės raktas – tai virsraktis, iš kurio negalima pašalinti jokio atributo, išsaugant viršrakčio savybę, o viršraktis – tai jos atributų aibės R poaibis S ($S \subseteq R$), kuriam $S \to R$. Galimybė patikrinti, ar atributų aibės poaibis S yra viršraktis, labai palengvintų raktų paieškos uždavinio sprendimą. Tam reikia rasti būdą sudaryti aibę visų atributų, kurie funkciškai priklauso nuo S, aibę, kuri

vadinama a**tributų aibės *S* uždariniu** f-sąryšių aibei *F*. Atributų aibės *S* uždarinys žymimas S^+.

Algoritmas atributų aibės *S* uždariniui S^+ f-sąryšių aibės *F* atžvilgiu rasti.
$S^+ := S$;
while (true)
  $T := S^+$;
  for each $X \to Y \in F$
    if $X \subseteq S^+$ then $S^+ := S^+ \cup Y$;
  endfor
  if $T \equiv S^+$ then
    leave loop ;
endwhile .

Pastebėsime, kad duotai f-sąryšių aibei *F* nesunku patikrinti, ar f-sąryšį $X \to Y$ galima išvesti iš *F*, nes taip gali būti tada ir tik tada, kai $Y \subseteq X^+$.

Dvi f-sąryšių aibės *F* ir *G* yra vadinamos **ekvivalenčiomis**, jei $F^+ \equiv G^+$.

Naudodamiesi, naujai įvestomis sąvokomis, dar kartą apibrėšime lentelės raktą. Lentelės *L*(*R*), kurioje galioja f-sąryšių aibė *F*, raktas yra toks jos atributų aibės *R* poaibis *K* ($K \subseteq R$), kad:

1) $K \to R$ (vienareikšmiška identifikacija),
2) $\forall B \subset K : B \to R \notin F^+$ (pertekliaus nebuvimas).

F-sąryšis yra semantinė sąvoka. F-sąryšiai susiję su duomenų prasme. Nurodysime svarbiausias priežastis, kodėl f-sąryšiai yra svarbūs reliacinėse DBVS:

- f-sąryšiais galima aprašyti gana svarbius reikalavimus duomenims (loginius ryšius tarp duomenų), kurių nepaisant duomenys gali tapti prieštaringais;
- f-sąryšiais galima išreikšti svarbias DB loginės struktūros savybes, kurios palengvina DB struktūros projektavimą ir įgalina gauti geresnį rezultatą;
- f-sąryšius gana paprasta išreiškšti, todėl atitinkamos priemonės įtrauktos į SQL kalbą, naudojant juos kaip pirminius ir išoriniais raktus bei indeksus;
- DBVS, prieš atlikdama kiekvieną duomenų modifikavimą, gali patikrinti, ar operacijos rezultate nebus pažeistas kuris nors f-sąryšis, tuo pačiu užkirsti galimybę duomenų bazei patekti į būseną, kai kuris nors f-sąryšis bus nepatenkintas.

3.8. Antroji normalinė forma

Atributų aibė *B* yra **visiškai priklausoma** nuo atributų aibės *A*, *f*-sąryšių aibės *F* atžvilgiu, jei *B* - funkciškai priklausoma nuo visos aibės *A*, bet nėra funkciškai priklausoma nei nuo jokio *A* poaibio. Formaliau, atributų aibė *B* visiškai priklausoma nuo atributų aibės *A*, jei $A \to B \in F^+$, tai $\forall C \subset A : C \to B \notin F^+$.

Lentelės *L*(*R*) atributas *a* ($a \in R$) vadinamas **pirminiu** lentelės *L* **atributu** *f*-sąryšių aibės *F* atžvilgiu, jei *a* priklauso kuriam nors lentelės *L* raktui, priešingu atveju *a* vadinamas **nepirminiu atributu**.

Lentelė *L*(*R*) yra **antros normalinės formos** (2NF) *f*-sąryšių aibės *F* atžvilgiu, jei ji yra 1NF ir kiekvienas jos nepirminis atributas visiškai priklauso nuo kiekvieno lentelės *L* rakto. **Duomenų bazė yra 2NF** *f*-sąryšių aibės *F* atžvilgiu tada ir tik tada, kai visos jos lentelės yra 2NF atžvilgiu *F*.

Paanalizuokime aukščiau pateiktą lentelę *Projektai_Vykdymas:*

Projektai_Vykdymas (*Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojas, Statusas, Valandos*).

Kaip jau nustatėme, šioje lentelėje galioja tokie f-sąryšiai:

$Projektas \to \{Pavadinimas, Svarba, Trukmė, Pradžia\}$, (3.1)
$\{Projektas, Vykdytojas\} \to \{Statusas, Valandos\}$. (3.2)

Ši lentelė turi vienintelį raktą {*Projektas, Vykdytojas*}. Pateiktieji du sąryšiai vienareikšmiškai nusako šį lentelės raktą, nes:

pagal refleksyvumo taisyklę –

$$\{Projektas, Vykdytojas\} \rightarrow Projektas, \quad (3.3)$$

iš (3.1) ir (3.3) pagal tranzityvumo taisyklę –

$$\{Projektas, Vykdytojas\} \rightarrow \{Pavadinimas, Svarba, Trukmė, Pradžia\}, \quad (3.4)$$

pagal refleksyvumo taisyklę –

$$\{Projektas, Vykdytojas\} \rightarrow Vykdytojas, \quad (3.5)$$

iš (3.2) - (3.5) pagal apjungimo taisyklę -

$$\{Projektas, Vykdytojas\} \rightarrow \{ Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojas, Statusas, Valandos\} . \quad (3.6)$$

F-sąryšis (3.6) reiškia, kad atributų aibė {*Projektas, Vykdytojas*} yra lentelės *Projektai_Vykdymas* viršraktis. Kitaip tariant, $\{Projektas, Vykdytojas\}^+ \equiv \{Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojas, Statusas, Valandos\}$. Kadangi atributų aibė {*Projektas, Pavadinimas, Svarba, Trukmė, Pradžia, Vykdytojas, Statusas, Valandos*} nėra nei atributo *Projektas*, nei atributo *Vykdytojas* uždarinys, tai {*Projektas, Vykdytojas*} yra ne tik viršraktis, bet ir raktas.

Vadinasi, lentelės *Projektai_Vykdymas* atributai *Projektas, Vykdytojas* yra pirminiai atributai, o visi kiti atributai: *Pavadinimas, Svarba, Trukmė, Pradžia, Statusas, Valandos* – nepirminiai. Atributai *Pavadinimas, Svarba, Trukmė, Pradžia* nevisiškai priklauso nuo lentelės rakto, jie funkciškai priklauso nuo atributo *Projektas*, kuris yra rakto dalis. Todėl lentelė *Projektai_Vykdymas* nėra 2NF. Jau anksčiau pastebėjome, kad šiai lentelei būdinga pertekliniai duomenys bei duomenų įvedimo, atnaujinimo ir šalinimo anomalijos. Šių nepageidautinų savybių nelieka išskaidžius lentelę į dvi lenteles *Projektai* ir *Vykdymas*. Abi lentelės *Projektai* ir *Vykdymas* yra 2NF. Tokiu būdu, 2NF sumažina duomenų perteklių ir pašalina duomenų anomalijas. Kita vertus, darbas su dviem lentelėmis yra šiek tiek sudėtingesnis, negu su viena. Prisiminkime, kad duomenų paieškai keliose lentelėse SQL turi galimybę jungti lenteles. Be to, kaip vėliau pamatysime, SQL leidžia apibrėžti virtualią lentelę, kuri didele dalimi imituoja neišskaidytą (apjungtą) lentelę. Tuomet darbas su dviem lentelėmis nedaug tesiskiria nuo darbo su viena lentele.

Lentelės, neatitinkančios 2NF, skaidymo procesas į kelias lenteles, atitinkančias 2NF, susideda iš kelių nesudėtingų žingsnių:

1) sukuriama nauja lentelė, kurios atributai yra pradinės lentelės atributai, įeinantys į f-sąryšį tarp nepirminių atributų (atributo) ir rakto dalies. Šio f-sąryšio determinantas tampa naujos lentelės raktu;
2) atributai, esantys dešinėje f-sąryšio pusėje, pašalinami iš pradinės lentelės;
3) jei pradinė lentelė nėra 2NF dėl daugiau nei vieno f-sąryšio, tai žingsniai 1 ir 2 kartojami kiekvienam tokiam sąryšiui.

Pavaizduosime pirmųjų dviejų žingsnių schemą. Tarkime, turime lentelę $L(\underline{A, B}, C, D)$, kurioje galioja f-sąryšiai: $AB \rightarrow CD$ ir $A \rightarrow D$. Tuomet lentelę L skaidome į dvi: $L_1(\underline{A, B}, C)$ ir $L_2(\underline{A}, D)$. Lentelėje L_1 galios f-sąryšis $AB \rightarrow C$, o lentelėje $L_2 - A \rightarrow D$.

Gali pasirodyti, kad naujose lentelėse galiojantys sąryšiai skiriasi nuo pradinėje lentelėje galiojusių. Tačiau taip nėra. Suskaidę lentelę nepraradome savybių, nes sąryšis $AB \rightarrow CD$ yra išvedamas iš $AB \rightarrow C$ ir $A \rightarrow D$, panaudojant kompozicijos taisyklę. Toks lentelių skaidymas, kai informacija neprarandama, vadinamas **dekompozicija be praradimo**. Suskaidžius pradinę lentelę L į dvi lenteles L_1 ir L_2, lentelės L_2 raktas A gali būti pirminiu, o lentelėje L_1 galima apibrėžti išorinį raktą, susiejant jos atributą A su L_2 pirminiu raktu.

Lentelė, kurią sudaro tik kai kurie kitos lentelės stulpeliai, vadinama **projekcija**. Taigi ir lentelė L_1, ir lentelė L_2 yra lentelės L projekcijos. Tai, kad išskaidžius lenteles neprarandama informacijos, garantuoja Hezo (I.J. Heath) teorema.

Hezo teorema. Tarkime, lentelėje *L(A, B, C)* galioja f-sąryšis $A \to B$, kur *A, B* ir *C* – lentelės atributų aibės. Tuomet lentelę *L* galima gauti jungiant jos projekcijas $L_1(A, B)$ ir $L_2(A, C)$.

Šią teoremą įrodyti paliksime skaitytojui, kaip uždavinį.

3.9. Trečioji normalinė forma

Lentelės $L(R)$ atributų aibės R poaibis C ($C \subset R$) **tranzityviai priklauso** nuo atributų aibės A *f*-sąryšių aibės F atžvilgiu, jei egzistuoja toks atributų aibės R poaibis B, kad A funkciškai apibrėžia B, bet ne atvirkščiai, bei B funkciškai apibrėžia C atžvilgiu F, t.y. $\exists B \subset R : A \to B \in F^+$, $B \to A \notin F^+$, $B \to C \in F^+$ ir $C \not\subset A \cup B$.

Lentelė $L(R)$ yra **trečios normalinės formos (3NF)** *f*-sąryšių aibės F atžvilgiu, jei ji yra 1NF ir nėra nepirminių atributų, tranzityviai priklausančių nuo rakto. **DB yra 3NF** *f*-sąryšių aibės F atžvilgiu tada ir tik tada, kai visos jos lentelės yra 3NF F atžvilgiu.

Tarkime, duomenų bazėje *Darbai* mums reikia saugoti ir aukštųjų mokyklų, kurias baigė darbuotojai – projektų vykdytojai, adresus. Vienas iš būdų tam pasiekti - papildyti lentelę *Vykdytojai* nauju stulpeliu, pvz., *AM_Adresas*. Tarkime, vietoj lentelės *Vykdytojai* turime lentelę *Vykdytojai_AM*:

Vykdytojai_AM (*Nr, Pavardė, Kvalifikacija, Kategorija, Išsilavinimas, AM_Adresas*).

Užpildę šią lentelę duomenimis gausime:

Vykdytojai_AM

Nr	*Pavardė*	**…**	*Išsilavinimas*	*AM_Adresas*
1	Jonaitis		VU	Universiteto 3, Vilnius
2	Petraitis		VU	Universiteto 3, Vilnius
3	Gražulytė		NULL	NULL
4	Onaitytė		VDU	Donelaičio 58, Kaunas
5	Antanaitis		VU	Universiteto 3, Vilnius

Kad būtų paprasčiau, pavaizdavome ne visus lentelės stulpelius.

Kaip ir lentelėje *Vykdytojai,* stulpelis *Nr* yra vienintelis šios lentelės raktas. Lentelėje *Vykdytojai_AM* atributas *AM_Adresas* funkciškai priklauso nuo atributo *Išsilavinimas*. Taigi lentelėje *Vykdytojai_AM* turime du f-sąryšius:

$Nr \to \{Pavardė, Kvalifikacija, Kategorija, Išsilavinimas, AM\_Adresas\}$,
$Išsilavinimas \to AM\_Adresas$.

Kadangi lentelės *Vykdytojai_AM* raktą sudaro tik vienas atributas, tai negali egzistuoti nepirminių atributų, priklausančių nuo rakto dalies. Tai reiškia, kad ši lentelė yra 2NF. Nors lentelė *Vykdytojai_AM* yra 2NF, jai būdinga duomenų perteklius bei anomalijos.

Kadangi lentelėje *Vykdytojai_AM* galioja f-sąryšis $Nr \to Išsilavinimas$, negalioja f-sąryšis $Išsilavinimas \to Nr$ ir bet to *Išsilavinimas* funkciškai apibrėžia *AM_Adresas*, tai šioje lentelėje nepirminis atributas *AM_Adresas* tranzityviai priklauso nuo lentelės rakto *Nr*. Todėl, ši lentelė nėra 3NF. Duomenų pertekliaus bei anomalijų nelieka, išskaidžius lentelę į dvi: *Vykdytojai* ir naują lentelę *AM_Adresai*(*Pavadinimas*, *Adresas*):

AM_Adresai

Pavadinimas	*Adresas*
VU	Universiteto 3, Vilnius
VDU	Donelaičio 58, Kaunas

Tiek lentelė *Vykdytojai,* tiek ir lentelė *AM_Adresai* yra 3NF. 3NF sumažino duomenų perteklių ir panaikino duomenų anomalijas.

Pavaizduosime tranzityvios vienos nepirminio atributo priklausomybės nuo rakto likvidavimo schemą. Tarkime, turime lentelę *L*(*A, B, C*), kurioje nepirminis atributas (atributų aibė) *C* tranzityviai priklauso nuo rakto *A,* kitaip tariant, galioja f-sąryšiai: $A \to B$ ir $B \to C$,

bei negalioja f-sąryšis $B \rightarrow A$. Tokia lentelė nėra 3NF. Tam, kad DB būtų 3NF, lentelę L išskaidome į dvi: $L_1(\underline{A, B})$ ir $L_2(\underline{B, C})$. Lentelėje L_1 galios f-sąryšis $A \rightarrow B$, o lentelėje L_2 - $B \rightarrow C$. Lentelės L_2 raktas B gali būti pirminiu raktu, o lentelėje L_1 galima apibrėžti išorinį raktą, susiejant jos atributą B su L_2 pirminiu raktu.

Literatūroje galima aptikti 3NF apibrėžimą, kuriame vietoj reikalavimo lentelei atitikti 1NF, yra reikalavimas atitikti 2NF. Galima įsitikinti, kad šie du reikalavimai, apibrėžiant 3NF, yra ekvivalentiški.

Teorema. Bet kuri lentelė, esanti 3NF *f*-sąryšių aibės F atžvilgiu, yra ir 2NF F atžvilgiu.

Įrodymas. Įrodysime prieštaros būdu. Tarkime, lentelė $L(R)$ yra 3NF, bet nėra 2NF *f*-sąryšių aibės F atžvilgiu. Tuomet $\exists a \in R : a$ - nepirminis ir a funkciškai priklauso nuo kurio nors rakto K dalies K', $K' \subset K \subseteq R$. Tai reiškia, kad iš F galima išvesti $K' \rightarrow a$ ($K' \rightarrow a \in F^+$) ir negalima išvesti $K' \rightarrow K$ ($K' \rightarrow K \notin F^+$), nes kitaip K' būtų L raktas, o ne K. Be to, $a \notin K$, kadangi K - raktas, o a - nepirminis. Taigi turime: $K \rightarrow K'$, ne $K' \rightarrow K$, $K' \rightarrow a$ bei $a \notin K$ ir juo labiau $a \notin K'$, t.y. $a \notin K \cup K'$. Tokiu būdu įrodėme, kad nepirminis atributas a tranzityviai priklauso nuo rakto K, o tai reiškia, kad lentelė $L(R)$ nėra 3NF; tai, savo ruožtu, prieštarauja prielaidai, padarytai įrodymo pradžioje.

Įrodę teoremą įsitikinome, kad dalinė priklausomybė reiškia ir tranzityvią priklausomybę. Kitaip tariant, jei lentelė nėra 2F, tai joje egzistuoja nepirminis atributas, tranzityviai priklausantis nuo rakto.

3.10. Boiso-Kodo normalinė forma

3NF, kaip ir 1NF bei 2NF, pirmasis apibrėžė Kodas (E.F.Codd). Kurį laiką buvo skaitoma, kad 3NF yra praktiškai pakankama. Vėliau paaiškėjo, kad 3NF nevisiškai tinka lentelėms, kurios turi du ar daugiau raktų, bent du iš jų yra sudėtiniai (sudaryti iš keleto atributų) raktai ir jie tarpusavyje kertasi (turi bent vieną bendrą atributą). Todėl tas 3NF apibrėžimas buvo pakeistas griežtesniu Boiso-Kodo (Boyce-Codd) apibrėžimu. Lentelė yra **Boiso-Kodo normalinės formos (BKNF)**, jei kiekvieno netrivialaus f-sąryšio determinantas yra raktas. Gana nesunku įsitikinti, kad lentelė, esanti BKNF, yra ir 3NF. BKNF įgavo tokį pavadinimą dėl to, kad jos suformulavimo metu jau buvo suformuluota 4NF. **DB yra BKNF** *f*-sąryšių aibės F atžvilgiu tada ir tik tada, kai visos jos lentelės yra BKNF F atžvilgiu.

Tarkime, projektai vykdomi etapais ir lentelėje *Etapai* yra saugoma informacija apie projektų etapų pabaigas:

Projektų_Etapai (*Nr, Pavadinimas, Etapas, Pabaiga*).

Šioje lentelėje pažymėtas vienas (pirminis) raktas, kurį sudaro du pabraukti stulpeliai. Tarus, kad negali būti dviejų projektų su vienodais pavadinimais, lentelė turi ir antrą raktą {*Pavadinimas, Etapas*}. Matome, kad abu raktai turi vieną bendrą stulpelį *Etapas*. Šiai lentelei būdingi tokie f-sąryšiai:

{*Pavadinimas, Etapas*} $\rightarrow$ {*Nr, Pabaiga*},
{*Nr, Etapas*} $\rightarrow$ {*Pavadinimas, Pabaiga*},
Nr $\rightarrow$ *Pavadinimas*,
Pavadinimas $\rightarrow$ *Nr*.

Išanalizavę šiuos f-sąryšius pastebėsime, kad lentelė *Projektų_Etapai* yra 3NF, tačiau jai būdingi aukščiau minėti požymiai. Užpildžius lentelę duomenimis nesunku pastebėti, kad lentelėje duomenys dubliuojami.

Projektų_Etapai

Nr	*Pavadinimas*	*Etapas*	*Pabaiga*
1	Studentų apskaita	1	2001.06.30
1	Studentų apskaita	2	2001.12.31
2	Buhalterinė apskaita	1	2001.05.31
2	Buhalterinė apskaita	2	2001.12.31
3	WWW svetainė	1	2001.07.31

Tarp lentelės *Projektų_Etapai* atributų galioja net du f-sąryšiai (*Nr* → *Pavadinimas* ir *Pavadinimas* → *Nr*), kurių determinantai nėra raktai. Todėl lentelė *Projektų_Etapai* nėra BKNF. Siekiant BKNF, lentelę reikia suskaidyti į dvi lenteles. Galimi du skaidymo variantai:

1) *Pavadinimai_Etapai*(*Pavadinimas, Etapas, Pabaiga*),
   Nr_Pavadinimai(*Nr, Pavadinimas*);
2) *Nr_Etapai*(*Nr, Etapas, Pabaiga*),
   Nr_Pavadinimai(*Nr, Pavadinimas*).

Abu lentelių rinkiniai yra BKNF. Teoriškai jie lygiaverčiai. Viena lentelė *Nr_Pavadinimai* yra bendra abiem atvejams. Ši lentelė turi du raktus ir nė vieno nepirminio atributo. Kitos dvi lentelės skiriasi tik vienu stulpeliu. Kadangi stulpelio *Pavadinimas* reikšmės yra ilgesnės negu stulpelio *Nr*, tai galima sakyti, kad lentelė *Nr_Etapai* yra pranašesnė už *Pavadinimai_Etapai*. Tas pranašumas dar sustiprėtų tarus, kad projektų pavadinimai gali keistis. Taigi grynai praktiniu požiūriu antrasis variantas yra geresnis už pirmąjį.

Jei lentelė nėra BKNF, tai egzituoja netrivialus f-sąryšis, kurio determinantas nėra raktas. Tai, kad netrivialaus f-sąryšio $A \to B \in F$ determinantas A nėra lentelės $L(R)$ ($A, B \subset R$) raktas, reiškia, kad iš atributų aibės A sąryšių aibėje F negalima išvesti visų lentelės atributų, t.y. $A^+ \subset R$. Norint patikrinti, ar f-sąryšio $A \to B \in F$ determinantas A nėra lentelės $L(R)$ raktas, galima pasinaudoti ankščiau pateiktu algoritmu atributų aibės A uždariniui A^+ f-sąryšių aibės F atžvilgiu rasti.

Jei lentelėje egzistuoja f-sąryšis, kurio determinantas nėra raktas, tai iš pradinės lentelės pašalinami atributai esantys tokio sąryšio dešinėje pusėje, o iš visų dalyvaujančių sąryšyje atributų sudaroma nauja lentelė. Tarkime, lentelėje $L(A, B, C)$ galioja netrivialus f-sąryšis $A \to B$ ir A nėra šios lentelės raktas. Tuomet lentelę L skaidome į dvi: $L_1(A, C)$ ir $L_2(\underline{A}, B)$. Lentelės L_2 raktas A gali būti pirminiu, o lentelėje L_1 galima apibrėžti išorinį raktą, susiejant jos atributus A su L_2 pirminiu raktu.

3.11. Ketvirtoji normalinė forma

Ketvirtajai normalinei formai apibrėžti funkcinio sąryšio sąvokos nepakanka. Tam įvedama daugiareikšmio sąryšio sąvoka, apibendrinanti funkcinį sąryšį.

Lentelėje $L(A, B, C)$ yra **daugiareikšmis** (angl. *multi valued*) **sąryšis**, sutrumpintai **MV-sąryšis**, $A \twoheadrightarrow B$ tada ir tik tada, kai atributų B reikšmių aibė, atitinkanti bet kurias atributų aibių A ir C reikšmes, priklauso tik nuo A ir nepriklauso nuo atributų C reikšmių. MV-sąryšis gali būti ir kitaip apibrėžiamas. Lentelėje $L(A, B, C)$ galioja MV-sąryšis $A \twoheadrightarrow B$, jei, egzistuojant dviem lentelės eilutėms (a, b, c) ir (a, b′, c′), būtinai egzistuoja ir trečia eilutė (a, b′, c), kur a, b, b′, c, c′ žymi atitinkamų atributų reikšmes.

Funkcinio sąryšio atveju kiekvieną determinanto reikšmę atitinka tik viena nuo jo priklausomo atributo reikšmė. Daugiareikšmio sąryšio atveju atributų reikšmę gali atitikti kelios kito atributo reikšmės. MV-sąryšio sąvoka yra f-sąryšio apibendrinimas ta prasme, kad kiekvienas f-sąryšis kartu yra ir MV-sąryšis. Tiksliau, f-sąryšis – tai MV-sąryšis, kuriame determinantą atitinkanti reikšmių aibė visada yra iš vieno elemento. Kaip ir f-sąryšių atveju, mus domina tik tokie MV-sąryšiai, kurie lentelėje galioja visada, o ne kuriuo nors konkrečiu momentu. **MV-sąryšis $A \twoheadrightarrow B$ yra netrivialus** lentelėje $L(R)$, jei nė vienas atributas iš B neįeina į A ir, be atributų A ir B, lentelėje yra dar ir kitų atributų, t.y. jei $A \cap B = \varnothing$ ir $A \cup B \subset R$.

Tarkime, kiekvienas firmos darbuotojas gali turėti kelias kvalifikacijas ir mokėti kelias užsienio kalbas, o informacijai apie tai skirta lentelė *Išsilavinimas*(*Nr*, *Kvalifikacija*, *Kalba*), kurios stulpeliai atitinkamai reiškia darbuotojo numerį, įgytos kvalifikacijos pavadinimą ir užsienio kalbą, kurią jis moka. Ši lentelė turi vieną raktą, kurį sudaro visi trys lentelės atributai. Tarkime, darbuotojas Nr. 1 turi informatiko ir fiziko kvalifikaciją, taip pat moka dvi užsienio kalbas: anglų ir vokiečių. Lentelės fragmentas su duomenimis apie šį darbuotoją gali būti toks:

Išsilavinimas

Nr	*Kvalifikacija*	*Kalba*
1	Informatikas	Anglų
1	Informatikas	Vokiečių
1	Fizikas	Anglų
1	Fizikas	Vokiečių

Lentelėje *Išsilavinimas* yra du MV-sąryšiai:

$Nr \rightarrow\!\!\!\rightarrow Kvalifikacija,$
$Nr \rightarrow\!\!\!\rightarrow Kalba.$

Iš pirmo žvilgsnio gali pasirodyti, kad turimai informacijai išreikšti pakanka tik dviejų eilučių lentelėje, pavyzdžiui, pirmos ir paskutinės. Tačiau tada galėtų atrodyti, kad žmogus moka tik anglų kalbą, kai dirba informatiko darbą, ir tik vokiečių kalbą, kai fiziko. Taip, žinoma, neturėtų būti. Vienintelis būdas išreikšti, kad kvalifikacijos niekaip nesusiję su užsienio kalbomis, yra kiekvieną kalbą pakartoti su kiekviena kvalifikacija. Tai akivaizdžiai sukelia duomenų perteklių ir su tuo susijusias duomenų anomalijas. Pastebėsime, kad lentelėje nėra nepirminių atributų, todėl ji yra BKNF.

Reliacinėje teorijoje nesunkiai įrodoma, kad jei lentelėje $L(A, B, C)$ galioja MV-sąryšis $A \rightarrow\!\!\!\rightarrow B$, tai galioja ir MV-sąryšis $A \rightarrow\!\!\!\rightarrow C$. Tokiu būdu netrivialūs daugiareikšmiai sąryšiai visuomet sudaro poras. Todėl MV-sąryšių pora dažnai vaizduojama taip: $A \rightarrow\!\!\!\rightarrow B \mid C$. Lentelėje *Išsilavinimas* galioja $Nr \rightarrow\!\!\!\rightarrow Kvalifikacija \mid Kalba$.

Lentelė $L(R)$ yra **ketvirtosios normalinės formos** (**4NF**) tada ir tik tada, kai egzistuojant netrivialiam MV-sąryšiui $A \rightarrow\!\!\!\rightarrow B$, $A \subset R$, $B \subset R$, visi kiti atributai, funkciškai priklauso nuo A. Neformaliai lentelė yra 4NF tik tuomet, kai kiekvienas joje galiojantis netrivialus sąryšis išreiškia atributų funkcinę priklausomybę nuo (galimo) rakto. **DB yra 4NF** tada ir tik tada, kai visos jos lentelės yra 4NF.

Teiginys. Lentelė L yra 4NF, jei visi lentelės netrivialūs daugiareikšmiai sąryšiai faktiškai yra funkciniai sąryšiai, apibrėžiantys priklausomybę nuo lentelės galimo rakto.

Paėmus kurį nors vieną iš dviejų MV-sąryšių, galiojančių lentelėje *Išsilavinimas*, pavyzdžiui, $Nr \rightarrow\!\!\!\rightarrow Kvalifikacija,$ likęs atributas *Kalba* nėra f-priklausantis nuo *Nr*. Todėl ši lentelė nėra 4NF.

Feigino teorema. Tarkime, A, B ir C yra lentelės $L(A, B, C)$ atributų aibės. Tuomet lentelė L gali būti gaunama sujungiant jos projekcijas $L_1(A, B)$ ir $L_2(A, C)$ tada ir tik tada, kai $A \rightarrow\!\!\!\rightarrow B \mid C$.

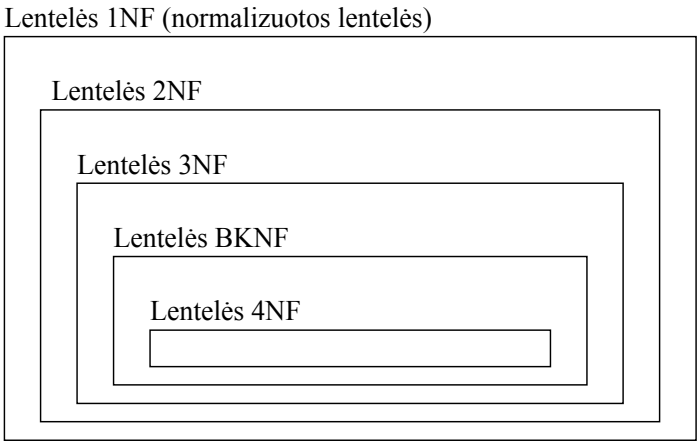
Prisiminus, kad MV-sąryšis yra f-sąryšio apibendrinimas, nesunku pastebėti, jog Feigino (R. Fagin) teorema yra Hezo teoremos apibendrinimas. Sekant Feigino teorema, lentelę *Išsilavinimas* galima suskaidyti į dvi projekcijas:

Kvalifikacijos(*Nr*, *Kvalifikacija*),
Kalbos(*Nr*, *Kalba*).

Abiejose šiose lentelėse yra po vieną MV-sąryšį, tačiau abu jie yra trivialūs, t.y. juose dalyvauja visi atitinkamos lentelės atributai. Todėl abi šios dvi lentelės yra 4NF.

3.12. Normalinių formų tarpusavio ryšys

Kiekvienas f-sąryšis yra ir MV-sąryšis. Tai reiškia, kad jei lentelė nėra BKNF, tai ji nėra ir 4NF. Kitaip tariant, kiekviena lentelė, esanti 4NF, kartu yra ir BKNF. BKNF pavadinime neatsispindi normalinės formos vieta jų hierarchijoje, kadangi 4NF jau buvo anksčiau apibrėžta. Taigi kiekviena aukštesnė NF patenkina vis griežtesnius reikalavimus lentelės reliacinei schemai.



3.2 pav. Normalinių formų tarpusavio padėtis

Realiuose uždaviniuose, jei duomenų bazė yra 3NF ar BKNF ar net 4NF, sakoma, kad to pakanka. Reliacinėje teorijoje nagrinėjamos ir aukštesnės normalinės formos nei ketvirtoji. Tačiau aukštesnės normalinės formos svarbios daugiau teoriniu, negu praktiniu požiūriu.

Kaip matyti iš šiame skyriuje pateiktų pavyzdžių, sudarant duomenų bazės schemą, įmanoma pasiekti BKNF, netgi 4NF, ir be formalių metodų. Į visas nagrinėtas normalines formas galima žiūrėti kaip į sukauptų praktinių žinių formalizavimą. Tačiau šis formalizavimas nėra savitikslis. Formalizuotos žinios leidžia automatizuoti šį procesą. Dabar praktikoje taikoma ne viena automatizuoto DB sudarymo priemonė, kuriomis generuojamos DB struktūros, esančios 3NF, BKNF ar net 4NF.