2. Informacijos išrinkimas

2.1. Duomenų bazė "Darbai"

Duomenų bazės duomenys yra žymiai dažniau peržiūrimi negu keičiami ar įvedami. Dar rečiau ir tik nedaugeliui vartotojų (DB administratoriams ir taikomųjų sistemų kūrėjams) tenka kurti lenteles ir duomenų bazes. Vienas iš paprasčiausių būdų susipažinti su DB ir SQL kalba yra pradėti nuo jau esamų duomenų duomenų bazėje peržiūrėjimo.

Tarkime, egzistuoja DB *Darbai*, kurioje yra trys lentelės *Vykdytojai*, *Projektai* ir *Vykdymas*. Norėdami išskirti DB objektų vardus, rašysime juos kursyvu. Tarkime, kad šios lentelės užpildytos duomenimis, kaip parodyta 2.1 pav.

Vykdytojai

Nr	*Pavardė*	*Kvalifikacija*	*Kategorija*	*Išsilavinimas*
1	Jonaitis	Informatikas	2	VU
2	Petraitis	Statistikas	3	VU
3	Gražulytė	Inžinierius	1	NULL
4	Onaitytė	Vadybininkas	5	VDU
5	Antanaitis	Informatikas	3	VU

Projektai

Nr	*Pavadinimas*	*Svarba*	*Pradžia*	*Trukmė*
1	Studentų apskaita	Maža	2001.01.01	12
2	Buhalterinė apskaita	Vidutinė	2001.03.01	10
3	WWW svetainė	Didelė	2001.06.01	2

Vykdymas

Projektas	*Vykdytojas*	*Statusas*	*Valandos*
1	1	Programuotojas	30
1	2	Dokumentuotojas	100
1	3	Testuotojas	100
1	4	Vadovas	100
2	1	Programuotojas	300
2	2	Analitikas	250
2	4	Vadovas	100
3	1	Programuotojas	250
3	2	Vadovas	400
3	3	Dizaineris	150

2.1 pav. Duomenų bazė *Darbai*

Duomenų bazėje *Darbai* yra informacija apie realiai neegzistuojančioje informacinių technologijų firmoje vykdomus projektus (atliekamus darbus, užsakymus). Lentelėje *Vykdytojai* yra užregistruoti visi projektų vykdytojai (firmos darbuotojai). Šioje lentelėje yra surašyta vykdytojų tapatumo firmoje numeriai, vykdytojų pavardės, įgyta kvalifikacija, tam tikra kategorija, nusakanti darbuotojo kvalifikacijos lygmenį, bei kur buvo įsigytas išsilavinimas. Viena lentelės eilutė skirta vienam darbuotojui. Lentelėje *Projektai* yra informacija apie vykdomus firmoje projektus. Lentelės eilutėje yra unikalus projekto tapatumo numeris, pavadinimas, svarbos lygmuo ir vykdymo trukmė mėnesiais. Trečioje lentelėje *Vykdymas* yra informacija apie tai, kokius darbus kurie vykdytojai atlieka. Lentelės eilutėje yra projekto ir darbuotojo numeriai, statusas, kuriame darbuotojas dalyvauja projekte, ir kiekis valandų, kurias vykdytojas skiria projektui. Kiekviena reikšmių pora (*Projektas*, *Vykdytojas*) lentelėje *Vykdymas* yra unikali, t.y. jei darbuotojas dalyvauja kokiame nors projekte, tai tam skirta tik viena eilutė.

Pastebėsime, kad DB *Darbai* yra grynai mokomoji. Ji neatspindinti realios situacijos. Pateikta DB yra per daug supaprastinta, kad ji atitiktų realių projektų vykdymą realioje firmoje. Šios DB pagrindinė paskirtis – išmokti sudaryti užklausas.

Užklausose, kaip ir daugumoje kitų SQL sakinių, nėra nurodoma, kuriai DB yra skirtas sakinys. Duomenų bazės vardas nurodomas prieš veiksmus su duomenimis. DB, kuriai yra skiriami tolimesni SQL sakiniai, nurodoma sakiniu:

`CONNECT TO <`DB vardas`>.`

Čia frazė "CONNECT TO" yra bazinis (raktinis) SQL žodis. Eilutės gale padėtas taškas nėra SQL sakinio dalis, tai teksto sakinio pabaiga. SQL standartas nenumato sakinio pabaigos požymio, nors kai kuriose komercinėse DBVS yra vartojamas sakinio pabaigos požymis, pvz., kabliataškis. Pačiame SQL sakinyje gali būti vartojami įvairūs atskyrėjai. Raktinius kalbos žodžius rašysime kitu šriftu, kad šie išsiskirtų. SQL baziniai žodžiai gali būti rašomi didžiosiomis ir mažosiomis raidėmis. Tas pat taikoma ir DB objektų (lentelių, stulpelių ir kt.) pavadinimams – raidžių registras neturi įtakos. Mes bazinius žodžius rašysime didžiosiomis raidėmis.

Sakinys, kuriuo nurodoma duomenų bazė, su kuria toliau bus dirbama, yra "aktyvus" – jo vykdymo metu vyksta fizinis ir loginis ryšio nustatymas su DB. Jei šio sakinio vykdymo metu DBVS negalės nustatyti ryšio, pavyzdžiui, dėl fizinio ryšio nebuvimo su DB serveriu, tai sakinys nebus įvykdytas. DBVS informuoja vartotoją apie SQL sakinio įvykdymo sėkmę atitinkamu pranešimu. Tam, kad pradėti darbą su DB *Darbai*, reikia įvykdyti sakinį

`CONNECT TO` *Darbai.*

Pabaigus darbą, loginis ryšys su DB nutraukiamas sakiniu

`CONNECT RESET.`

Pastebėsime, kad nereikia palaikyti ryšio su DB be reikalo, kadangi tai susiję su tam tikrų resursų sąnaudomis tiek vartotojo darbo vietoje, tiek ir serveryje. Rekomenduojama nustatyti ryšį prieš pat pradedant darbą su DB ir nutraukti jį iš karto, kai tik nustojama (laikinai ar visiškai) naudotis ja.

2.2. Sakinys SELECT

SQL sakinys `SELECT` yra pagrindinis sakinys DB-je esantiems duomenims peržiūrėti. Šis sakinys turi daug įvairiausių variantų ir galimybių. Išsamiai šio sakinio sintaksei pateikti prireiktų keleto puslapių. Sakinį bendriausiu atveju galima užrašyti taip:

```
SELECT   [DISTINCT]  <stulpelių vardai>
         FROM        <lentelių vardai>
         [WHERE      <paieškos sąlyga>]
         [GROUP BY   <stulpelių vardai> [HAVING  <Paieškos sąlyga>]]
         [ORDER BY   <stulpelių vardai>] .
```

Įvykdžius šį sakinį, DBVS suformuoja ir pateikia vartotojui užklausos rezultatą - laikiną lentelę, kuri egzistuoja tik užklausos rezultato peržiūros metu. Sistemos vartotojas rezultatą gali pamatyti monitoriaus ekrane ar apdoroti jį programoje.

Nesunku pastebėti, kad pati paprasčiausia užklausa atrodo taip:

`SELECT` <stulpelių vardai> `FROM` <lentelės vardas>.

Tokios užklausos rezultatą sudaro išvardinti nurodytos lentelės stulpeliai. Į rezultatą įtraukiamos visos lentelės eilutes. Sakinio dalį iki bazinio žodžio FROM vadinsime SELECT fraze. Šia fraze nusakomi stulpeliai, kurie turi patekti į užklausos rezultatą.

Kadangi lentelėje gali būti labai daug eilučių, tai rezultatas, kurį sudaro lentelės visos eilutės, gali būti nepriimtinas peržiūrai. Sakinį papildžius fraze WHERE <paieškos sąlyga>, užklausos rezultatą sudarys tik tos eilutės, kurios tenkins nurodytą sąlygą. Detaliau paieškos sąlygų formavimą bei kitas sakinio dalis aptarsime tolimesniuose skyreliuose.

2.3. Paprasčiausios užklausos duomenims išrinkti

SQL sakinį SELECT, kaip ir daugelį kitų sakinių, paaiškinsime nagrinėdami pavyzdžius. Tarkime, mes norime sužinoti visų informatikų – projektų vykdytojų pavardes ir kategorijas. Nesunku pastebėti, kad visa ši informacija yra vienoje lentelėje *Vykdytojai*. Reikiamą informaciją galima sužinoti įvykdžius sakinį:

SELECT *Pavardė, Kategorija* FROM *Vykdytojai* WHERE *Kvalifikacija* = 'Informatikas'.

Šioje užklausoje pavartota frazė 'Informatikas' yra simbolių eilutė – SQL konstanta. Simbolių eilutė (tekstinių duomenų konstanta) SQL sakiniuose rašoma tarp apostrofų. Šios užklausos rezultatas bus laikina lentelė, kurią sudarys du stulpeliai, ir į ją įeis tos lentelės *Vykdytojai* eilutės, kuriose stulpelio *Kvalifikacija* reikšmė yra 'Informatikas':

Pavardė	*Kategorija*
Jonaitis	2
Antanaitis	3

Lentelės stulpelius užklausoje galima patikslinti, lentelės vardu:

SELECT *Vykdytojai.Pavardė, Vykdytojai.Kategorija* FROM *Vykdytojai*
WHERE *Vykdytojai.Kvalifikacija* = 'Informatikas'.

Stulpelių vardų patikslinimai nėra būtini, jei SQL sakinyje vartojama tik viena lentelė, kaip pastarajame pavyzdyje. Kaip pamatysime vėliau, stulpelių vardus gali prireikti patikslinti, kai sakinyje dalyvauja keletas lentelių. Patikslinant vardus, dažnai nėra patogu rašyti gana ilgus lentelių vardus. Šito galima išvengti, naudojant lentelių vardų sinonimus, pvz.:

SELECT *A.Pavardė, A.Kategorija* FROM *Vykdytojai* AS *A*
WHERE *A.Kvalifikacija* = 'Informatikas'.

Paminėjus lentelės *Vykdytojai* vardą frazėje FROM, čia pat jam suteikiamas sinonimas *A* (raktinis žodis AS gali būti ir praleistas). Lentelei suteiktas sinonimas galioja tik tame viename SQL sakinyje.

Tarkime, mums reikia sužinoti visas aukštąsias mokyklas, kurias yra baigę projektų vykdytojai. Šį uždavinį galime išspręsti tokia užklausa:

SELECT *Išsilavinimas* FROM *Vykdytojai* ,

kurios rezultatas bus:

Išsilavinimas
VU
VU
NULL
VDU
VU

Kadangi užklausoje nėra nurodyta jokia sąlyga, tai rezultate yra tiek eilučių, kiek jų yra lentelėje *Vykdytojai*. Šiame rezultate tekstas 'VU' kartojasi tris kartus, kas visiškai nebūtina. Vienodos eilutės galėtų netgi trukdyti, jei jų būtų daug. Vienodų eilučių galima išvengti panaudojant bazinį žodį DISTINCT. Paskutinį sakinį perrašę taip:

SELECT DISTINCT *Išsilavinimas* FROM *Vykdytojai* ,

gausime rezultatą, kurį sudarys tik trys eilutės. Taigi, neįtraukus į užklausą bazinio žodžio DISTINCT, užklausos rezultate galimos vienodos eilutės, tiksliau, vienodos eilutės nėra pašalinamos.

Kai kurie stulpeliai gali būti apskaičiuojami. Tarkime, kad lentelėje *Projektai* projektų trukmė nurodyta mėnesiais, o mes tą laiką norime sužinoti dienomis. Paprastumo dėlei tarkime, kad kiekviename mėnesyje yra 30 dienų. Uždavinį išspręsime sakiniu:

SELECT *Pavadinimas, Trukmė* * 30 FROM *Projektai* .

Šiame sakinyje yra pavartotas skaičius 30 – tai skaitinių duomenų konstanta. Skaičiai SQL sakiniuose rašomi be jokių papildomų atskyrėjų. Šios užklausos rezultatas yra laikina lentelė, turinti du stulpelius. Pirmojo stulpelio reikšmės – tai lentelės *Projektai* stulpelio *Pavadinimas* reikšmės. Antrojo stulpelio reikšmės – tai tos pačios lentelės stulpelio *Trukmė* reikšmės, padaugintos iš 30. Antrasis stulpelis, skirtingai nuo pirmojo, nėra paprastas lentelės stulpelis, tai aritmetinis reiškinys, kuriame gali būti ir keli lentelės stulpeliai. Kai užklausos SELECT frazėje vartojamas reiškinys, rezultato stulpelio pavadinimas gali būti dviprasmiškas. Tokiais atvejais sistema, vaizduodama užklausos rezultatą, stulpeliui suteikia tarnybinį pavadinimą, kuris atitinka stulpelio eilės numerį užklausoje (antrajam stulpeliui suteikiamas vardas "2"). Tai nevisada priimtina. Vartotojas gali pats nurodyti norimą stulpelio pavadinimą, pavartojant bazinį žodį AS. Suteikiamas stulpeliui pavadinimas turi tenkinti tuos pačius reikalavimus, kaip ir duomenų bazės stulpelio pavadinimas. Norint stulpeliui suteikti pavadinimą, kuriame būtų vienas ar keli specialieji simboliai, būtina stulpelio pavadinimą rašyti tarp kabučių:

SELECT *Pavadinimas, Trukmė* * 30 AS "*Trukmė dienomis*" FROM *Projektai* .

Šioje užklausoje antrojo stulpelio pavadinimas yra tarp kabučių, nes jį sudaro du žodžiai (tiksliau, pavadinime yra tarpas). Užklausos rezultatas:

Pavadinimas	*Trukmė dienomis*
Studentų apskaita	360
Buhalterinė apskaita	300
WWW svetainė	60

Pastebėsime, kad ir anksčiau mes vartojome stulpelių pavadinimus, kuriuos kai kuriose komercinėse DBVS reikėtų rašyti tarp kabučių arba jie apskritai būtų neleistini. Tai stulpelių pavadinimai, kuriuose pavartotos lietuvių kalbos alfabeto raidės, nesančios lotynų alfabete, pvz., *Pavardė*, *Pradžia*. Daugumoje komercinių DBVS duomenų bazės objektų (lentelių, stulpelių ir kt.) pavadinimuose (**SQL varduose**) leidžiama vartoti tik lotynų alfabeto raides. SQL kalboje raidėmis taip pat laikomi trys pavieniai simboliai: $, # ir @. SQL vardai turi prasidėti raide. Sudarant SQL vardus galima vartoti raides, skaitmenis bei pabraukimo ženklą (_). Mes, patogumo dėlei, laikysime, kad, sudarant SQL vardus, galima vartoti visas lietuvių kalbos raides. Tarp kabučių rašysime tik vardus, kuriuose yra specialieji simboliai.

Ankstesniajame pavyzdyje, SELECT frazėje, pavartojome aritmetinį reiškinį. Ypatingas reiškinio atvejis yra konstanta. Pateiksime užklausą su konstanta SELECT frazėje:

```
SELECT Pavadinimas,
       'Trukmė dienomis: ' AS Pastaba,
       Trukmė * 30 AS "Trukmė dienomis"
FROM Projektai .
```

Užklausos antrojo stulpelio reikšmė yra konstanta. Visos šio stulpelio reikšmės yra vienodos ir lygios tekstinei konstantai 'Trukmė dienomis: ':

Pavadinimas	*Pastaba*	*Trukmė dienomis*
Studentų apskaita	Trukmė dienomis:	360
Buhalterinė apskaita	Trukmė dienomis:	300
WWW svetainė	Trukmė dienomis:	60

Atkreipsime dėmesį į tai, kad užklausoje pavartoti tiek apostrofai, tiek ir kabutės. Priminsime, kad apostrofais žymimos tekstinės konstantos, o kabutėmis – lentelės stulpelių bei kitų duomenų bazės objektų pavadinimai.

Pastebėsime, kad konstanta gali sudaryti ir visą užklausos rezultatą, pvz., užklausos

SELECT 'Trukmė dienomis' AS *Pastaba* FROM *Projektai*

rezultatas bus suformuotas tik iš vienos konstantos:

Pastaba
Trukmė dienomis
Trukmė dienomis
Trukmė dienomis

Nežiūrint į tai, kad iš užklausos formuluotės aišku, kokia reikšmė sudarys užklausos rezultate, DBVS vistiek turi peržiūrėti lentelę *Projektai*, nes rezultatą sudaro tiek eilučių, kiek jų yra lentelėje.

Norint užklausos rezultate gauti visų lentelės stulpelių reikšmes, vietoj to, kad išvardinti juos SELECT frazėje, galima pavartoti specialųjį simbolį '*'. Visi duomenys, esantys lentelėje *Vykdytojai* bus pateikti įvykdžius vieną iš šių užklausų:

SELECT * FROM *Vykdytojai*,

SELECT *Vykdytojai*.* FROM *Vykdytojai*,

SELECT A.* FROM *Vykdytojai* A.

Dažnai didelę duomenų aibę žymiai patogiau peržiūrėti, kai ši išdėstyta pagal vieną ar kitą kriterijų. DBVS sutvarko užklausos rezultatą pagal tam tikrą kriterijų (kriterijus), kai sakinyje SELECT yra tvarką pibrėžianti frazė ORDER BY. Šioje frazėje nurodomi stulpelių vardai arba stulpelių eilės numeriai, atskiriant juos kableliais. Po kiekvieno stulpelio vardo (arba numerio) galima rašyti vieną iš bazinių žodžių: ASC (angl. *ascending*) arba DESC (angl. *descending*), kuriais nustatoma reikšmių didėjimo ar mažėjimo tvarka. Jei nepavartotas nei vienas iš šių žodžių, tai, pagal nutylėjimą, suprantama ASC. Pavyzdžiui, įvykdžiusi užklausą

SELECT *Pavardė, Išsilavinimas, Kategorija* FROM *Vykdytojai*
ORDER BY 2, *Kategorija* DESC,

sistema pateiks duomenis apie visų vykdytojų pavardes, jų išsilavinimą ir kategoriją, sutvarkytus pagal antrojo ir trečiojo stulpelių reikšmes. Rezultatas bus pateiktas leksikografine tvarka pagal išsilavinimą. Esant vienodiems aukštųjų mokyklų pavadinimams, eilučių tarpusavio padėtis bus nustatoma pagal kategoriją, jų mažėjimo tvarka. Atkreipsime dėmesį į tai, kad antrojo stulpelio reikšmės yra tekstinės. SQL kalboje tarp tekstinių duomenų yra nustatyta leksikografinė tvarka.

2.4. Reiškiniai

Užklausose jau vartojome pačius paprasčiausius reiškinius: konstantas, stulpelių pavadinimus ir aritmetinius reiškinius. Aptarsime reiškinius detaliau.

Vardinės konstantos (sisteminiai pseudokintamieji) - tai reikšmės, kurias saugo DBVS ir kurias galima vartoti SQL sakiniuose. Paprastai, tai išorinė DBVS požiūriu informacija, žyminti einamąją sisteminę datą (CURRENT DATE), laiką (CURRENT TIME), datą ir tikslų laiką (CURRENT TIMESTAMP), sistemos vartotojo vardą (USER) ir pan. Kai kuriose komercinėse DBVS, sisteminės reikšmės pasiekiamos ne per vardines konstantas, bet kviečiant specialias funkcijas.

Funkcija - tai operacija, nusakoma funkcijos vardu ir apskliaustais lenktiniais skliaustais argumentais, kurie tarpusavyje atskiriami kableliais (atskiru atveju argumentų gali ir nebūti). Funkcija visuomet grąžina tam tikrą rezultatą (tai gali būti ir specialioji reikšmė NULL). Funkcijos yra skirtomos į agregatines (stulpelių) ir skaliarines.

Agregatinės funkcijos - tai funkcijos, kurias galima pritaikyti eilučių aibėms: visoms eilutėms, tenkinančioms frazėje WHERE nurodytą sąlygą, arba eilučių grupei, apibrėžiamai fraze GROUP (eilučių grupavimą aptarsime vėliau). Agregatinės funkcijos bet kokiam eilučių rinkiniui apskaičiuoja vieną reikšmę. Šios rūšies funkcijos dažnai vartojamos SELECT frazėje, nurodant stulpelį, pagal kurio reikšmes apskaičiuojamas funkcijos rezultatas.

Agregatinės funkcijos vartojamos ir frazėje HAVING, kai funkcijos reikšmė skaičiuojama kiekvienai eilučių grupei. Keletas dažnai vartojamų agregatinių funkcijų:

Agregatinė funkcija	Rezultatas
SUM([DISTINCT] <reiškinys>)	(Skirtingų) ne NULL reikšmių suma
AVG([DISTINCT] <reiškinys>)	(Skirtingų) ne NULL reikšmių vidurkis
COUNT([DISTINCT] <reiškinys>)	(Skirtingų) ne NULL reikšmių kiekis
COUNT(*)	Eilučių kiekis aibėje
MAX(<reiškinys>)	Maksimali reikšmė
MIN(<reiškinys>)	Minimali reikšmė

Jei visos funkcijoje nurodyto stulpelio reikšmės yra NULL, arba stulpelis yra tuščias, tai funkcijų SUM, AVG, MIN, MAX rezultatas yra NULL, o funkcijos COUNT – nulis.

Pateiksime keletą užklausų su agregatinėmis funkcijomis. Visų vykdytojų skaičius arba, tiksliau, eilučių kiekis lenteleje *Vykdytojai* gali būti sužinomas sakiniu:

SELECT COUNT(*) FROM *Vykdytojai*.

Visų vykdytojų, dalyvaujančių bent viename projekte, skaičius - sakiniu:

SELECT COUNT(DISTINCT *Vykdytojas*) FROM *Vykdymas*.

Pirmosios (iš pastarųjų dviejų) užklausos rezultate gausime vieną eilutę, kurioje bus pateiktas skaičius 5. Antrosios užklausos rezultatas - taip pat viena eilutė, bet joje bus skaičius 4 – tiek skirtingų vykdytojų numerių yra paminėta lentelės *Vykdymas* stulpelyje *Vykdytojas*.

Bendrą kiekį valandų, kurias vykdytojas Nr. 1 skiria visiems projektams, galima sužinoti įvykdžius užklausą:

SELECT SUM(*Valandos*) AS “Vykdytojo Nr. 1 valandos” FROM *Vykdymas*
WHERE *Vykdytojas* = 1.

Kadangi šis vykdytojas dalyvauja trijuose projektuose, t.y. lentelėje *Vykdymas* yra 3 eilutės tenkinančios nurodytą sąlygą (*Vykdytojas* = 1), ir 30 + 300 + 250 = 580, tai pastarosios užklausos rezultatas bus:

Vykdytojo Nr. 1 valandos
580

Skaliarinės funkcijos argumentas visuomet yra viena reikšmė. Funkcija gali turėti ir kelis argumentus, tačiau kiekvienas argumentas – viena reikšmė, o ne reikšmių aibė, kaip agregatinėje funkcijoje. Vartojant skaliarines funkcijas frazėje WHERE, funkcijos rezultatas apskaičiuojamas tikrinant paieškos sąlygą kiekvienai lentelės eilutei atskirai. Šios rūšies funkcijos dažnai vartojamos ir SELECT frazėje, kuomet rezultatas skaičiuojamas ne visoms eilutėms iš karto, kaip yra agregatinių funkcijų atveju, o kiekvienai eilutei atskirai. Paprastai, DBVS leidžia vartoti gana daug skaliarinių funkcijų. Paminėsime tik keletą iš jų: DAY(<data>) – diena (skaičius nuo 1 iki 31) datoje, MONTH(<data>) – mėnesis datoje, YEAR(<data>) – metai datoje, LENGTH(<simbolių eilutė>) – simbolių eilutės ilgis, SUBSTR(<simbolių eilutė>, <pradžia>, <ilgis>) – simbolių eilutės fragmentas ir pan.

Iš konstantų, kreipinių į funkcijas bei vardų, žyminčių DB objektus, panaudojant operacijas, galima sudaryti paprasčiausius reiškinius. SQL leidžia naudoti dvi unarines operacijas: + (pliusas), - (minusas), bei keletą binarinių: + (sudėtis), - (atimtis), * (daugyba), / (dalyba), || (apjungimas). Jei bent vienas iš pateiktų operacijų argumentų yra NULL, tai ir rezultatas yra NULL.

Su tekstiniais duomenimis galima atlikti tik dviejų simbolių eilučių nuoseklų apjungimą į vieną (||). Su skaičiais, kaip įprasta, galima atlikti visas aritmetines operacijas. Sudėtį ir atimtį galima atlikti ir su datos bei laiko duomenimis, tiksliau, turimą reikšmę galima padidinti, sumažinti bei rasti dviejų reikšmių skirtumą. Prie datos galima pridėti ar iš jos atimti tam tikrą sveiką skaičių dienų, mėnesių ar metų. Laiką galima pakeisti laiko vienetais. Atliekant tokias

operacijas, vienas iš argumentų – sveikas skaičius turi turėti dimensiją – raktinį žodį, nurodantį vienetus. Dimensijų pavyzdžiai: YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS. Reiškinio '2000.31.31' + 3 DAYS rezultatas yra '2000.01.03', o reiškinys *Projektai.Pradžia* + 3 MONTHS nusako datą, kuri bus praėjus 3 mėnesiams nuo *Projektai.Pradžia*. Atimant vieną iš kitos dvi datas ar du laikus, gaunamas laikotarpis. Galimas ir neigiamas laikotarpis.

Sudarant reiškinius, galima naudoti lenktinius skliaustus, taip keičiant operacijų atlikimo tvarką.

Reiškiniai gali būti predikatų argumentais (operandais). **Predikatas -** tai sąlyga lentelės eilutei ar eilučių grupei, kuri gali būti teisinga, neteisinga arba neapibrėžta. Visos predikate dalyvaujančios reikšmės tarpusavyje turi būti suderintos. Be to, simbolių eilutės, įeinančios į predikatus, dažnai, negali būti ilgesnės už tam tikrą konkrečiai DBVS būdingą simbolių kiekį, pvz., 4000. Paprasčiausi predikatai sudaromi panaudojant palyginimo operacijas. Palyginime visuomet dalyvauja dvi reikšmės. SQL leidžiamos tokios palyginimo operacijos: =, <, <=, >, >=, <>. Jei palyginimo operacijose vienas iš operandų yra NULL**,** tai predikato rezultatas yra neapibrėžtas. Palyginimo operacijose operandais gali būti ne tik skaitiniai duomenys, bet ir tekstiniai bei datos ir laiko duomenys.

Prie paprasčiausių predikatų priskiriama:

- x BETWEEN y AND z - rezultatas teisingas tik tuomet, kai x reikšmė yra tarp y ir z, t.y. kai x >= y ir x <= z;
- x NOT BETWEEN y AND z - rezultatas teisingas tik tuomet, kai x nėra tarp y ir z, t.y. kai x < y arba x > z;
- x IN (y1, y2,..., yn) - rezultatas teisingas tik tuomet, kai x reikšmė sutampa bent su viena iš reikšmių y1, y2,..., yn;
- x NOT IN (y1, y2,..., yn) - rezultatas teisingas tik tuomet, kai x nesutampa nei su viena iš reikšmių y1, y2, ..., yn;
- x LIKE y - rezultatas teisingas tik tuomet, kai simbolių eilutė x yra “panaši” į simbolių eilutę y. Simbolių eilutėje y galima panaudoti formato simbolius: %, kuris atitinka bet kokį kiekį, bet kokių simbolių, įskaitant tuščią (nulinio ilgio) eilutę; _(pabraukimo ženklas), kuris atitinka bet kurį vieną simbolį. Eilutėje y galimi ir bet kokie kiti simboliai, atitinkantys juos pačius;
- x NOT LIKE y - rezultatas yra teisingas tik tuomet, kai simbolių eilutė x nėra panaši į simbolių eilutę y. Eilutėje y gali būti pavartoti tie patys simboliai kaip ir predikate LIKE;
- x IS NULL - rezultatas yra teisingas tik tuomet, kai reiškinio x reikšmė yra NULL;
- x IS NOT NULL - rezultatas yra teisingas tik tuomet, kai reiškinio x reikšmė nėra NULL.

Iš predikatų, panaudojant logines operacijas: AND (loginis “ir”), OR (loginis “arba”) arba NOT (loginis “ne”), galima konstruoti paieškos sąlygas. Paieškos sąlygos rezultatas gali būti: “tiesa” (sąlyga patenkinta, teisinga), “netiesa” (sąlyga nepatenkinta, neteisinga) arba neapibrėžtas.

Loginių operacijų panaudojimas SQL kalboje atitinka priimtus matematinėje logikoje susitarimus, tačiau atsižvelgiama į tai, kad argumentų reikšmės gali būti neapibrėžtos. Pateiksime loginių operacijų AND ir OR teisingumo lentelę:

X	Y	X AND Y	X OR Y
Tiesa	Tiesa	Tiesa	Tiesa
Tiesa	Netiesa	Netiesa	Tiesa
Tiesa	Neapibrėžta	Neapibrėžta	Tiesa
Netiesa	Tiesa	Netiesa	Tiesa
Netiesa	Netiesa	Netiesa	Netiesa
Netiesa	Neapibrėžta	Netiesa	Neapibrėžta
Neapibrėžta	Tiesa	Neapibrėžta	Tiesa
Neapibrėžta	Netiesa	Netiesa	Neapibrėžta
Neapibrėžta	Neapibrėžta	Neapibrėžta	Neapibrėžta

Loginė operacija NOT apibėžiama taip: NOT (Tiesa) yra Netiesa, NOT (Netiesa) yra Tiesa ir NOT(Neapibėžta) yra Neapibėžta. Jei paieškos sąlygoje yra pavartoti skliausteliai, tai sąlyga, esanti tarp jų įvertinama pirmiau.

Pavyzdžiui, projektų, kurių pavadinime yra frazė "apskaita", jų svarba yra vidutinė ar didelė, ir kurie turėjo būti pabaigti iki šiandien, pavadinimus, vykdymo pradžios bei pabaigos datas galima sužinoti tokia užklausa:

```
SELECT Pavadinimas, Pradžia, Pradžia+Trukmė MONTHS AS Pabaiga
FROM Projektai
WHERE Pavadinimas LIKE '%apskaita%' AND
      Svarba IN ('Vidutinė', 'Didelė') AND
      Pradžia+Trukmė MONTHS < CURRENT DATE.
```

Informacija apie vykdytojus – informatikus arba vykdytojus, baigusius Vilniaus universitetą (nepriklausomai nuo kvalifikacijos) ir turinčius aukštesnę nei trečią kategoriją, bus pateikta įvykdžius užklausą:

```
SELECT * FROM Vykdytojai
WHERE Kvalifikacija = 'Informatikas' OR (Išsilavinimas = 'VU' AND Kategorija > 3).
```

2.5. Kelių lentelių jungimas

Viena iš svarbiausių SELECT sakinio galimybių yra dviejų ar daugiau lentelių jungimas (angl. *join*). Tarkime, mums reikia sužinoti pavardes vykdytojų, vykdančių projektą Nr. 1. Visa reikalinga informacija yra dviejose lentelėse: *Vykdytojai* ir *Vykdymas,* todėl užklausoje turi dalyvauti ne viena, o dvi lentelės. Šios lentelės turi bendrą dalį, kuri jas sieja - tai vykdytojo numeris. Pagal šį numerį lenteles galima logiškai jungti. SQL kalba šį uždavinį galima užrašyti taip:

```
SELECT Pavardė FROM Vykdytojai, Vykdymas
WHERE Vykdytojas = Nr AND Projektas = 1.
```

Šioje užklausoje, sąlyga *Vykdytojas = Nr* yra nusakytas loginis ryšys tarp dviejų lentelių.

Jungiant lenteles, kiekviena vienos lentelės eilutė siejama su kiekviena kitos lentelės eilute. Bendru atveju, jei užklausoje dviems lentelėms nėra jokios sąlygos ir vienoje iš lentelių yra n eilučių, o kitoje - m eilučių, tai rezultatą sudaro $m \times n$ eilučių. Tarkime, turime dvi lenteles:

LentelėA

A1	*B1*
a_1	b_1
a_2	b_1
a_2	b_2

LentelėB

A2	*B2*	*C2*
a_1	b_1	c_1
a_2	b_2	c_2

Tuomet užklausos

SELECT *A1, B1, A2, B2, C2* FROM *LentelėA, LentelėB*

rezultatas atrodys taip

A1	*B1*	*A2*	*B2*	*C2*
a_1	b_1	a_1	b_1	c_1
a_2	b_1	a_1	b_1	c_1
a_2	b_2	a_1	b_1	c_1
a_1	b_1	a_2	b_2	c_2
a_2	b_1	a_2	b_2	c_2
a_2	b_2	a_2	b_2	c_2

Paprastai užklausoje dviems lentelėms yra paieškos sąlyga, kuri logiškai susieja vienos lentelės vieną ar kelis stulpelius su kitos lentelės atitinkamais stulpeliais. Papildžius pastarąją užklausą paieškos sąlyga, kuri sujungtų abi lenteles, sulyginant dviejų vienos lentelės stulpelių (*A1*, *B1*) reikšmes su kitos lentelės dviejų stulpelių (*A2*, *B2*) reikšmėmis:

SELECT *A1, B1, C2* FROM *LentelėA, LentelėB*
WHERE *A1= A2* AND *B1 = B2*,

gausime tik dvi eilutės:

A1	*B1*	*C2*
a_1	b_1	c_1
a_2	b_2	c_2

Suprantama, stulpelių porų (*A1*, *A2*) ir (*B1*, *B2*) reikšmės turi būti tarpusavyje palyginamos, t.y., kaip minimum, jų tipai (reikšmių aibės) neturi konfliktuoti. Paskutinės užklausos SELECT frazėje, vietoje anksčiau pavartotų 5 stulpelių, turime tik 3, nes likusieji 2 stulpeliai naujų reikšmių neduoda.

Užrašysime užklausą apie tai, kokie vykdytojai kokius projektus vykdo ir kiek kiekvienam projektui skiria valandų. Jei rezultate norime matyti vykdytojo pavardę ir projekto pavadinimą, o ne jų numerius, tai mums reikia formuluoti užklausą net trims lentelėms:

SELECT *Pavardė, Pavadinimas, Valandos*
FROM *Vykdytojai, Projektai, Vykdymas*
WHERE *Projektas = Projektai.Nr* AND *Vykdytojas = Vykdytojai.Nr*.

Šioje užklausoje dvi lentelės (*Vykdytojai* ir *Projektai*) turi po vieną stulpelį tuo pačiu pavadinimu *Nr*, todėl šio stulpelio pavadinimą būtina patikslinti lentelės pavadinimu.

Lentelę galima sujungti ir su ja pačia. Sudarykime poras vykdytojų, turinčių tą pačią kvalifikaciją:

SELECT *A.Pavardė, B.Pavardė* FROM *Vykdytojai A, Vykdytojai B*
WHERE *A.Kvalifikacija = B. Kvalifikacija* AND *A.Nr < B.Nr*

Jungiant lentelę su ja pačia, galime laikyti, kad turime du tos pačios lentelės egzempliorius (kopijas). Tam, kad būtų galima kreiptis ir į abudu vienos lentelės egzempliorius, būtina kiekvienam egzemplioriui suteikti pavadinimą (*A* ir *B*). Kadangi abiejų lentelių (tiksliau dviejų vienos lentelės egzempliorių) stulpelių vardai sutampa, tai užklausoje negalime apsieiti be stulpelių vardų patikslinimo lentelės egzemplioriaus pavadinimu. Predikatas *A.Kvalifikacija = B. Kvalifikacija* atlieka dviejų lentelių loginio susiejimo funkciją. Sąlyga *A.Nr < B.Nr* pavartota tik tam, kad išvengti vykdytojo poros su juo pačiu ir porų pasikartojimo. Jei rezultate turime porą (x, y), tai pora (y, x) nereikalinga, nes ji nepateikia naujos informacijos. Pastebėsime, kad bazinio žodžio DISTINCT pavartojimas čia nepadeda, nes (x, y) ir (y, x) yra skirtingos eilutės, jei tik $x \neq y$.

2.6. Struktūrinės užklausos

Vienoje užklausoje galima pavartoti ir kitą užklausą. Todėl, žodis "struktūrinė" įeina į kalbos pavadinimą. Keli SELECT sakiniai, pavartoti vienoje užklausoje, visuomet yra griežtoje priklausomybėje. Ši priklausomybė yra hierarchinė - viena užklausa yra išorinė kitos atžvilgiu, o ši yra vidinė (dalinė, angl. *subquery*) išorinės atžvilgiu. Tačiau tai tik sintaksinė bei loginė priklausomybė, užklausų vykdymo tvarka nebūtinai yra tokia.

Apibrėšime pavardes vykdytojų, dalyvaujančių projekte Nr. 1, pasitelkdami vidinę užklausą:

SELECT *Pavardė* FROM *Vykdytojai*
WHERE *Nr* IN (SELECT *Vykdytojas* FROM *Vykdymas* WHERE *Projektas* = 1).

Šioje užklausoje pavartotas bendresnis predikato IN variantas, kai antrasis parametras yra nurodytas vidine užklausa. Vykdant tokią užklausą, DBVS, loginiu požiūriu (realiai gali būti ir kitaip), iš pradžių įvykdo vidinę (esančią sąlygoje) užklausą, suformuodama jos rezultatą – aibę reikšmių. Po to vykdoma išorinė užklausa, kiekvienai išorinės užklausos eilutei tikrinant, ar vykdytojo numeris priklauso aibei, gautai įvykdžius vidinę užklausą.

Pastarojoje užklausoje pavartoti du SELECT sakiniai nėra riba - jų gali būti ir daugiau. Pavyzdžiui, užklausą "pavardės vykdytojų, dalyvaujančių bent viename didelės svarbos projekte" galimai suformuluoti trimis SELECT sakiniais:

```
SELECT Pavardė FROM Vykdytojai
WHERE Nr IN
        (SELECT Vykdytojas FROM Vykdymas
         WHERE Projektas IN
                 (SELECT Nr FROM Projektai WHERE Svarba = 'Didelė')).
```

Priminsime, kad SQL yra deklaratyvi kalba, t.y. formuluojant užklausas reikia apibrėžti, kas norima gauti užklausos rezultate, nesirūpinant, kaip rezultatą gauti. Pastaroji užklausa tėra formalus ir detalus užrašas taip perfrazuotos užduoties: "pavardės vykdytojų, kurių numeriai yra tarp numerių vykdytojų, dalyvaujančių didelės svarbos projekte". Pastarojoje užklausoje visi SELECT sakiniai gali būti vykdomi nuosekliai, pradedant nuo pačios giliausios vidinės užklausos ir baigiant išorine.

Nors pastaroji užklausos formuluotė gana gerai atspindi žodinę užduoties formuluotę, didelis dalinių užklausų kiekis, nereiškia gerą užklausų sudarymo stilių. Dažnai analogiška užklausa, užrašyta vienu sakiniu SELECT, jungiant keletą lentelių, nors, iš pirmo žvilgsnio, ir nėra aiškesnė, tačiau yra trumpesnė, o galiausiai ir aiškesnė. Tačiau lentelių jungimas reikalauja daugiau įgūdžių, nei dalinių užklausų vartojimas. Pateiksime uždavinio apie vykdytojus, dalyvaujančius bent viename didelės svarbos projekte, sprendinį be dalinių užklausų:

```
SELECT DISTINCT Pavardė FROM Vykdytojai, Vykdymas, Projektai
WHERE Projektas = Projektai.Nr AND Vykdytojas = Vykdytojai.Nr  AND
        Svarba = 'Didelė'.
```

Tai, kad lentelių jungimas gali būti žymiai pranašesnis už dalines užklausas, galima įsitikinti palyginus pastarąją užklausos formuluotę su ankstesnio skyrelio užklausa, kurioje jungiamos tos pačios trys lentelės. Pastebėsime, kad abiejų užklausų paieškos sąlygos yra labai panašios, skiriasi tik papildoma sąlyga, lentelių jungimo sąlyga nesikeičia. Vieną kartą išsiaiškinus, kaip logiškai lentelės siejamos tarpusavyje, vėliau, daug užklausų galima parašyti tik papildant paieškos sąlygą.

Atkreipsime dėmesį, kad ta pati lentelė gali būti tiek vidinėje, tiek ir išorinėje užklausoje. Uždavinį "numeriai vykdytojų, kurie dalyvauja bent viename projekte, kuriame dalyvauja vykdytojas Nr. 1" galime išreikšti ir taip:

```
SELECT DISTINCT Vykdytojas FROM Vykdymas
WHERE Vykdytojas <> 1 AND
        Projektas IN (SELECT Projektas FROM Vykdymas WHERE Vykdytojas = 1).
```

Nors šioje užklausoje ta pati lentelė pavartota du kartus, tačiau nėra būtina patikslinti stulpelių vardų dėl panaudotų skliaustų ir galiojančios taisyklės: jei stulpelio vardas nėra tikslus, tai jis priklauso lentelei iš einamosios (vidinės ar išorinės) užklausos.

Anksčiau jau keletą kartų formuluotą uždavinį "pavardės vykdytojų, dalyvaujančių projekte Nr. 1" galima suformuluoti ir taip:

```
SELECT Pavardė FROM Vykdytojai
WHERE 1 IN (SELECT Projektas FROM Vykdymas WHERE Vykdytojas = Vykdytojai.Nr).
```

Šią užklausą neformaliai galima perfrazuoti taip: vykdytojai, dalyvaujantys projekte Nr.1, tai tokie vykdytojai, kuriems tarp visų projektų, kuriuose jie dalyvauja, yra projektas Nr. 1. Šioje, gana painioje, užklausoje, dalinė užklausa negali būti apskaičiuojama iš anksto, prieš pradedant vykdyti išorinę. Dalinė užklausa vykdoma kiekvienai pagrindinės užklausos eilutei. Tai - priklausomos dalinės užklausos pavyzdys. **Priklausoma užklausa** - tai užklausa, kurios vidinės užklausos rezultatas priklauso nuo išorinės užklausos rezultato. Priklausomoje užklausoje, vidinė užklausa yra parametrizuota, į ją įeina parametras (*Nr*), įgyjantis reikšmę išorinėje užklausoje. Vidinę užklausą tenka vykdyti kiekvienai to kintamojo reikšmei. Priklausomose užklausose negalima abi užklausas (vidinę ir išorinę) vykdyti nuosekliai. Priklausomos užklausos dažnai yra gana painios ir neefektyvios, todėl jų reikia vengti.

Pateiksime ankstesnio uždavinio "pavardės vykdytojų, dalyvaujančių projekte Nr. 1" dar vieną formuluotę, pavartojant predikatą - egzistavimo kvantorių:

```
SELECT Pavardė FROM Vykdytojai
WHERE EXISTS ( SELECT *  FROM  Vykdymas
                    WHERE Vykdytojas = Nr AND Projektas = 1 ) .
```

Šiame sakinyje išorinės užklausos paieškos sąlygoje, pavartotas predikatas, kurio sintaksė yra `EXISTS (SELECT * FROM…)`. Šio predikato reikšmė yra "tiesa" tuomet, ir tik tuomet, kai vidinės užklausos rezultatas yra netuščia aibė. Atkreipsime dėmesį į tai, kad tai nėra vienintelis predikatas, realizuojantis kvantoriaus sąvoką. Be egzistavimo kvantoriaus užklausose galima vartoti ir visuotinumo kvantorių. Pastaroji užklausa taip pat yra priklausoma. Vidinės, predikate pavartotos užklausos rezultatas priklauso nuo parametro *Vykdytojai.Nr*, kuris įgyja reikšmę išorinėje užklausoje.

2.7. Laikinos lentelės

Kadangi užklausos rezultatas yra lentelė (nors ir laikina), tai užklausą galima vartoti ir sakinio `SELECT` frazėje `FROM`. Pavyzdžiui, anksčiau suformuluotą užklausą "numeriai vykdytojų, kurie dalyvauja bent viename projekte, kuriame dalyvauja vykdytojas Nr. 1", galima užrašyti ir taip:

```
SELECT DISTINCT Vykdytojas
FROM Vykdymas,
        (SELECT Projektas FROM Vykdymas WHERE Vykdytojas = 1) AS Projektai1
WHERE Vykdymas.Vykdytojas <> 1 AND Projektai1.Projektas = Vykdymas.Projektas .
```

Šio SQL sakinio `FROM` frazėje pavartota užklausa, kuriai suteiktas vardas *Projektai1*. Tai - laikinos lentelės apibrėžimas. Taip apibėžta lentelė egzistuoja tik užklausos vykdymo metu. Į laikiną lentelę galima kreiptis tik užklausoje, kurioje ji yra apibrėžta. Į laikinos lentelės apibrėžimą galima žiūrėti kaip į vidinę užklausą, kuriai suteiktas vardas.

Laikinos lentelės apibrėžimas `FROM` frazėje, ypač kai tokių apibrėžimų yra keletas, gali padaryti užklausą sunkiai suprantama. Kad taip neatsitiktų, galima pasinaudoti specialia, sudėtingoms užklausoms formuluoti skirta konstrukcija `WITH`. Taip formuluojant užklausas elgiamasi panašiai, kaip ir apibrėžiant laikiną lentelę frazėje `FROM`. Iš pradžių apibėžiama

(“sukuriama”) laikina lentelė suteikiant jai vardą, o paskui rašomas SELECT sakinys, kuriame kreipiamasi į aukščiau apibrėžtą laikiną lentelę. Užklausa formuluojama nuosekliai: laikina lentelė yra apibrėžiama prieš (užklausos tekste) jos pavartojimą. Ankstesniąją užklausą galima suformuluoti taip:

```
WITH Projektai1 AS (SELECT Projektas FROM Vykdymas WHERE Vykdytojas = 1)
   SELECT DISTINCT Vykdytojas
   FROM Vykdymas, Projektai1
   WHERE Vykdymas.Vykdytojas <> 1 AND Projektai1.Projektas = Vykdymas.Projektas.
```

Viena fraze WITH galima apibėžti kelias laikinas lenteles, atskiriant jas tarpusavyje kableliais. Apibrėžiant laikiną lentelę, iš karto po jos pavadinimo, tarp skliaustų galima išvardinti ir jos stulpelių pavadinimus. Formuluojant sudėtingas užklausas, ši konstrukcija leidžia uždavinį spręsti, skaidant jį į keletą smulkesnių. Suformulavus užklausą tarpiniams rezultatams gauti, jai suteikiamas vardas ir toliau formuluojama aukštesnio lygio užklausa.

2.8. Duomenų grupavimas

Tarkime, kiekvienam projektui reikia apskaičiuoti, kiek visi vykdytojai bendrai skiria laiko jo vykdymui. Šiam uždaviniui spręsti iki šiol išdėstytų kalbinių priemonių nepakanka. Nesunku apskaičiuoti bendrą valandų kiekį kuriam nors vienam projektui, pvz. Nr. 1:

```
SELECT SUM(Valandos) FROM Vykdymas WHERE Projektas = 1.
```

Šiame SQL sakinyje paieškos sąlygoje yra fiksuotas vienas projektas. Visoms tenkinančioms sąlygą eilutėms yra pritaikyta funkcija valandoms sumuoti. Pradiniam mūsų uždaviniui spręsti, šią stulpelių funkciją reikia pritaikyti kiekvienai eilučių grupei, atitinkančiai visus skirtingus projektus. Eilučių grupavimą užklausose realizuoja speciali konstrukcija GROUP BY**,** kurią pavartojus uždavinio sprendinys atrodo taip:

```
SELECT Projektas, SUM(Valandos) AS Valandos FROM Vykdymas
GROUP BY  Projektas.
```

Visos lentelės *Vykdymas* eilutės grupuojamos taip, kad į kiekvieną grupę patenka eilutės su vienodomis stulpelio *Projektas* reikšmėmis. Po grupavimo, SELECT frazė yra taikoma kiekvienai grupei (o ne eilutei), suformuojant vieną užklausos rezultato eilutę. Vykdant pateiktą užklausą, DBVS kiekvienam projektui (kiekvienai grupei, kurią sudaro duomenys apie vieno projekto vykdymą) įtraukia į užklausos rezultatą projekto numerį ir sumą valandų, kurias skiria visi vykdytojai einamajam projektui vykdyti.

Pastebėsime, kad nežymiai pakeitus pastarąją užklausą:

```
SELECT  Projektas, Vykdytojas, SUM(Valandos) AS Valandos FROM Vykdymas
GROUP BY  Projektas
```

gauname sintaksiškai neteisingą užklausą. Sugrupavus lentelės *Vykdymas* eilutes taip, kad į vieną grupę patenka visos eilutės su ta pačia stulpelio *Projektas* reikšme, grupėje gali atsirasti kelios eilutės su skirtingomis stulpelio *Vykdytojas* reikšmėmis. Tuomet taps neaišku, kuri *Vykdytojo* reikšmė turėtų “atstovauti” grupę užklausos rezultate.

Kadangi, pritaikant SELECT frazę grupei eilučių, suformuojama tik viena rezultato eilutė, tai visi šios frazės reiškiniai turi būti vienareikšmiškai apskaičiuojami, t.y. turi įgyti vieną reikšmę visoje grupėje. Užklausos su grupavimu SELECT frazėje, su nedidelėmis išimtimis, tegali būti:

- stulpelis, paminėtas frazėje GROUP BY (grupavimo stulpelis);
- konstanta;
- reiškinys pagal konstantas ir grupavimo stulpelius;
- agregatinė (stulpelių) funkcija (SUM, MIN, MAX, COUNT, AVG ir kt.), kuri eilučių grupei pateikia vieną reikšmę.

Prieš GROUP BY frazę galima vartoti paieškos sąlygą, kuri tikrinama prieš eilučių grupavimą. Jei norime paieškos sąlygą pritaikyti ne kiekvienai lentelės eilutei, bet kiekvienai grupei, tai frazė GROUP BY vartojama kartu su fraze HAVING. Sąlyga grupėms yra tikrinama po grupavimo. Grupės "atstovas" į užklausos rezultatą įtraukiamas tik tuomet, kai grupė tenkina paieškos sąlygą, nurodytą frazėje HAVING.

Tokiu būdu, jei užklausoje su grupavimu yra pavartota frazė WHERE, tai, iš pradžių, kiekvienai eilutei tikrinama ši sąlyga ir visos eilutės, tenkinančios šią sąlygą, yra sugrupuojamos. Jei užklausoje yra pavartota ir frazė HAVING**,** tai pastarojoje frazėje nurodyta sąlyga yra tikrinama kiekvienai grupei. Pavyzdžiui, numeriai projektų, kuriuos vykdo daugiau negu vienas vykdytojas, gali būti sužinomi užklausa:

SELECT *Projektas* FROM *Vykdymas*
GROUP BY *Vykdytojas* HAVING COUNT(*) > 1.

Galima grupuoti ir pagal kelis stulpelius. Tuomet sakoma, kad grupuojama grupės viduje. Tarkime, kad mums reikia sužinoti, kiek yra vykdytojų pagal kiekvieną baigtą aukštąją mokyklą ir turimą kategoriją.Užklausai sudaryti, reikia apibrėžti grupavimą pagal du lentelės *Vykdytojai* stulpelius: *Išsilavinimas* ir *Kategorija*:

SELECT *Išsilavinimas*, *Kategorija*, COUNT(*) AS "*Vykdytojų Skaičius*"
FROM *Vykdytojai* GROUP BY *Išsilavinimas*, *Kategorija*
ORDER BY *Išsilavinimas*.

Šios užklausos rezultatas, išdėstytas pagal išsilavinimą leksikografine tvarka, atrodo taip:

Išsilavimas	*Kategorija*	*Vykdytojų Skaičius*
NULL	5	1
VDU	6	1
VU	2	1
VU	3	2

Tarkime, reikia sužinoti, kiek kiekvienas vykdytojas skiria valandų visiems projektams vykdyti. Be to, mus domina tik tie vykdytojai, kurie visų projektų vykdymui skiria daugiau nei 300 valandų. Jeigu mums pakanktų vykdytojo numerio, tai uždavinį galėtume spręsti panašiai kaip uždavinį, kuriame valandos buvo skaičiuojamos kiekvienam projektui. Jei mus domina vykdytojų pavardės, užklausą reikia formuluoti dviems lentelėms:

SELECT *Pavardė*, SUM(*Valandos*) AS "*Visos Valandos*"
FROM *Vykdytojai*, *Vykdymas* WHERE *Nr* = *Vykdytojas*
GROUP BY *Pavardė* HAVING SUM(*Valandos*) > 300
ORDER BY "*Visos Valandos*".

Sąlygoje grupei (frazėje HAVING) negalima vartoti SELECT frazėje apibrėžto stulpelio pavadinimo "*Visos Valandos"*. Tokio stulpelio pavadinimas grupėse negalioja. Jo tiesiog nėra grupėse. Stulpelio pavadinimas prasmingas tik užklausos rezultato kontekste, todėl jį galima vartoti apibrėžiant eilučių tvarką.

Jei užklausos rezultate norime pamatyti tiek vykdytojų pavardes, tiek ir jų numerius, tai, atsižvelgiant į jau pateiktus apribojimus SELECT frazei, tenka grupuoti pagal du stulpelius:

SELECT *Pavardė*, *Nr*, SUM(*Valandos*) AS "*Visos Valandos*"
FROM *Vykdytojai*, *Vykdymas* WHERE *Nr* = *Vykdytojas*
GROUP BY *Pavardė, Nr* HAVING SUM(*Valandos*) > 300
ORDER BY *Pavardė*.

Tai, kad kiekvienam vykdytojui (jo pavardei) mes gausime ne daugiau kaip po vieną eilutę, lems savybė (žinoma, jei tokia savybė tenkinama), kad nėra dviejų vykdytojų vienodomis pavardėmis. Jei keli vykdytojai galėtų turėti vienodas pavardes, bet skirtingus numerius, tai pastaroji užklausa išliktų teisinga, kai tuo tarpu ankstesnė, kurioje grupuojama tik pagal pavardes, logiškai būtų klaidinga (sintaksiškai išliktų teisinga), nes visi bendrapavardžiai būtų laikomi tuo pačiu asmeniu. Taigi, ir ankstesnėje užklausoje būtų buvę teisingiau (užklausos teisingumas nepriklausytų nuo apribojimų duomenims) pavartoti grupavimą pagal abudu vykdytojo tapatumo atributus: numerį ir pavardę.

2.9. Aibių operacijos

Kadangi užklausos rezultatas yra eilučių aibė, tai prasminga dviejų užklausų rezultatams taikyti aibių operacijas: sąjungą, sankirtą, skirtumą. Kiekvienai iš šių trijų aibių teorijos operacijų SQL kalboje yra po dvi savas operacijas. Taip yra todėl, kad užklausų rezultatuose, skirtingai negu aibių teorijoje, yra galimos vienodos eilutės. SQL kalboje yra šešios aibių operacijos: UNION, UNION ALL, INTERSECT, INTERSECT ALL, EXCEPT ir EXCEPT ALL. Apibrėšime šių operacijų rezultatus.

UNION ir UNION ALL - rezultatas yra dviejų užklausų rezultatų (*R1* ir *R2*) eilučių sąjunga. Operacijos rezultatas yra formuojamas iš visų *R1* ir *R2* eilučių, po to, jei nepavartotas bazinis žodis ALL, pašalinamos pasikartojančios eilutės, paliekant po vieną kiekvienos eilutės egzempliorių.

INTERSECT ir INTERSECT ALL - rezultatas yra dviejų užklausų rezultatų (*R1* ir *R2*) eilučių sankirta. Rezultatas formuojamas iš eilučių, kurios įeina tiek į *R1*, tiek ir į *R2*, po to, jei nepavartotas bazinis žodis ALL, pašalinamos pasikartojančios eilutės, paliekant po vieną kiekvienos eilutės egzempliorių.

EXCEPT ir EXCEPT ALL - rezultatas yra dviejų užklausų rezultatų (*R1* ir *R2*) eilučių skirtumas. Rezultatas formuojamas iš *R1* eilučių, kurių nėra rezultate *R2*, atliekant tai kiekvienam eilutės egzemplioriui atskirai, bet prieš tai, jei nepavartotas bazinis žodis ALL, tiek iš *R1,* tiek ir iš *R2* yra pašalinamos pasikartojančios eilutės, paliekant po vieną kiekvienos eilutės egzempliorių.

Tam, kad būtų galima atlikti aibių operacijas su užklausų rezultatais, būtina, kad stulpelių skaičius operacijų argumentuose (užklausų rezultatuose *R1* ir *R2*) sutaptų, o atitinkamų stulpelių tipai nekonfliktuotų. Jeigu, pavyzdžiui, vienos užklausos rezultate būtų vienas stulpelis, o kitos – du, tai taptų neaišku, kiek stulpelių turėtų būti tokių užklausų rezultatų sankirtoje. Todėl panašios situacijos SQL sintaksė neleidžia.

Visas SQL aibių operacijas paaiškinsime schematiškai. Tarkime, turime du vienodos struktūros užklausų rezultatus (lenteles) *R1* ir *R2*, ir iš viso yra tik penkios skirtingos eilutės, kurias pažymėsime numeriais nuo 1 iki 5. Tarkime, pirmosios užklausos rezultate *R1* yra dešimt eilučių: tris kartus pasikartoja eilutė, pažymėta Nr. 1, tris kartus – Nr. 2, vieną kartą – Nr. 3, du kartus – Nr. 4 ir vieną kartą – Nr. 5, o *R2* sudaro du kartus pasikartojanti eilutė Nr. 1, keturis kartus – Nr. 3 ir viena eilutė, pažymėta Nr. 4. Tuomet, visų šešių aibių operacijų rezultatus galima pavaizduoti tokia lentele:

R1	*R2*	UNION ALL	UNION	EXCEPT ALL	EXCEPT	INTERSECT ALL	INTERSECT
1	1	1	1	1	2	1	1
1	1	1	2	2	5	1	3
1	3	1	3	2		3	4
2	3	1	4	2		4	
2	3	1	5	4			
2	3	2		5			
3	4	2					
4		2					
4		3					
5		3					
		3					
		3					
		3					
		4					
		4					
		4					
		5					

Užklausos rezultatą, gautą aibių operacijos rezultate galima rūšiuoti. Tokiu atveju, eilučių tvarką apibrėžianti frazė yra rašoma po antrojo aibių operacijos operando. Pavyzdžiui, visus vykdytojus (jų numerius), kurie nedalyvauja nei viename projekte, galima sužinoti, įvykdžius užklausą su aibių skirtumo operacija EXCEPT:

```
SELECT Nr FROM Vykdytojai
EXCEPT
SELECT Vykdytojas FROM Vykdymas
ORDER BY 1.
```

Kadangi operacijoje EXCEPT dalyvaujančių užklausų rezultatuose stulpelių pavadinimai (*Nr* ir *Vykdytojas*) yra skirtingi, tai rūšiavimo stulpelis šioje užklausoje nurodytas eilės numeriu (1). Pastebėsime, kad šioje užklausoje operaciją EXCEPT pakeitę operacija EXCEPT ALL gausime tą patį rezultatą. Taip yra todėl, kad lentelėje *Vykdytojai* nėra vienodų stulpelio *Nr* reikšmių.

2.10. Sąlyginiai reiškiniai

Sudarykime projektų sąrašą, kuriame šalia projekto pavadinimo būtų pažymėta, ar jis trumpalaikis, ar ilgalaikis. Projektą laikysime trumpalaikiu, jei jo vykdymo trukmė neilgesnė už 6 mėn., kitaip projektą laikysime ilgalaikiu. Tokį sąrašą galima gauti užklausa:

```
SELECT Pavadinimas, 'Trumpalaikis' FROM Projektai WHERE Trukmė <= 6
UNION
SELECT Pavadinimas, 'Ilgalaikis'     FROM Projektai WHERE Trukmė > 6 .
```

Šioje užklausoje, priklausomai nuo sąlygos teisingumo, stulpelio reikšmė (bendruoju atveju - reiškinys) keičiama viena ar kita reikšme (kitu reiškiniu). Jei mes norėtume projektų vykdymo trukmę skirstyti į daugiau intervalų, tai, taip sprendžiant uždavinį, kiekvienam papildomam trukmės intervalui reikėtų pavartoti naują aibių sąjungos operaciją. Tokio uždavinio sprendimą galima supaprastinti pavartojus konstrukciją CASE, įgalinančią užrašyti sąlyginį reiškinį. Sąlyginio reiškinio sintaksę galima užrašyti taip:

```
CASE WHEN <paieškos sąlyga> THEN NULL | <reiškinys>
     {WHEN <paieškos sąlyga> THEN NULL | <reiškinys>}
     [ELSE NULL | <reiškinys>] END.
```

Sąlyginio reiškinio reikšmė priklauso nuo paieškos sąlygų teisingumo. Vykdant užklausą, kiekvienai eilutei paeiliui peržiūrimos visos paieškos sąlygos tikrinant jų teisingumą, kol randama teisinga sąlyga. Tuomet, viso reiškinio reikšme tampa reikšmė, esanti teisingos sąlygos dešinėje, už bazinio žodžio THEN. Jei tokios sąlygos nėra, viso sąlyginio reiškinio reikšmė apskaičiuojama pagal reiškinį, esantį dešinėje raktinio žodžio ELSE, jei tik toks yra, kitaip viso reiškinio reikšmė yra NULL.

Pastarąją užklausą galima užrašyti be aibių operacijos vienu sakiniu SELECT:

```
SELECT Pavadinimas,
        CASE WHEN Trukmė <= 6 THEN 'Trumpalaikis' ELSE 'Ilgalaikis' END
FROM Projektai.
```

Pateiksime dar vieną sąlyginio reiškinio vartojimo pavyzdį. Tarkime, kiekvienam vykdomam projektui mums reikia pateikti statistinius duomenis: bendrą visų projekto vykdytojų kiekį ir jų skiriamą laiką, bei tokius pat duomenis apie informatikų bei statistikų dalyvavimą projekte atskirai. Visą šią informaciją galima sužinoti trimis labai panašiomis užklausomis:

```
SELECT  Pavadinimas, COUNT(DISTINCT Vykdytojas) AS "Visi vykdytojai",
        SUM(Valandos) AS "Visų valandos"
FROM Projektai, Vykdymas WHERE Nr = Projektas GROUP BY Pavadinimas;

SELECT  Pavadinimas, COUNT(DISTINCT Vykdytojas) AS Informatikai,
        SUM(Valandos) AS "Informatikų valandos"
FROM Projektai, Vykdymas, Vykdytojai
WHERE Projektai.Nr = Projektas AND Vykdytojai.Nr = Vykdytojas AND
      Kvalifikacija = 'Informatikas'
GROUP BY Pavadinimas;

SELECT  Pavadinimas, COUNT(DISTINCT Vykdytojas) AS Statistikai,
        SUM(Valandos) AS "Statistikų valandos"
FROM Projektai, Vykdymas, Vykdytojai
WHERE Projektai.Nr = Projektas AND Vykdytojai.Nr = Vykdytojas AND
      Kvalifikacija = 'Statistikas'
GROUP BY Pavadinimas.
```

Netgi visas šias užklausas apjungus į vieną, pasinaudojant aibių sąjungos operacija, rezultatą nepatogu peržiūrėti, nes informacija apie vieną projektą bus trijose eilutėse. Informaciją apie projektą būtų patogiau peržiūrėti, jei visa ji būtų vienoje eilutėje. Tokį rezultatą gausime pavartoję sąlyginius reiškinius:

```
SELECT  Pavadinimas,
   COUNT(DISTINCT Vykdytojas) AS "Visi vykdytojai",
   SUM(Valandos) AS "Visų valandos",
   COUNT(DISTINCT CASE WHEN Kvalifikacija = 'Informatikas' THEN Vykdytojas)
         AS "Visi informatikai",
   SUM(CASE WHEN Kvalifikacija = 'Informatikas' THEN Valandos)
         AS "Informatikų valandos"
   COUNT(DISTINCT CASE WHEN Kvalifikacija = 'Statistikas' THEN Vykdytojas)
         AS "Visi statistikai",
   SUM(CASE WHEN Kvalifikacija = 'Statistikas' THEN Valandos)
         AS "Statistikų valandos"
FROM Projektai, Vykdymas, Vykdytojai
WHERE Projektai.Nr = Projektas AND Vykdytojai.Nr = Vykdytojas
GROUP BY Pavadinimas.
```

Dažnai vartojamiems sąlyginiams reiškiniams užrašyti yra numatyti sutrumpinimai - skaliarinės funkcijos: `NULLIF` ir `COALESCE`. Šios funkcijos apibrėžiamos taip:

Sąlyginis reiškinys	Ekvivalenti funkcija
`CASE WHEN` e1 = e2 `THEN NULL ELSE` e1 `END`	`NULLIF`(e1,e2)
`CASE WHEN` e1 `IS NOT NULL` `THEN` e1 `ELSE` e2 `END`	`COALESCE`(e1,e2)
`CASE WHEN` e1 `IS NOT NULL` `THEN` e1 `ELSE COALESCE`(e2,...,eN) `END`	`COALESCE`(e1,e2,...,eN)

Tarkime, mums reikia sužinoti, kokios bus vykdytojų kategorijos, jei visiems vykdytojams jas padidintume vienetu, o tiems vykdytojams, kuriems iki šiol kategorija nebuvo suteikta (tarkime, tokių gali būti), priskirsime pirmą kategoriją. Šiam uždaviniui spręsti pavartosime sutrumpintą sąlyginį reiškinį:

```
SELECT Pavardė, Kategorija AS "Esama kategorija",
       COALESCE(Kategorija, 0) + 1 AS "Naujoji kategorija"
FROM Vykdytojai.
```

2.11. Sisteminis katalogas

DBVS tam, kad galėtų sėkmingai vykdyti vartotojo užklausas ir valdyti duomenis, turi "prisiminti" gana didelį kiekį informacijos, susijusios su DB struktūra. Reliacinėje DB tokia informacija saugoma **sisteminiame kataloge** – sistemos lentelėse, kurias DBVS naudoja savo vidiniams poreikiams. Sisteminiame kataloge saugomas ir DB struktūros (lentelių, stulpelių ir kt.) aprašas.

Nors sisteminis katalogas skirtas sistemos vidiniams poreikiams, jame esanti informacija prieinama ir DBVS vartotojams. Reliacinėje DB yra gana detalus jos pačios aprašas, kurį vartotojas gali sužinoti užklausomis.

Sisteminio katalogo lentelės yra sukuriamos automatiškai sukuriant DB. Vartotojui nereikia rūpintis jų turiniu. Sistema pati rūpinasi, kad sisteminis katalogas atspindėtų einamąją DB būseną. Vartotojui griežtai uždrausta keisti sisteminio katalogo turinį – tai išimtinė DBVS privilegija. Vartotojas gali tik užklausti sisteminio katalogo duomenų.

Vykdydama SQL sakinius DBVS pastoviai kreipiasi į sisteminį katalogą. Pavyzdžiui, kad įvykdyti užklausą dviem lentelėms, DBVS turi:

- patikrinti, ar egzistuoja tos dvi lentelės;
- patikrinti, ar einamasis vartotojas, turi teisę kreiptis į abi lenteles;
- patikrinti, ar lentelėse yra stulpeliai, nurodyti užklausoje;
- nustatyti, kuriai lentelei priklauso stulpeliai, kurių vardai užklausoje nurodyti be lentelės pavadinimo;
- kiekvienam stulpeliui nustatyti duomenų tipą.

Visa ši informacija yra iš anksto apibrėžtose sisteminio katalogo lentelėse. Todėl DBVS informacijos paieškai sisteminiame kataloge gali naudoti ypač efektyvius metodus ir algoritmus.

Praktiškai visose komercinėse RDBVS, tarp kitų sisteminio katalogo lentelių, yra lentelės, aprašančios visas DB lenteles ir visus visų lentelių stulpelius. DBVS DB2 tai lentelės *SYSCAT.TABLES* ir *SYSCAT.COLUMNS*. Šios dvi lentelės, kaip ir visos kitos sisteminio katalogo lentelės, taip pat yra aprašytos jose pačiose. Todėl, užklausa:

```
SELECT * FROM SYSCAT.COLUMNS
```

galima sužinoti informaciją apie visų DB lentelių (tame tarpe ir pačios lentelės *SYSCAT.COLUMNS*) stulpelius. Detalesne užklausa:

```
SELECT NAME FROM SYSCAT.COLUMNS
WHERE TABSCHEMA = 'SYSCAT' AND TABNAME = 'TABLES'
```

sužinosime sisteminio katalogo lentelės, kurioje yra DB lentelių aprašai, stulpelių pavadinimus. Sisteminio katalogo lentelių stulpelių pavadinimai yra gana informatyvūs – iš stulpelio pavadinimo galima suprasti jo paskirtį. Visų sisteminio katalogo lentelių pavadinimus, išdėstytus leksikografine tvarka, galima sužinoti įvykdžius užklausą:

```
SELECT TABNAME FROM SYSCAT.TABLES
WHERE TABSCHEMA = 'SYSCAT'
ORDER BY 1.
```

Toks pat lentelių sąrašas bus gautas įvykdžius ir tokią užklausą:

```
SELECT DISTINCT TABNAME FROM Syscat.Columns
WHERE TABSCHEMA = 'SYSCAT'
ORDER BY 1.
```

Lentelės pavadinimą *Syscat.Columns* šioje užklausoje užrašėme pavartodami ir didžiąsias, ir mažąsias raides – tai neturi įtakos. Tačiau sąlygoje pavartotas schemos pavadinimas, būtinai turi būti užrašytas didžiosiomis raidėmis, nes tai tekstinė konstanta. Kadangi lentelėje *SYSCAT.COLUMNS* kiekvienai duomenų bazės lentelei gali būti po keletą eilučių (tiek, kiek lentelėje yra stulpelių), tai bazinis žodis `DISTINCT` užklausoje užtikrina, kad rezultate bus pateikti tik skirtingi lentelių pavadinimai.

Duomenų bazėje gali būti keletas lentelių tuo pačiu pavadinimu, bet skirtingose schemose. Pilną lentelės pavadinimą sudaro schemos ir lentelės pavadinimai kartu paėmus. Pilnus visų duomenų bazės lentelių pavadinimus galima sužinoti užklausa:

```
SELECT TABSCHEMA, TABNAME FROM SYSCAT.TABLES
ORDER BY 1, 2.
```

Detalesnį sisteminio katalogo aprašą, galima rasti konkrečios DBVS dokumentacijoje.