KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMATIKOS STUDIJŲ PROGRAMA

GEDIMINAS KULIAVAS

PHP DUOMENŲ BAZIŲ PLĖTINIŲ GALIMYBIŲ ANALIZĖ IR TYRIMAS

Magistro baigiamasis darbas

Darbo vadovas
doc. dr. S. Drąsutis

KAUNAS, 2013

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMATIKOS STUDIJŲ PROGRAMA

GEDIMINAS KULIAVAS

PHP DUOMENŲ BAZIŲ PLĖTINIŲ GALIMYBIŲ ANALIZĖ IR TYRIMAS

Magistro baigiamasis darbas

Darbo vadovas
doc. dr. S. Drąsutis

Recenzentas
prof. dr. R. Butleris

KAUNAS, 2013

TURINYS

SUMMARY 4
TERMINŲ IR SANTRUMPŲ ŽODYNAS 5
ĮVADAS 6
1. DUOMENŲ BAZIŲ VALDYMO SISTEMŲ ANALIZĖ 8
1.1. MySQL analizė 8
1.1.1. MySQL galimybės 9
1.1.2. MYSQL administravimo sąsajos 11
1.1.4. PHP ir MySQL susijungimo sąsajos 12
1.1.5. MySQL privalumai ir trūkumai 13
1.2. Oracle analizė 14
1.2.1. Oracle galimybės 14
1.2.2. Oracle administravimo sąsajos 15
1.2.3. PHP ir Oracle susijungimo sąsajos 16
1.2.4. Oracle privalumai ir trūkumai 16
1.3. PostgreSQL analizė 17
1.3.1. PostgreSQL galimybės 17
1.3.2. PostgreSQL administravimo sąsajos 17
1.3.3. PHP ir PostgreSQL susijungimo sąsajos 19
1.3.4. PostgreSQL privalumai ir trūkumai 19
1.4. MsSQL analizė 20
1.4.1. MsSQL galimybės 20
1.4.2. MsSQL administravimo sąsajos 21
1.4.3. PHP ir MsSQL susijungimo sąsajos 22
1.4.4. MsSQL privalumai ir trūkumai 22
1.5. DBVS analizės išvados 23
2. SISTEMOS PROJEKTAVIMAS 24
2.1. Funkciniai reikalavimai: 24
2.2. Sistemos projektas 24
2.2.1. Duomenų bazės struktūros diagrama 25
2.2.2. Sistemos komponentų diagrama 26
2.2.3. Panaudos atvejai 27
2.3. Sistemos realizacija 27
2.3.1. Eksperimentų vykdymo įrankis 27
2.3.2. DBVS paleidimas ir išjungimas 29
2.3.3. MySQL ir PHP sąsajos tvarkyklės suderinimas 29
2.3.4. Oracle ir PHP sąsajos tvarkyklės suderinimas 29
2.3.5. PostgreSQL ir PHP sąsajos tvarkyklės suderinimas 30

2.3.6. MSSQL ir PHP sąsajos tvarkyklės suderinimas 30
2.4. Apibendrinančios išvados 30
3. EKSPERIMENTINIS TYRIMAS IR REZULTATŲ ANALIZĖ 31
3.1. Eksperimentų tyrimų nustatymai ir specifikacija 31
3.2. Eksperimentas su duomenų išrinkimu 32
3.2.1 Eksperimento tikslas 32
3.2.2 Eksperimento atlikimo tvarka 32
3.2.3 Eksperimento rezultatai 33
3.2.4 Analizė susijusi su eksperimento tikslu ir gautų rezultatų apibendrinimas 45
3.3. Eksperimentas su duomenų įterpimu 47
3.3.1 Eksperimento tikslas 47
3.3.2 Eksperimento atlikimo tvarka 47
3.3.3 Eksperimento rezultatai 47
3.3.4 Analizė susijusi su eksperimento tikslu ir gautų rezultatų apibendrinimas 49
3.4. Apibendrinančios eksperimentų išvados 50
IŠVADOS 51
LITERATŪROS SĄRAŠAS 52
PRIEDAI 53
1. Priedas. Duomenų bazės lentelių atributai skirtingose DBVS 53
2. Priedas. Užklausos 55
4. Priedas. Vidinių procedūrų kodai 56

ANALYSIS AND FEASIBILITY STUDY OF PHP DATABASES EXTENSIONS PERFORMANCE

SUMMARY

Nowadays, while working with large amount of information the most important moment is the time of the beginning of the required material search. However, many emerging systems are usually developed as if it could work with small amounts of information which increases during a long time of maintenance. The response time to queries tend to slow down and the scientists are looking for the modern ways to optimize the systems due to that. In order to optimize all the internet applications the modern search is being carried out to find the best receipt and the optimal technological systems are implemented to enhance the systems productivity. The study of the RDBMS is used to find an answer to the question of optimization. This work aims at analyzing the comparison of the principles of the concerned database management systems operation and the evaluation of the speed of the work with data. Trying to fulfill the main aim of the research three tasks are set, which are successfully accomplished and let to make the revealing conclusion. The conclusion has been made that in Windows server Apache the best PHP database extensions is such as PDO mysql. Also, it is found that for the MSSQL database is suitable to use PDO mssql extension to PostgreSQL - PDO pgsql extension, and with Oracle - oci8 extension.

TERMINŲ IR SANTRUMPŲ ŽODYNAS

Terminas	Paaiškinimas
DBVS	Duomenų bazių valdymo sistema.
Vidinė procedūra (SP)	Duomenų bazės vidinė procedūra (angl. *Stored procedure*) atliekanti numatytas funkcijas.
Trigeris	Trigeris tai aibė veiksmų, kurie automatiškai atliekami vykdant: lentelės eilučių šalinimą, keitimą, naujų eilučių įterpimą.
PHP	Plačiai naudojama internetinių sistemų programavimo kalba.
ACID	Atomiškumas, Neprieštaringumas, Izoliavimas, Ilgalaikiškumas (Atomicity, Consistency, Isolation, Durability) savybių rinkinys kuris garantuoja duomenų bazės transakcijų patikimumą.
Transakcija	Loginis duomenų bazių vienetas, veiksmų duomenų bazėje seka.
BPS	Informacijos perdavimo greičio matavimo sutrumpinimas, matuojamas baitais per sekundę.
TPS	Duomenų bazės operacijų kiekio matavimo sutrumpinimas, matuojamas transakcijomis per sekundę.
MySQL	MySQL duomenų bazių valdymo sistema
MsSQL	Microsoft SQL Server duomenų bazių valdymo sistema
PostgreSQL	PostgreSQL duomenų bazių valdymo sistema
Oracle	Oracle duomenų bazių valdymo sistema
PDO	PHP biblioteka skirta susieti skirtingų duomenų bazių sąsajas naudojant tą patį programinį kodą.

ĮVADAS

Didėjant informacijos kiekiui šių dienų sistemose kyla problema dėl informacijos įrašymo greičio, bei informacijos gavimo iš saugojimo saugyklų, reakcijos laiko. Dirbant su dideliais informacijos kiekiais svarbiausia yra reikiamos medžiagos gavimo laikas nuo paieškos pradžios. Visgi, daugelis kuriamų sistemų dažniausiai kuriamos darbui su nedideliais informacijos kiekiais, kuri išauga eksploatuojant sistemą ilgą laiką. Dėl minėtos priežasties, reakcijos laikas į užklausas lėtėja ir kūrėjai ieško būdų kaip optimizuoti sistemą.

Temos aktualumas

Duomenų bazių optimizacija dažniausiai pradedama taisant klaidas užklausose. Užklausos būna paskirstytos per daugelį skirtingų bylų ir sunku rasti reikiamas užklausas. Todėl reiktų visas sudėtingesnes užklausas saugoti duomenų bazių valdymo sistemos (toliau - DBVS) vidinėse procedūrose (angl. *stored procedure*). Šiomis dienomis siekiant optimizuoti internetinių projektų veiklą ieškomi įvairūs technologiniai sprendimai ir diegiamos technologijos sistemos jų našumui didinti.

Šiame tiriamajame darbe bus apžvelgiamos skirtingos duomenų bazių valdymo sistemos, kurios pasiekiamos naudojantis PHP kalbos sąsaja darbui su jomis. Apžvelgiamos bus šios DBVS: MySQL, Oracle, PostgreSQL ir MsSQL. Pagrindinis dėmesys bus skiriamas duomenų įrašymo, bei nuskaitymo efektyvumui nustatyti pasinaudojant paprastomis užklausomis ir vidinėmis duomenų bazių procedūromis. Ištyrus dažniausiai naudojamas DBVS ir PHP duomenų bazių plėtinius būtų rastas atsakymas į optimizavimo klausimą.

Darbo objektas

PHP programavimo kalbos plėtinių sąveika su skirtingomis duomenų bazių valdymo sistemomis.

Darbo tikslas - palyginti nagrinėjamų duomenų bazių valdymo sistemų veikimo principus ir apskaičiuoti darbo su duomenimis spartą. Tiriamasis sprendimas leistų įvertinti tinkamiausią DBVS plėtinį darbui su PHP.

Darbo tikslo įgyvendinimui pasiekti iškeliami tokie **uždaviniai**:

- Išanalizuoti PHP plėtinius skirtus bendrauti su MySQL, Oracle, PostgreSQL ir MsSQL duomenų bazėmis.
- Suprojektuoti tiriamų duomenų bazių lenteles, užpildyti jas vienodais duomenimis ir sukurti vidines procedūras duomenų išrinkimui.
- Atlikti eksperimentus susijusius su duomenų išrinkimu ir įterpimu su MySQL, Oracle, PostgreSQL ir MsSQL duomenų bazėmis naudojant PHP plėtinius.

Tyrimo dokumento struktūra:

- Pirmoje dalyje yra nagrinėjamos duomenų bazių valdymo sistemos, išanalizuotos jų galimybės, bei sąsajos darbui su PHP, nagrinėjamos administravimo sąsajos su duomenų bazių valdymo sistemomis bei atlikti tarpusavio palyginimai.
- Antroje dalyje yra nagrinėjama kuriamos analizavimo sistemos projekto savybės bei apibrėžiami funkciniai reikalavimai. Aptariama iš ko sudarytos testavimui parengtos duomenų bazės. Aprašomi būdai kaip sukonfigūruoti atitinkamų duomenų bazių valdymo sistemų ir PHP plėtinių sąryšį su Apache serveriu.
- Trečioje dalyje yra aprašomas eksperimentinis tyrimas ir jo rezultatai bei susisteminta informacija apie atliktą tyrimą.

1. DUOMENŲ BAZIŲ VALDYMO SISTEMŲ ANALIZĖ

Šiame skyriuje bus nagrinėjama MySQL, Oracle, PostgreSQL ir MSSQL duomenų bazių valdymo sistemų galimybės ir atliekama DBVS sąsajų su PHP analizė, DBVS administravimo įrankių analizė, bei DBVS trūkumų ir privalumų analizė.

Tyrimui atlikti pasirinkta PHP programavimo kalba. Ši nemokama, sparčiai tobulinama ir atnaujinama kalba, naudojama iki 79,5% internetinių paslaugų sistemų [1]. Dėl didelės šios kalbos populiarumo prisideda lengvas sintaksės įsisavinimas, duomenų tipų laisvas pasirinkimas, didelė programuotojų bendruomenė, bei gera kalbos dokumentacija. PHP kalba pirmaujanti tarp internetinių programavimo kalbų smulkiuose, bei vidutinio sunkumo projektuose [2].

Tyrimo analizei atlikti pasirinktos duomenų bazių valdymo sistemos:

- MySQL – atviro kodo, reliacinių duomenų bazių valdymo sistema. Ši sistema buvo pasirinkta tyrimui, nes tai dažniausiai pasitaikanti, internetinių projektų, parašytų PHP kalba, duomenų bazių valdymo sistema. MySQL yra nemokama, bet tik bendruomeninė (angl. *community*) versija, kurios užtenka ir mažiems ir vidutiniams sistemų projektų poreikiams užtikrinti.
- Oracle – objektinė–reliacinių duomenų bazių valdymo sistema. Oracle yra labai plačiai naudojama bankinėse, finansinėse ir mokslinėse sistemose duomenims saugoti, apdoroti ir analizuoti.
- PostgreSQL – atviro kodo, reliacinių duomenų bazių valdymo sistema. Sistema veikia daugelyje operacinių sistemų. Be standartinių SQL galimybių, palaiko indeksavimą, vidines procedūras ir trigerius įvairiomis kalbomis (įskaitant vidinę PL/pgSQL). PostgreSQL buvo pasirinkta, nes tai gana sparčiai populiarėjanti DBVS ir dideliems projektams naudojama sistema.
- MsSQL – tai populiariausia Windows tipo serverių reliacinė duomenų bazių valdymo sistema. Verslo sprendimams yra apmokestinta.

1.1. MySQL analizė

MySQL – reliacinė duomenų valdymo sistema, kuria šio metu domisi pasaulio programuotojai per pastaruosius metus [3]. Jos populiarumą lemia tai, kad ji yra nemokama ir ganėtinai lengvai išmokstama naudotis, dėl didelių išteklių mokomosios medžiagos, bei didelės MySQL bendruomenės.

1995 metų kovo 23 dieną buvo išleista pirmoji MySQL versija [4] ir per tą laiką buvo išleistos penkios pagrindinės versijos. Jau 2009 metų kovo 22 dieną buvo išleista alfa šešta versija, tačiau dar 2013 metų pradžioje oficialioje MySQL svetainėje siūlomos versijos yra

5.5.29 arba 5.6. Iki penktosios versijos MySQL pagrindiniai trūkumai buvo tai, kad ji neturėjo nei kursorių, nei vidinių procedūrų, nei trigerių, nei vaizdų (angl. *views*), bei transakcijų. Po penktosios versijos išleidimo MySQL tapo tikras konkurentas tokiems duomenų bazių gigantams kaip Oracle ir Microsoft. Šį faktą įžvelgę 2008 metais MySQL AB kompaniją nusipirko Sun Microsystems [5], tačiau nepraėjus porai metų Oracle kompanija nusiperka pačius Sun Microsystems. Nors ir Oracle nusipirko *MySQL*, tačiau paliko nemokamam naudojimui tik MySQL Community leidimą, bei apmokestino komerciniam naudojimui skirtui leidinius.

1.1.1. MySQL galimybės

MySQL nuo 5.5 versijos ir naujesnės siūlo šiuos duomenų bazės valdymo ypatumus [4]:

- Daugiaplatformė – MySQL palaiko visas pagrindines operacines sistemas, kaip Windows, Unix ir Mac.
- Vidinės procedūros – tai duomenų bazės programavimui skirtos procedūros kurios atlieka numatytus veiksmus ir palengvina duomenų įvedimą ir išvedimą, bei duomenų apsaugą nuo netinkamų užklausų.
- Trigeriai – tai užklausa ar jų rinkiniai, kurie įvykdomi prieš arba po įterpimo, atnaujinimo ar ištrynimo veiksmų. Jų pagalba galima įtvirtinti sistemos logiką, sekti vartotojo veiksmus, pranešti apie klaidas kurios įvyksta dėl klaidingų duomenų ar sistemos sutrikimo.
- Kursoriai (angl. *Cursors*) – naudojami vidinėse procedūrose, tačiau jie skirti tik skaitymui. Gali būti skaitomi tik viena kryptimi ir einama per visas kursoriaus eilutes, nepraleidžiant nei vienos.
- Atnaujinami lentelės vaizdai (angl. *Updatable views*) – tai užklausos, kurios sukuria vaizdą tik iš būtinų lentelės ar lentelių laukų. Taip atskleidžiama mažiau svarbių laukų, kaip asmeniniai duomenys ar mokėjimo kortelių informacija kurios nereikia konkrečioje situacijoje, ir leidžiamas didesnis saugumas dirbant su duomenimis.
- Informacijos schema (angl. *Information schema*) – tai tik skaitymui skirta duomenų bazė, kurioje saugoma visa informacija apie visų duomenų bazių esančių MySQL sandarą. Jos pagalba greitai galima gauti informaciją apie konkrečią lentelę ar apie konkretų lentelės lauką, bei kitą informaciją.
- Strict mode – palaiko teisingų ar neteisingų duomenų įvedimą kaip kokia informacija yra nežinoma. Pavyzdžiui į duomenų bazę reikia įvesti datą „2013-02-30“. Tokios datos kalendoriuje nėra, todėl pagal nutylėjimą MySQL sugeneruos įspėjimo tipo klaidą ir įves į duomenų bazę „0000-00-00“. Šį nustatymą galima pakeist įjungus

STRICT_ALL_TABLES arba STRICT_TRANS_TABLES. Įjungus nustatymus ir bandant įvesti blogus duomenis bus sugeneruotas klaidos kodas ir įterpimo užklausa nutraukiama.

- Nepriklausomi duomenų bazės lentelių saugojimo variklių tipai – leidžia saugoti ir naudoti duomenis pagal savo poreikį. Jai projektui reikia paprastos duomenų bazės tik skaityti kokiai nors informacijai kuri retai kinta, tai pilnai tokiam projektui užteks MyISAM lentelės saugojimo variklio. O jai projektui reikia atlikinėti skaičiavimus su piniginiais reikalais tai be transakcijinės duomenų bazės saugojimo variklio lentelės nereiktų pradėt projekto. Šiam projektui reiktų naudot InnoDB varikliu paremtas lenteles.

Dažniausiai duomenų bazių projektuotojų naudojami MySQL duomenų bazės lentelių saugojimo variklių (angl. - *Storage Engines*) tipai yra šie [4]:

- MyISAM – tai pagrindinė pagal nutylėjimą iki 5.5 versijos buvęs lentelių saugojimo variklis, kuris nepalaiko transakcijų, išorinių raktų, bei ganėtinai lėtas įterpimo operacijoje. Tačiau šis variklis greitas duomenų išgavimui iš lentelių ir kol kas vienintelis palaiko pilną teksto paieškos indeksavimą.
- InnoDB – nuo 5.5 MySQL versijos šis duomenų lentelės saugojimo variklis tapo numatytuoju. Šis variklis palaiko transakcijas, palaiko išorinius lentelių raktus ir palaiko ACID standartą. Šis variklis (FTS) nepalaiko, bet aktyviai tobulinamas palaikyti šią galimybę [7].

1 Lentelė. MyISAM ir InnoDB saugojimo variklių palyginimas

	MyISAM	**InnoDB**
Didžiausias lentelės dydis	256TB	64TB
Transakcijos	Nėra	Yra
Duomenų kešavimas	Nėra	Yra
Užklausų kešavimo palaikymas	Yra	Yra
Išorinio rakto palaikymas	Nėra	Yra
FTS palaikymas	Yra	Nėra(iki 5.6.4 versijos)
Indeksų kešavimas	Yra	Yra
Lentelės užrakinimas	Visa lentelė	Viena eilutė

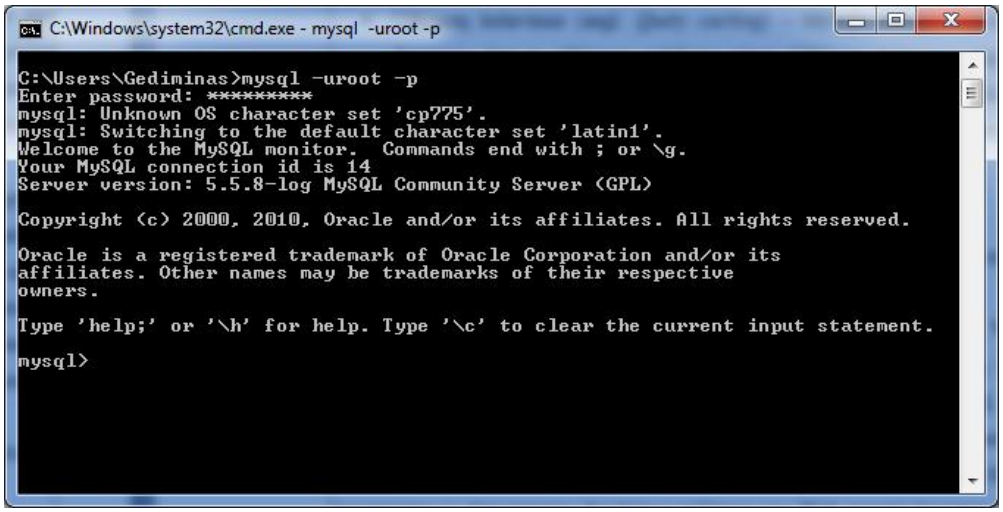
Taip pat yra ir daugelis kitų duomenų bazės lentelių saugojimo variklių kaip: MEMORY(HEAP), CSV, ARCHIVE, BLACKHOLE.

- Užklausų kešavimas (angl. *Query caching*) – MySQL kešuoja tik užklausos gautus duomenis, bet ne užklausos vykdymo planą. Užklausų kešavimas taip pat įgalina tų pačių užklausų iš skirtingų sesijų kešavimą. Jai dvi vienodos užklausos parašytos

viena iš didžiųjų raidžių, kita iš mažųjų, jos traktuojamos kaip skirtingos ir kešuojamos atskirai.

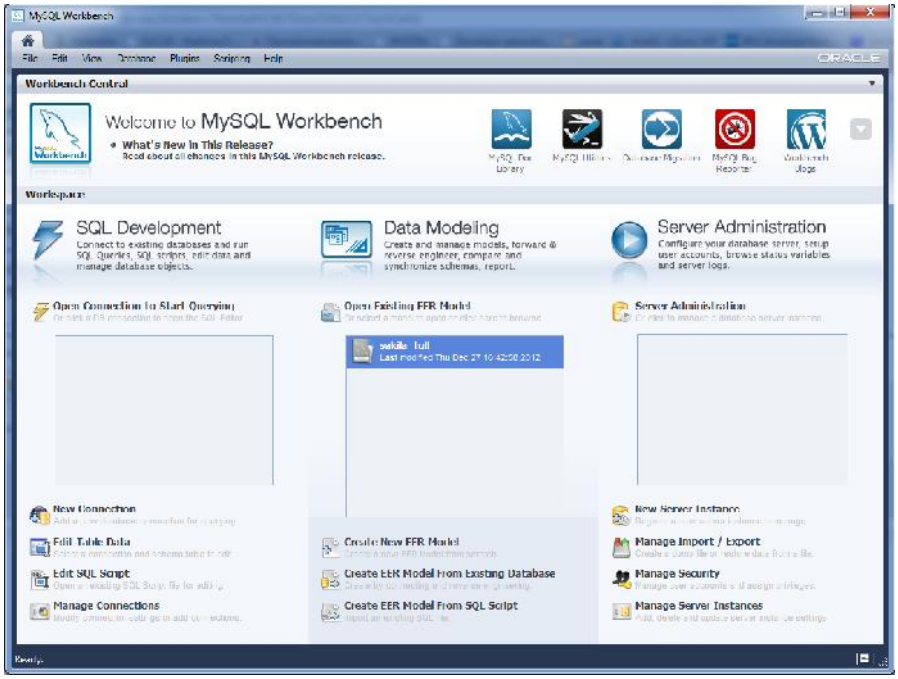
1.1.2. MYSQL administravimo sąsajos

Įdiegus MySQL darbui su duomenų baze naudojama tik komandinės eilutės (žr. 1 pav.) komanda: „mysql –u vartotojo_vardas –p slaptažodis“.



1 pav. Komandinė eilutė

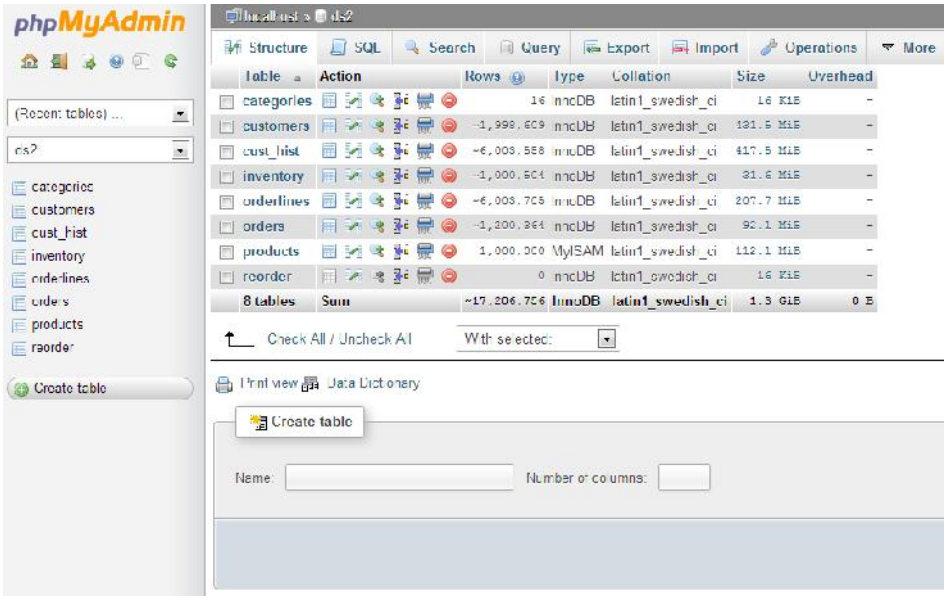
Galima parsisiųsti programą sukurta MySQL programuotojų MySQL Workbench (žr. 2 pav.). Tai grafinė sąsaja modeliuoti, administruoti ir vykdyti užklausas duomenų bazėje. Tačiau vienintelis apribojimas tai, kad programa skirta palaikyti 5.1 ir didesnes MySQL versijas.



2 pav. MySQL Workbench vaizdas

Taip pat galima tiesiogiai per naršyklę administruoti MySQL duomenų bazę su phpMyAdmin (žr. 3 pav.). Ši programa įgalina dauguma pagrindinių administravimo

funkcijų. Tačiau importavimo ir eksportavimo funkcijos priklauso nuo Apache serverio kuriame jinai yra įrašyta. Norint importuoti didelius kiekius informacijos reikia ją suskaldyti į mažesnius failus, arba serverio nustatymuose pakeisti failo dydžio limitą.



3 pav. phpMyAdmin vaizdas

1.1.4. PHP ir MySQL susijungimo sąsajos

PHP ir MySQL duomenų bazės susijungimo sąsajos yra šios [6]:

1. MYSQL (angl. - Original *MySQL API*) - dažniausiai naudojama prisijungimo sąsaja pradedančiųjų programuotojų. Naujiems projektams kurti nepatariama [8], nes nepalaiko naujausių MySQL galimybių, kaip vidinių procedūrų naudojančių kursorius, paruoštų užklausų (angl. - *Prepared Statemetns*), bei kelių užklausų vienu metu (angl. - *Multiple Statements*). Šios sąsajos tobulinimo darbai yra nutraukti dėl to tinkama naudot paprastoms sistemoms, kurios nereikalauja naujausių MySQL duomenų bazės galimybių.

2. MYSQLI (angl. - *MySQL improved Extension*) - objektiniam programavimui skirta sąsaja prisijungimui prie MySQL duomenų bazės. Šią sąsają patariama naudoti kuriant naujas sistemas. Sąsaja palaiko vidines procedūras, paruoštas užklausas, gali atlikti kelias užklausas vienu metu, transakcijas ir visą kitą MySQL 4.1 ir tolimesnės versijos funkcionalumą. Visos šios galimybės įgalinamos tik programuojant ne mažesne kaip PHP 5.0 versija.

3. PDO (angl. *PHP Data Objetcs*) MySQL - sąsaja palaikanti daugelį DBVS. Programos kodas nepriklauso nuo duomenų bazės sistemos. Užtenka pakeisti duomenų bazės prisijungimo duomenis ir nurodyti kurią DBVS sistemą naudoti. Sąsaja palaikanti beveik visas MYSQLI galimybes.

Nuo PHP 5.3.0 versijos sukurta mysqlnd (angl. - *MySQL Native Driver*) tvarkyklė, kurią suprogramavo PHP bendruomenės nariai ir yra labiau optimizuota darbui su PHP. Pirminė tvarkyklė buvo kurta Oracle korporacijos ir buvo saugoma autorinių teisių. Mysqlnd

yra optimizuota darbui su PHP kalba, ne kaip pirminė tvarkyklė kuri buvo daugiau skirta darbui su C programavimo kalbos ypatumais.

Mysqlnd tvarkyklė nuo PHP 5.4.0 versijos jau įdiegiama kaip numatytoji MySQL tvarkyklė mysql sąsajai. Mysql PHP plėtinių palyginimas aprašomas 2 lentelėje.

2 Lentelė. MySQL PHP plėtinių palyginimas

	mysqli	**PDO MySQL**	**mysql**
Įdiegta nuo PHP versijos	5.0	5.1	2.0
Ar plėtinys yra PHP 5 ir didesnėje versijoje?	Taip	Taip	Taip
Tobulinimo statusas	Tobulinama	Tobulinama	Tik palaikymas
Gyvavimo ciklas	Aktyvus	Aktyvus	Pasenęs
Ar rekomenduojamas naujiems projektams?	Taip	Taip	Ne
Objektinio programavimo sąsaja	Taip	Taip	Ne
Procedūrinė sąsaja	Taip	Ne	Taip
Ar palaiko asinchronines neblokuojamas užklausas?	Taip	Ne	Ne
Nenutraukiamo, pastovaus prisijungimo galimybė	Taip	Taip	Taip
Ar sąsaja palaiko skirtingų simbolių rinkinius?	Taip	Taip	Taip
Ar sąsaja palaiko serverio pusės paruoštas užklausas?	Taip	Taip	Taip
Ar sąsaja palaiko kliento pusės paruoštas užklausas?	Ne	Taip	Ne
Ar sąsaja palaiko vidines procedūras su kursoriais?	Taip	Taip	Ne
Ar sąsaja atlieka kelias užklausas iškarto?	Taip	Taip	Ne
Ar sąsaja palaiko transakcijas?	Taip	Taip	Ne
Ar palaiko MySQL 5.1+ funkcionalumą?	Taip	Didžiaja dalį	Ne

1.1.5. MySQL privalumai ir trūkumai

Privalumai iš analizės metu surinktų duomenų:

- Nemokama;
- Didelė sparta pasiekiama nenaudojant užklausų analizavimo įrankio;
- Geriausiai tinka smulkiems ir vidutiniams sistemų sprendimams įgyvendinti;
- Nuolat tobulinama sąsaja.

Trūkumai iš analizės metu surinktų duomenų:

- Ne visos sąsajos ar lentelių saugojimo varikliai palaiko transakcijas, vidines užklausas ar trigerius.
- Vidinės užklausos sukūrimas ir paleidimas reikalauja administratoriaus teisių.
- Duomenų bazės dydžio apribojimai.
- MySQL neatitinka dabartinių SQL standartų, nes vis dar naudojamasi SQL99 [10].
- Trigeriai gali atlikti tiktai vieną veiksmą į užklausą ir trigerių negalima naudoti vaizdų lentelėse.
- MySQL sparta tiesiogiai priklauso nuo kietojo disko tipo. Disko įrašymo vėlinimai daro įtaką duomenų įterpimui į duomenų bazę.

1.2. Oracle analizė

Oracle – tai objektinė - reliacinė duomenų bazės valdymo sistema, kuri turi labai didelę rinkos dalį didelių kompanijų, bankų, bei valstybinėse organizacijose, turinti jau virš 30 metų patirtį apdorojant, bei saugant duomenis.

Oracle duomenų bazės yra pati seniausia iš darbe nagrinėjamų DBVS sistemų. 1978 metais buvo išleista pati pirmoji komercinė versija. Oracle kompanija savo produkto liniją išskaidė į šios leidimus:

- Enterprise – brangiausia DBVS versija skirta didelėms įmonėms ar valstybinėms duomenų bazėm kurti. Jokių funkcinių apribojimų nėra.
- Standard - mokama, tačiau žymiai pigesnė už Enterprise ir daugiausiai gali būti naudojama serveriuose su ne daugiau kaip keturiais procesoriais.
- Express – nemokama, tačiau didžiausias duomenų bazės dydis vartotojui yra tik 11Gb, ir DBVS naudoja tik 1Gb atminties. Puikiai tinkanti akademiniams reikalams ir projektuose kuriuose nereikalingas didelis duomenų kiekis ar duomenų atnaujinimas.

1.2.1. Oracle galimybės

Oracle 11g ir naujesnės versijos ypatumai [11]:

- Daugiaplatformė – Oracle palaiko visas pagrindines operacines sistemas, kaip Windows, Unix ir Mac.
- Stantardizuotos užklausos – palaiko SQL, PL/SQL, ANSI/SQL, bei pilnai atitinka SQL2003 standartą [10].
- Vidinės procedūros – galimybė rašyti ne tik SQL užklausomis, bet ir Java bei PL/SQL kalbomis. PL/SQL vidinės procedūros eilučių skaičius yra apribojamas iki 3000.
- Transakcijos – gali būti pavaidinamos atskirais vardais, turėti užbaigimo komentarus, bei turėti apsaugos taškus (*angl.* Savepoints), kurie apsaugo jei įvyksta klaida visos užklausos neįvykdymo.
- Užklausų bibliotekos kešavimas – unikalių bendrai naudojamų užklausų vykdymo plano kešavimas pagreitina darbą kai skirtingos programos naudojasi jomis bendrai.
- Lentelių saugojimas – Oracle palaiko iki 128TB lentelės dydžio failą (*angl.* namespaces), tačiau juos galima apjungti į vieną duomenų bazę taip išplečiant pačios duomenų bazės dydį.
- Trigeriai – prieš ir po įterpimo, atnaujinimo ar ištrynimo atveju gali apdoroti net 32 paeiliui vykdomus trigerius.
- Lentelės dydis – ribojamas iki 1000 stulpelių iš kurių 30 gali būti suindeksuoti.

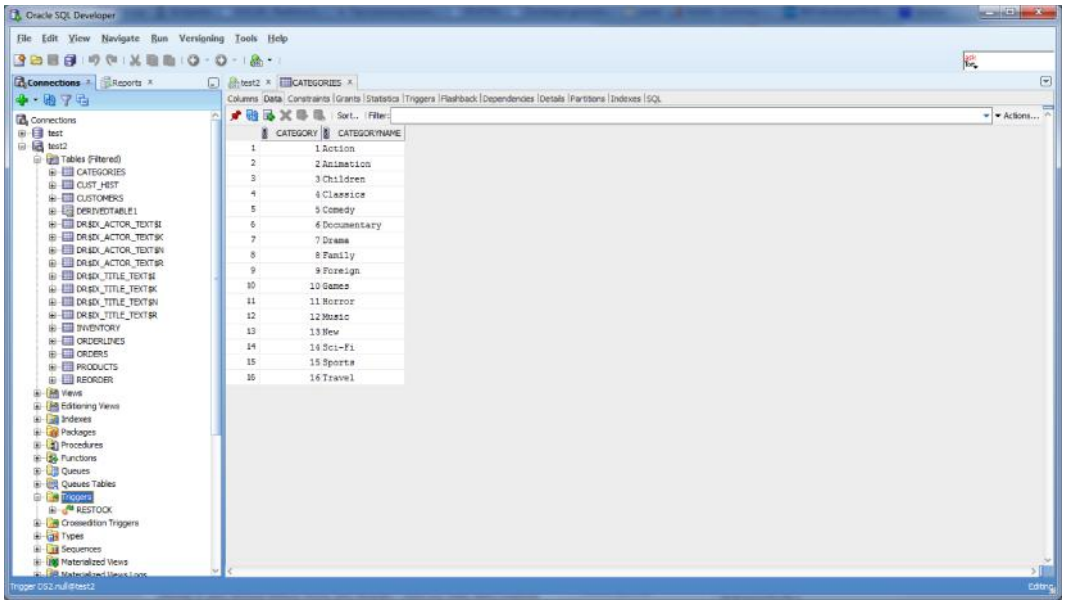
1.2.2. Oracle administravimo sąsajos

Java kalba parašyta „*SQL developer*“ (žr. 4 pav.) programa leidžia administruoti Oracle duomenų bazę. Įrankis turi daug funkcijų DBVS administravimui, tačiau Java parašyta programa veikia palyginus lėtai su kitomis nagrinėjamomis administravimo sąsajomis.

Valdymas yra gana gerai išdėstytas, išskleidžiamame lange galima lengvai matyti lentelių atributų turinį, bet nematomas duomenų tipas. Programa leidžia peržiūrėti lentelių indeksus, trigerius, vidines procedūras ir funkcijas.

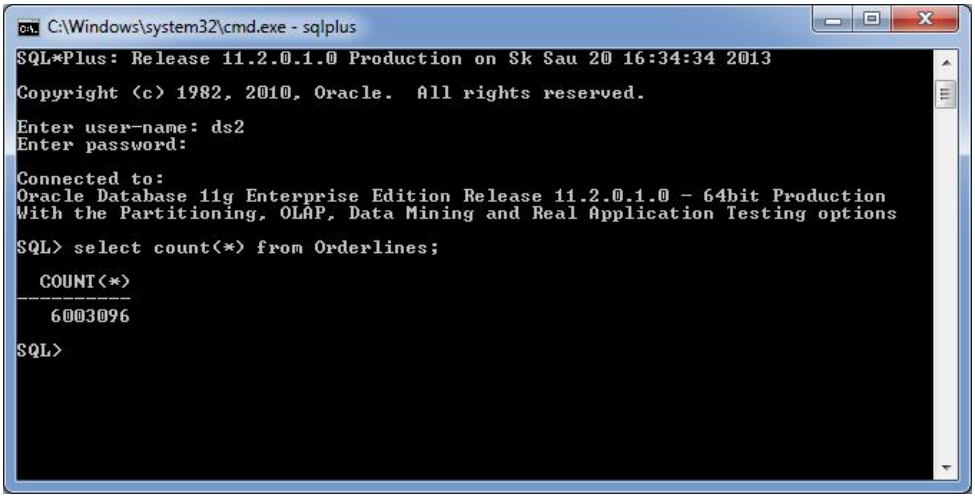
Programoje įdiegta versijavimo sistema leidžianti administruoti duomenų bazę lengviau. Jeigu kas nors blogo įvyksta galima atsistatyti prie tai buvusią versiją ir dirbti toliau.

Įdiegta daug papildomų galimybių leidžiančių importuoti bei eksportuoti duomenis.



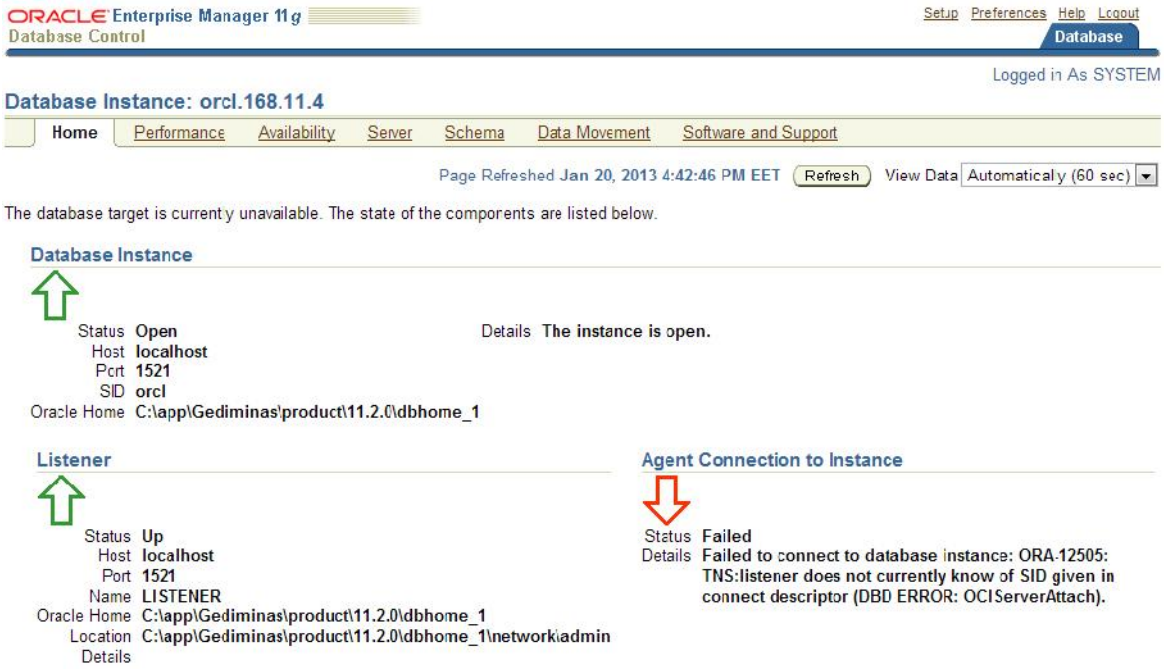
4 pav. Oracle administravimas per SQL Developer

Taip pat galima prisijungti ir per komandinės eilutės (žr. 5 pav.) komandą įvedus „sqlplus”. Paleidus komandą bus prašomas vartotojo vardas, bei slaptažodis.



5 pav. Oracle prisijungimas per komandinę eilutę

Trečias prisijungimo prie DBVS administravimo yra per naršyklę (žr. 6 pav.). Tereikia į naršyklę įvesti adresą kuris buvo nurodytas DBVS įdiegimo metu. *Enterprise* versijos numatytasis adresas yra: „https://localhost:1158/em“. Atsidarius puslapiui reikia prisijungti prie sistemos. Prisijungus galima stebėti DBVS būklę. Matyti kas blogai su sistema ar atlikti pakeitimus lemiančius našumą.



6 pav. Oracle administravimas per WEB sąsają

1.2.3. PHP ir Oracle susijungimo sąsajos

PHP ir Oracle duomenų bazės sąsajos yra šios [6]:

1. OCI8 - šį sąsaja įgalina prisijungimus prie Oracle duomenų bazės 11g, 10g, 9i ir 8i versijų. Užklausas galima perduoti SQL ar PL/SQL kalbomis. Ši sąsaja yra procedūrinio tipo, norint programuoti objektiškai reiktų rinktis sekančią sąsają.

2. PDO (OCI) - sąsaja palaikanti daugelį DBVS. Programos kodas nepriklauso nuo duomenų bazės sistemos. Užtenka pakeisti duomenų bazės prisijungimo duomenis ir nurodyti kurią DBVS sistemą naudoti. Šios sąsajos naudojimui reikia atsiminti, kad naudojama serverio architektūra turėtų atitikti įrašytos Oracle duomenų bazių valdymo sistemos architektūrą. Taigi jai abi sistemos yra 32bitų ar 64 bitų tai jos veiks be papildomo konfigūravimo.

1.2.4. Oracle privalumai ir trūkumai

Privalumai iš analizės metu surinktų duomenų:

- Pilnai palaiko ACID reikalavimus;
- Neribota duomenų bazės lentelių talpa;
- Palaiko transakcijas, vidines procedūras, trigerius.

Trūkumai iš analizės metu surinktų duomenų:

- Didžioji dalis siūlomų DBVS leidimų yra mokamos;
- Prastas PHP sąsajos atnaujinimo galimybės.

1.3. PostgreSQL analizė

PostgreSQL – viena iš nemokamų į objektus orientuotų reliacinių duomenų bazių valdymo sistemų, turinti daugiau nei 15 metų patirtį saugant ir apdorojant duomenis. PosgreSQL yra sukurta Kalifornijos Berklio universiteto akademinės veiklos tyrėjų. Pradžioje ši duomenų bazės valdymo sistema vadinosi Ingres ir toliau viena iš jos atšakų pavadinta Postgres. Ši duomenų bazių valdymo sistema yra kuriama ne vienos kompanijos, o daugelio programuotojų, kurie prisideda prie šios sistemos tobulinimo.

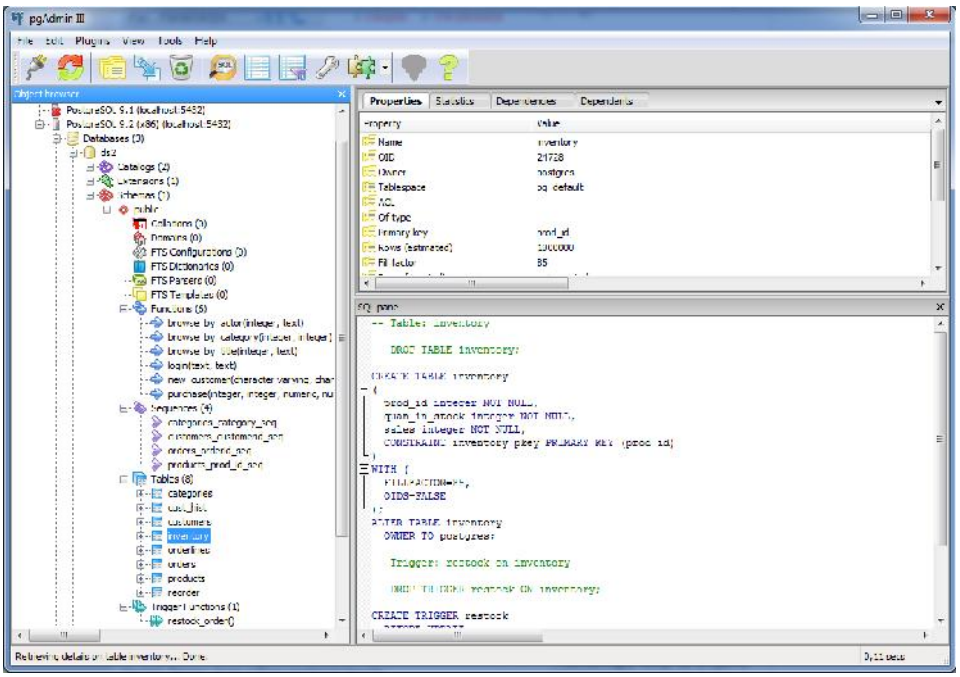
1.3.1. PostgreSQL galimybės

PosgreSQL 9.2 versijos ir naujesnės ypatumai [13]:

- Daugiaplatformė – Oracle palaiko visas pagrindines operacines sistemas, kaip Windows, Unix ir Mac.
- Transakcijos – pilnai palaiko ACID keliamus reikalavimus.
- Palaiko indeksavimą, išorinius raktus, trigerius ir vaizdus.
- Duomenų bazės lentelės didžiausias dydis yra 32Gb.
- SQL2008 standartas [10] – DBVS palaiko daugelį šio standarto nuostatų.
- Programavimo galimybės – vidines procedūras galima rašyti tiek PL/SQL, tiek vidine PLpg/SQL, tiek C, tiek perl, bei python procedūrinėm kalbomis.

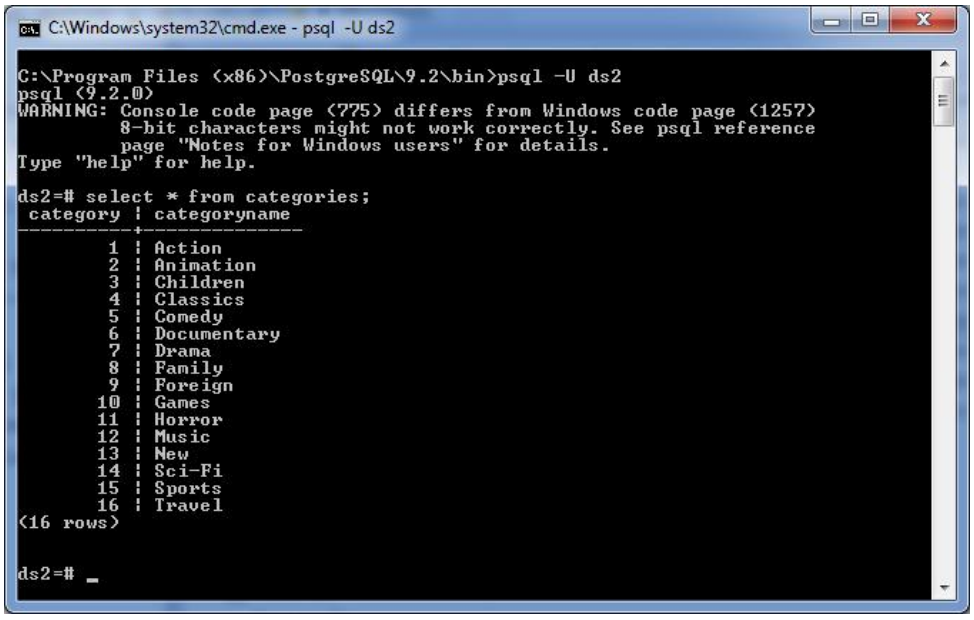
1.3.2. PostgreSQL administravimo sąsajos

PostgreSQL duomenų bazės valdymo sistemą kūrėjai rekomenduoja administruoti per pgAdmin III administravimo programą (žr. 7 pav.). Programa yra lengva naudotis, dėl minimalios sąsajos ir aiškių komponentų vartotojo sąsajoje. Kairėje pusėje aiškiai matomos pasirinktos duomenų bazės lentelės, funkcijos, trigeriai ir visi kiti nustatymai. Pasirenkant konkrečią lentelę generuojami sukūrimo SQL fragmentai, tas leidžia lengvai matyti visą informaciją apie lentelės atributų tipus, bei ryšius su kitomis lentelėmis.



7 pav. PgAdmin III administravimo programa

Taip pat prisijungti prie PostgreSQL duomenų bazės galima ir per komandinę eilutę. Tereikia komandinėje eilutėje (žr. 8 pav.) įvesti „psql –U vartotojo_vardas“ ir bus paprašomas slaptažodis. Pagal nutylėjimą yra išjungti užklausos įvykdymo laiko pranešimai.



8 pav. PostgreSQL administravimas per komandinę eilutę

Kaip ir MySQL administravimo sistemoje per naršyklę yra sukurtas panašus analogas į phpMyAdmin. Programa vadinasi phpPgAdmin (žr. 9 pav.). Ji yra nemokama ir galima atsisiuntus iš interneto įsidiegti serveryje ir lengvai administruoti DBVS.

9 pav. PosgreSQL administravimas per phpPgAdmin naršyklėje

1.3.3. PHP ir PostgreSQL susijungimo sąsajos

PHP-PostgreSQL duomenų bazės susijungimo sąsajos yra šios [6]:

1. PGSQL - sąsaja įgalina prisijungimus prie PostgreSQL 6.5 ir naujesnės versijos duomenų bazių. Ši sąsaja yra procedūrinio tipo.

2. PDO PGSQL - sąsaja palaikanti daugelį DBVS. Programos kodas nepriklauso nuo duomenų bazės sistemos. Užtenka pakeisti duomenų bazės prisijungimo duomenis ir nurodyti kurią DBVS sistemą naudoti. Norint naudoti šią sąsaja būtina PostgreSQL prieigos biblioteka Apache serveryje.

1.3.4. PostgreSQL privalumai ir trūkumai

Privalumai iš analizės metu surinktų duomenų:

- Nemokama;
- Neribota duomenų bazės lentelių talpa;
- Palaiko transakcijas, vidines procedūras, trigerius;
- Galima susikurti savo duomenų saugojimo struktūras;
- Pilnai palaiko ACID reikalavimus.

Trūkumai iš analizės metu surinktų duomenų:

- Prastas sąsajos atnaujinimo galimybės;
- Labai mažai internetinių puslapių talpinimo kompanijų siūlo šia DBVS;
- Reikia daug pastangų optimizuojant sprendimo našumui;
- Replikavimo galimybės yra ribotos.

1.4. MsSQL analizė

MsSQL – reliacinė duomenų bazių valdymo sistema iš leista Microsoft. Ši DBVS naudojama siekiant užtikrinti saugomą informaciją didelio saugumo reikalaujančiuose projektuose. MsSQL duomenų bazių valdymo sistemos paketus Microsoft išskirstė į šiuos specializuotus leidimus [14]:

- *Datacenter* – didžiausias paketas kuris leidžia serveriui palaikyti iki 256 loginių procesorių ir neribotą virtualią atmintį.
- *Enterprise* – didelėms duomenų saugykloms skirta versija palaikanti iki 524Pb dydžio duomenų bazes ir apribota naudoti iki dviejų terabaitų atminties, bei palaikanti iki 8 serverio procesorių.
- *Standard* – vidutinių galimybių leidimo paketas.
- *Web* – tai mažai kainuojanti ir internetiniam naudojimui skirta sistema. Funkcijos yra labai apribotos.
- *Express* – tai nemokama duomenų bazės valdymo sistema su daug apribojimų. Nors vartotojų kiekiui apribojamas nekeliamas, tačiau serveris gali veikti tik su vienu procesoriumi ir palaikyti iki 10Gb duomenų bazės bylą.

Kiekvienas leidimas skiriasi savo specifikacija ir leidžiamomis funkcijomis, dėl to norint naudoti šią DBVS projekte reikia apgalvoti kurį leidimą pasirinkti. Priklausomai nuo leidimo ir reikalingų funkcijų galima būtų mažinti finansinius projekto kaštus.

1.4.1. MsSQL galimybės

MSSQL 2008 R2 duomenų bazių valdymo sistemos galimybės [14]:

- Transakcijos – pilnai palaiko transakcijas. Taip pat kaupia transakcijų metu vykusias operacijas vykdymo operacijų žurnale.
- Vidinės procedūros – labai gerai optimizuotos procedūros, kurios sukompiliuojamos pačiame duomenų bazės serveryje ir taip pagreitinančios operacijų atlikimo spartą.
- Trigeriai – vienas iš palengvinimų plėtojant projekto logiką, pagerina projekto palaikymo darbus ir pagreitina darbą tarp kliento bei serverio, nes trigeriai vykdomi duomenų bazės serveryje.
- Saugumas – duomenų bazių administratoriai gali duoti prieigą ne tik prie lentelių, bet ir prie specifinių lentelės eilučių, taip užtikrinant, vartotojas naudosis tik jam suteiktomis privilegijomis.
- Klasterizuoti indeksai – tai duomenų bazės lentelės indeksavimas, pagal kokią nors grupę, kuris skiriasi nuo paprastojo indeksavimo, kai procesas vyksta visiems įrašams iš

eilės pagal surūšiavimą. Klasterizuoti indeksai padidina našumą, nes nereikia ieškoti visos grupės elementų indeksų kietajame diske, nes jie jau surūšiuoti.

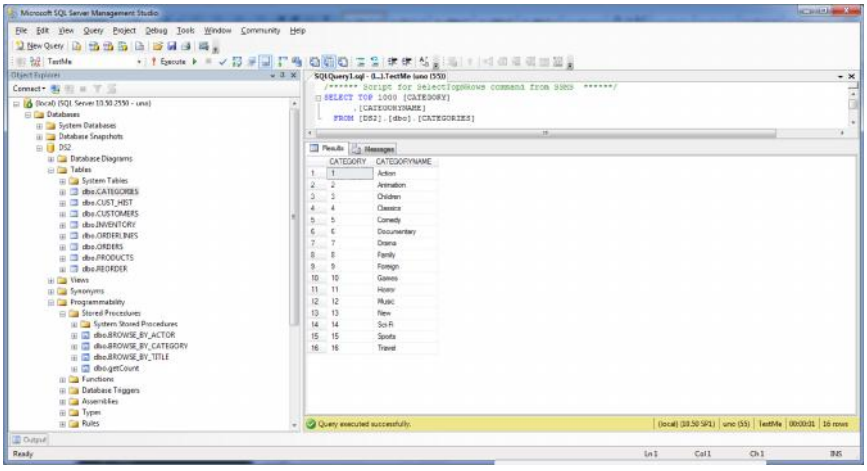
- Analizės įrankiai – įdiegus MsSQL duomenų bazės valdymo sistemą kartu įdiegiama programinė įranga skirta manipuliuoti duomenų baze bei analizuoti pačios duomenų bazės darbą ir atliekamų užklausų vykdymo statistiką.
- Atsarginių kopijų darymas – turi integruotą atsarginių kopijų darymo įrankį, kurio pagalba galima lengvai daryti atsargines duomenų kopijas į laikmenas ir nelaimės atveju greitai atstatyti duomenis iš laikmenų.

1.4.2. MsSQL administravimo sąsajos

SQL Server Management Studio (žr. 10 pav.) siūlomas Microsoft naudoti įdiegus duomenų bazės valdymo sistemą. Integruotas duomenų bazės lentelių diagramų kūrimo sąsaja, leidžia lengvai peržiūrėti sukurtos duomenų bazės lenteles, bei jų atributų sąryšius. Taip pat galima lengvai rasti vidinių procedūrų, bei funkcijų aprašymus, jas modifikuoti, bei testuoti debug režimu.

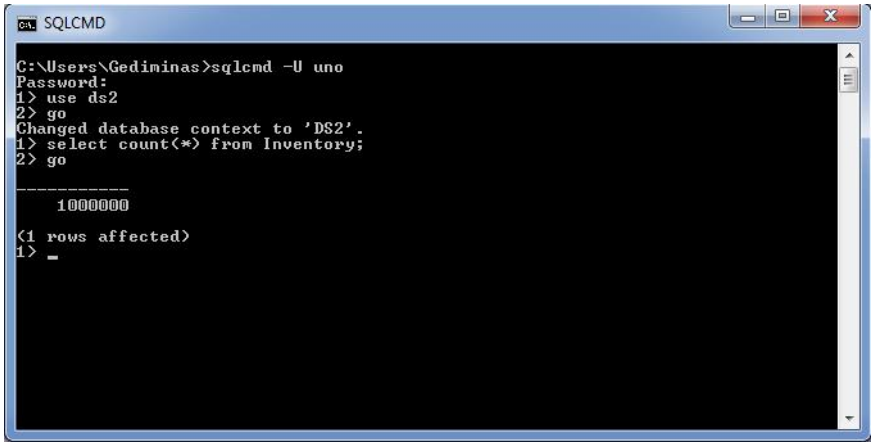
Užklausų atlikimo planavimo (angl. *Execution plan*) vaizdavimas palengvina duomenų bazės administravimo ir optimizavimo darbus. Kiekvienai užklausai formuojamas planas kiek procesai trunka, kokia procentinė kiekvienos užklausos įtaka darbui. Visgi, duomenų importavimo vedlio įrankis nelabai atitinka vartotojų lūkesčių. Reikia didelių pastangų tam ,kad importuoti duomenis iš CSV bylų, nes vedlys automatiškai nenori atpažinti duomenų bylos atributų reikšmių.

Vartotojų prieigą taip pat lengvai galima administruoti per šią programą. Sąsaja gan lengvai supranta ir skaitoma. Pasirinkus norimą vartotoją per nustatymus galima lengvai priskirti jam atitinkamą rolę, keisti prieigą prie leidžiamų duomenų bazių, keisti vartotojo slaptažodžius ir daugelį kitų dalykų.



10 pav. MsSQL administravimas per SQL Server Management Studio

Taip pat galima prie duomenų bazės valdymo sistemos prisijungti ir per komandinę eilutę (žr. 11 pav.). Šis administravimo įrankis tinka labiau pažengusiems administratoriams, kai reikia atlikti kokią nors paprastą administravimo užklausą. Pagal nutylėjimą yra išjungti užklausų vykdymo laikai, bei kiekviena kartą rašant užklausą reikia parašyti žodį „go“.



11 pav. MsSQL administravimas per komandinę eilutę

1.4.3. PHP ir MsSQL susijungimo sąsajos

Microsoft siūlo oficialias prisijungimo prie PHP sąsajas „Microsoft Drivers for PHP for SQL Server“ [6]. Galimos versijos yra 2.0 ir 3.0. Pirmoji 2.0 versija skirta PHP 5.2.X ir 5.3.X versijoms tačiau bendrauti gali tik su MSSQL 2008 versija. Antroji 3.0 versija skirta PHP 5.3.X ir 5.4.X versijoms ir gali bendrauti tik su MSSQL 2012. Abiejų versijų sąsajos tvarkyklės siūlo prisijungimą per Sqlsrv sąsają, bei per PDO mssql.

Sqlsrv sąsaja – yra funkcinė, nepalaiko objektinio programavimo principų. Tačiau gali atlikti prisijungimo prie duomenų bazės, užklausos formavimo, užklausos rezultatų išvedimą. Ši sąsaja tiktų mažiems projektams kuriems nereikia daug objektinio kodo.

PDO mssql sąsaja – yra skirta objektiniam programavimui.

Šios abi sąsajos veikia tik Windows operacinėj sistemoj įdiegtiems PHP serveriams.

1.4.4. MsSQL privalumai ir trūkumai

Privalumai iš analizės metu surinktų duomenų:

- Paprasta administravimo sąsaja
- Klasterizuoti indeksai

Trūkumai iš analizės metu surinktų duomenų:

- Veikia tik ant Microsoft serverių
- Mokama komercinei veiklai
- Prastas duomenų importavimas iš kitų duomenų bazių

1.5. DBVS analizės išvados

Analizės metu gauti rezultatai apibendrinti 3 lentelėje:

3 Lentelė. PHP duomenų bazių plėtinių palyginimas

	MySQL	**Oracle**	**PostgreSQL**	**MSSQL**
Minimali PHP versija	Mysql – 4.0, Mysqli – 5.0, PDO mysql – 5.1	OCI8 – 5.0 PDO oci – 5.1	PGSQL – 4.0 PDO pgsql – 5.1	SQLSRV – 5.2 PDO mssql - 5.3
PHP plėtinių skaičius	3	2	2	2
Vidinių procedūrų programavimas	Turi tik naudojant su mysqli ir PDO sąsajomis	Turi	Turi	Turi
Transakcijos	Tik su mysqli, PDO ir tik su InnoDB lentelėmis	Turi	Turi	Turi
Indeksai, Išoriniai raktai, Vaizdai, Trigeriai	Turi, išskyrus MYISAM – neturi išorinių raktų	Turi	Turi	Turi
Programavimo kalbos	SQL	SQL, PL/SQL	SQL, PL/SQL, ANSI SQL, PLpg/SQL	SQL, PL/SQL, ANSI SQL
Ar DBVS mokama?	Ne	Taip	Ne	Taip
Didžiausias lentelės dydis	MYISAM – 256TB InnoDb – 64Gb	128Tb per failą, tačiau jie gali būti apjungti	32 Gb	524Pb
Atnaujinamų versijų dažnumas	Dažnas	Retas	Vidutinis	Retas
SQL standartai	SQL99	SQL2003	SQL2008	
Integruota administra-vimo sąsaja	Neturi	Turi	Turi	Turi

Atlikus duomenų bazių valdymo sistemos analizę gauta, kad MySQL duomenų bazė su PHP turi daugiausia susijungimo sąsajų. Žemesnę nei PHP5 versijoje galimos tik dvi susijungimo sąsajos su MySQL ir PostgreSQL duomenų bazėmis. Iš visų analizuotų PHP sąsajų PDO oci yra eksperimentinė ir kuriama, o mysql PHP sąsaja nuo 5.5.0 yra skaitoma pasenusi ir tolimesnėse PHP versijose bus panaikinta. Iš analizės metu gautų rezultatų bus suformuluoti reikalavimai eksperimentiniam tyrimui atlikti.

2. SISTEMOS PROJEKTAVIMAS

Šiame skyriuje apžvelgiami sistemos projektavimo reikalavimai, duomenų bazės lentelių kūrimo eiga bei jų struktūra. Taip pat apžvelgiami eksperimento komponentų ir panaudos atvejų diagramų aprašai ir PHP plėtinių skirtų duomenų bazių valdymui suderinimo specifikacija. Tai atliekama tam, kad sukurti eksperimento aplinką, skirtą ištirti duomenų bazių ir PHP plėtinių greitaveikos nustatymui.

2.1. Funkciniai reikalavimai:

Pasinaudojant analizės dalyje nagrinėjamų duomenų bazių savybėmis iškeliami šie funkciniai reikalavimai, kurie reikalingi duomenų bazių lentelių užpildymui ir eksperimentams atlikti.

1. Tyrime tiriamos duomenų bazės lentelės turi turėti bent vieną šių galimybių: indeksuotus atributus, atributų išorinius raktus, didesnį nei 1mln kiekį eilučių.
2. Tyrime turi būti testuojamos bent dviejų DBVS ir PHP susijungimo sąsajos.
3. Sistemos kūrime sukonfigūruoti taip PHP serverio nustatymus, kad visos nagrinėjamos DBVS galėtų susijungti su serveriu.
4. Padaryti įrankį, kuris leistų įjungti/ išjungti testų metu nenaudojamas duomenų bazes.
5. Sudaryti duomenų bazes, užpildyti testiniais duomenimis ir parašyti testavimo užklausas eksperimentiniam tyrimui atlikti.

Gauti eksperimentinio tyrimo rezultatai leis nustatyti PHP duomenų bazių plėtinių ir DBVS veikimo greitaveiką.

2.2. Sistemos projektas

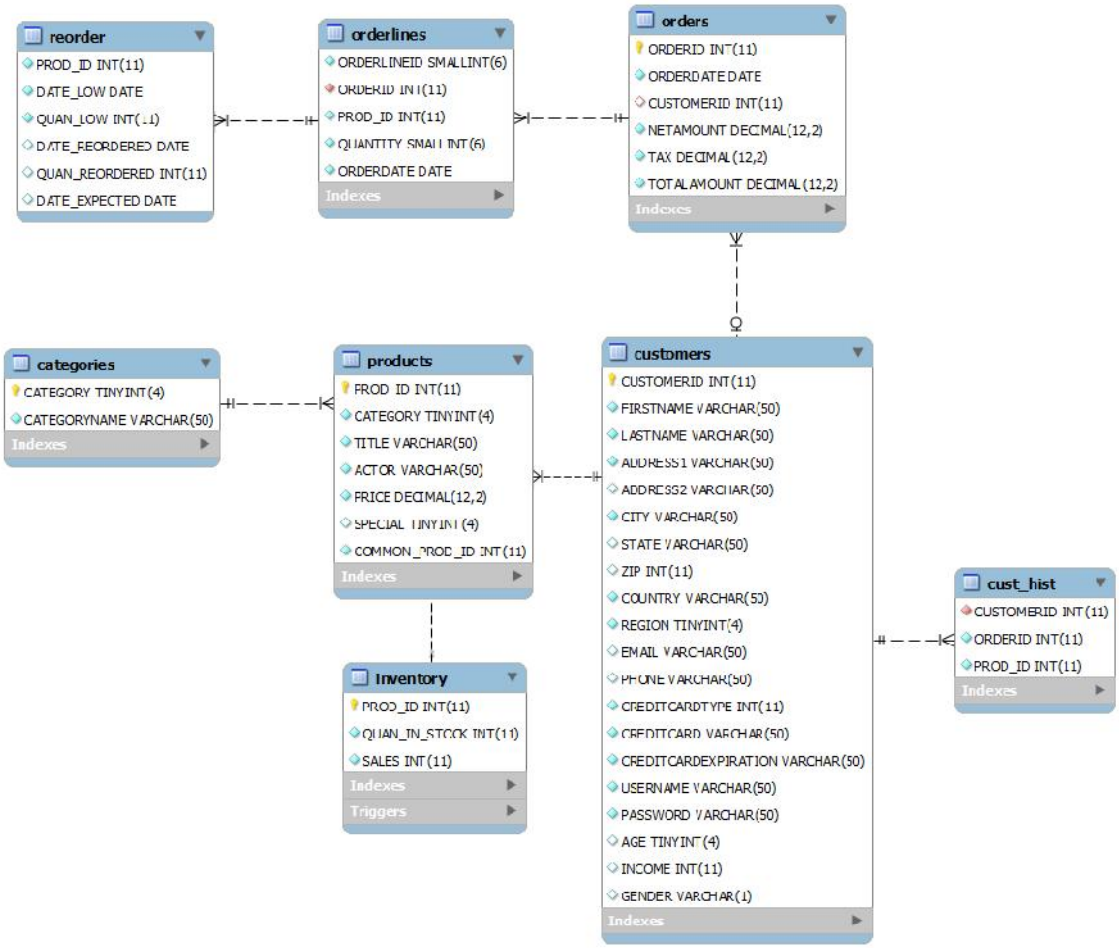
Sistema yra sudaryta iš keturių pasirinktų duomenų bazių valdymo sistemų: MySQL 5.5.8, Oracle 11g, PostgreSQL 9.2 ir MSSQL 2008 R2. Serveriui pasirinkta Apache 2.2.21 versija su PHP 5.3.5 versija.

Kiekvienoje sistemoje duomenų bazės schema yra sudaryta iš vienodų lentelių. Lentelėse atributų laukai skiriasi tik nuo duomenų bazės valdymo sistemos nustatytų duomenų tipo, nors patys duomenys yra identiški.

Atlikti bandymus su pasirinkta duomenų bazės schema (MySQL, Oracle, PostgreSQL, MsSQL), bei užsaugoti gautus tyrimo rezultatus, tam kad galėtume vėliau juos išanalizuoti ir aprašyti.

2.2.1. Duomenų bazės struktūros diagrama

Duomenų bazių struktūros diagrama pavaizduota 12 paveiksle. Duomenų bazėse sukurtos aštuonios lentelės: categories, cust_hist, customers, inventory, orderlines, orders ir products. Įterptų į lenteles įrašų skaičius matomas 4 lentelėje.



12 pav. Duomenų bazės lentelių schema

Duomenų bazės lentelės skirtingose DBVS turi tuos pačius atributų pavadinimus, tik duomenų tipai skiriasi priklausomai nuo DBVS galimų atributų duomenų tipų (skirtumus žiūrėti 1 priede).

4 Lentelė. Duomenų kiekis(eilučių skaičius) sukurtose lentelėse

Lentelės pavadinimas	**DBVS pavadinimas**			
	MySQL	**Oracle**	**PostgreSQL**	**MSSQL**
categories	16	16	16	16
Cust_hist	6003096	6003096	6003096	6003096
Customers	2000000	2000000	2000000	2000000
Inventory	1000000	1000000	1000000	1000000
Orderlines	6003096	6003096	6003096	6003096
Orders	1200000	1200000	1200000	1200000
Products	1000000	1000000	1000000	1000000

Duomenų bazes duomenimis užpildyti buvo naudojama Dell DVDStore [15] duomenų generavimo programa. Programos pagalba sukuriama vienoda daugeliui DBVS sistemos suprantama CSV byla, kurioje yra duomenys.

2.2.2. Sistemos komponentų diagrama

Sistemos komponentų diagramoje 13 pav. pavaizduotas eksperimentui naudojami komponentai. Testavimo programos LoadUI komponentą sudaro: MySQL, MsSQL, PostgreSQL ir Oracle testavimo užklausos.

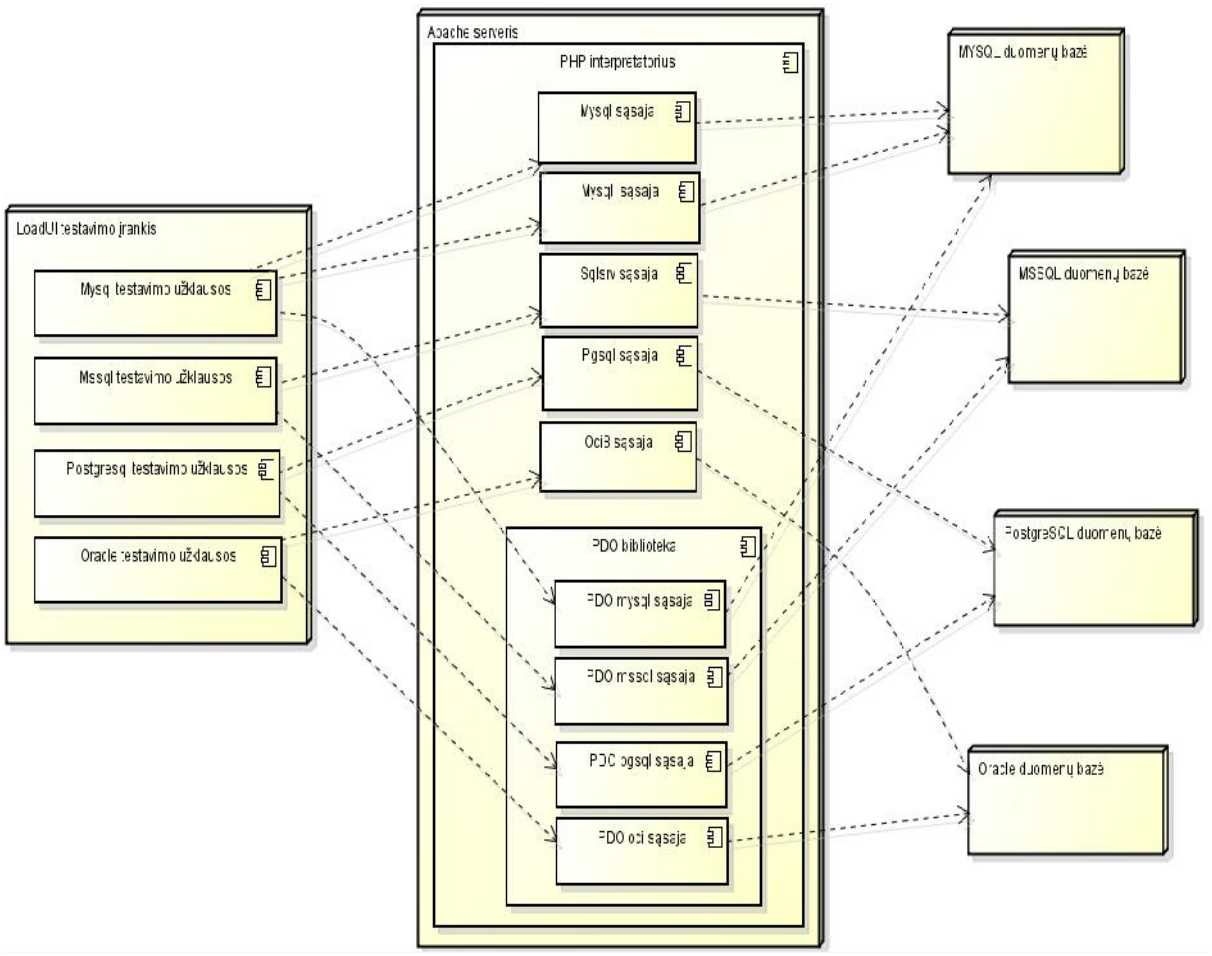
Apache serveryje įdiegtos PHP plėtinių sąsajos skirtos ryšiui su MySQL, MsSQL, PostgreSQL ir Oracle duomenų bazių valdymo sistemomis.

Su MySQL duomenų baze susijungimams naudojamos: mysql, mysqli ir PDO mysql sąsajos.

Su MsSQL duomenų baze susijungimams naudojamos: sqlsrv ir PDO mssql sąsajos.

Su PostgreSQL duomenų baze susijungimams naudojamos: pgsql ir PDO pgsql sąsajos.

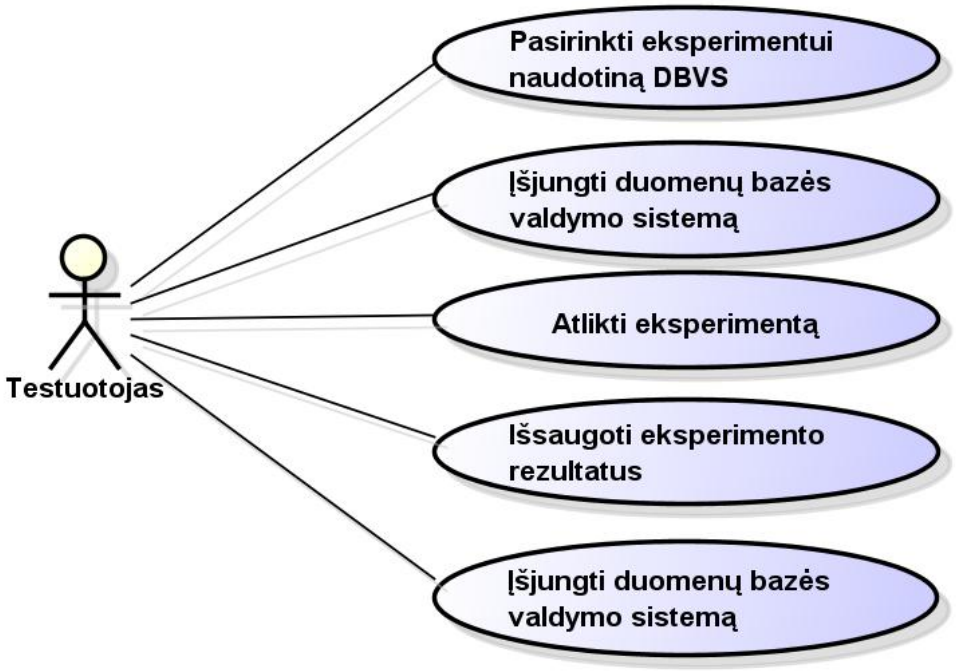
So Oracle duomenų baze susijungimams naudojamos: oci8 ir PDO oci sąsajos.



13 pav. Sistemos komponentų diagrama

2.2.3. Panaudos atvejai

Eksperimentams įvykdyti naudojama panaudos atvejų diagrama pavaizduota 14 pav.



14 pav. Eksperimentinių tyrimų panaudos atvejai

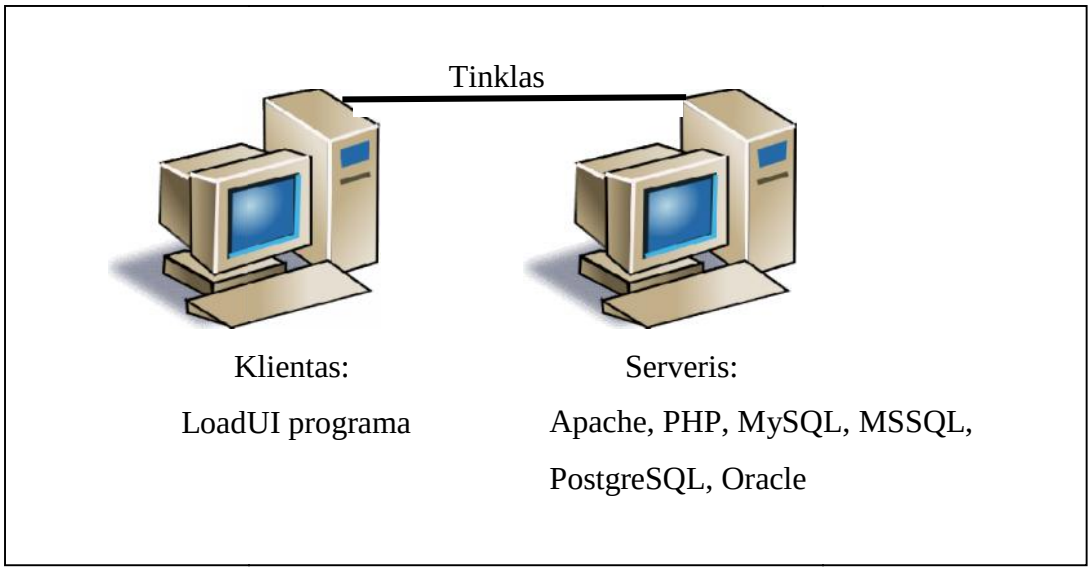
Eksperimento vykdymo eigoje testuotojas pirmiausia norimą testuoti duomenų bazės valdymo sistemą. Testuotojas įjungia duomenų bazės valdymo sistemą. Įsijungus duomenų bazės valdymo sistemai atlieka eksperimentą. Pasibaigus eksperimentui testuotojas išsaugo gautus rezultatus ir išjungia duomenų bazės valdymo sistemą.

2.3. Sistemos realizacija

2.3.1. Eksperimentų vykdymo įrankis

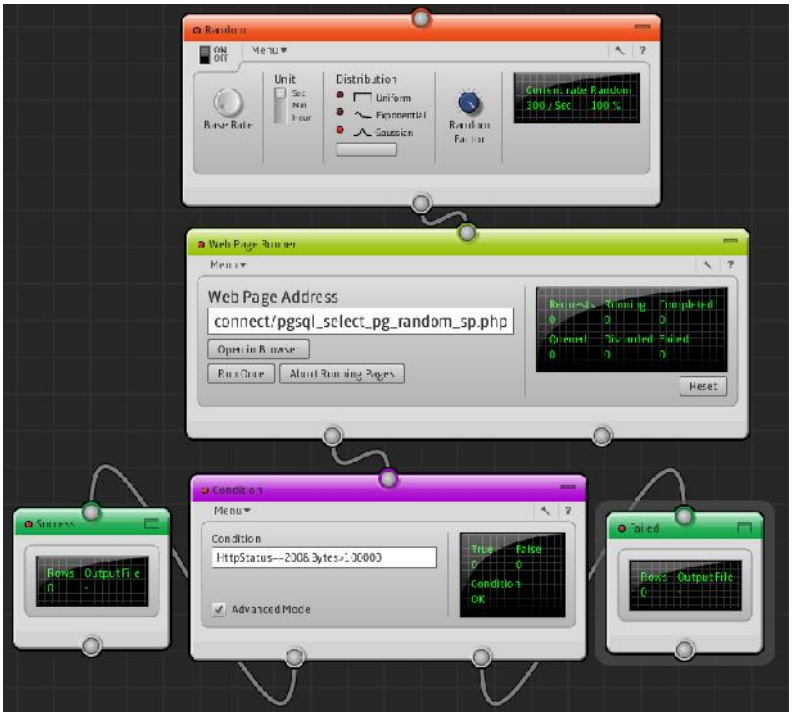
Testavimo įrankis LoadUI yra pasirinktas atlikti eksperimentiniam tyrimui. Šio įrankio pagalba yra generuojami norimo vartotojų kiekio kreipiniai į Apache serverį. Parinkus reikiamą eksperimento testą LoadUI programoje yra nurodomas vykdymo laikas, taip pat yra pasirenkamas testavimo greitis ir atsitiktinis kreipinių kitimas, tam kad imituotų serverio darbą.

Eksperimentiniam tyrimui atlikti yra naudojami du kompiuteriai, viename yra testavimo programa LoadUI, o sekančiame kompiuteryje yra įdiegta Apache programa ir keturios duomenų bazių valdymo sistemos: MySQL, Oracle, MsSQL ir PostgreSQL. Šie du kompiuteriai yra sujungti LAN kabeliu. Bendra eksperimentinio tyrimo aparatūrinė schema pavaizduota 15 paveiksle.



15 pav. Eksperimentų atlikimo schema

Kiekvieno testo metu yra įjungiamas tos duomenų bazės serverio paslaugos, kurios bus testuojamos.



16 pav. LoadUI testavimo įrankio vaizdas

Random bloke 16 pav. nustatomas užklausų generavimo dažnumas bei atsitiktinis paskirstymo laikas pagal Gauso kreivę. Testo adresas įrašomas į lauką „*Web Page Address*“, kuris kiekvienam testui yra atskiras. Vykdant testą jo informaciniuose laikuose matoma būsena: užklausų skaičius, veikiančių, bei įvykdytų užklausų skaičius. Taip pat rodoma eilė, kuri susidaro dėl nespėjamų įvykdyti užklausų bei nesėkmingų užklausų įvykdymų skaičius.

Condition skiltyje galima stebėti ar įvykdyta užklausa buvo tikrai įvykdyta. Užklausos įvykdymo kodas, bei gautas duomenų kiekis parodo jos teisingumą ir toliau užklausos statistika yra užsaugoma į „Success“ arba „Failed“ langus.

2.3.2. DBVS paleidimas ir išjungimas

Sistemos paleidimui reikia įjungti konkrečias duomenų valdymo sistemų servisus. Testavimas yra atliekamas su viena mašinos ir palengvinant duomenų bazių įjungimus ir išjungimus yra sukurti specialūs skriptai (žr. 5 lentelę).

5 Lentelė Duomenų bazių valdymo sistemų išjungimo ir įjungimo servisų bylos

Veiksmas	**Duomenų bazės valdymo sistemos pavadinimas**			
	MySQL	**Oracle**	**PostgreSQL**	**MSSQL**
Įjungimas	mysql_start.bat	oracle_start.bat	pg_start.bat	ms_start.bat
Išjungimas	mysql_stop.bat	oracle_stop.bat	pg_stop.bat	ms_stop.bat

Priklausomai nuo operacinės sistemos nustatymų gali būti pareikalautos administratoriaus teisės paleidžiant šiuos skriptus.

2.3.3. MySQL ir PHP sąsajos tvarkyklės suderinimas

Mysql ir PHP prisijungimas yra vienas iš paprasčiausių iš nagrinėjamų atvejų. Įrašius PHP, prisijungimai jau yra įdiegti.

Norint įgalinti PHP bendrauti su MySQL reikia tik konfigūraciniam „php.ini“ faile įtraukti šias bylas:

```
extension=php_mysql.dll
extension=php_mysqli.dll
extension=php_pdo_mysql.dll
```

Atlikus šį veiksmą reikia iš naujo paleisti MySQL servisą.

2.3.4. Oracle ir PHP sąsajos tvarkyklės suderinimas

Oracle ir PHP sąsajos skirtingų versijų suderinamumas yra ganėtinai komplikuotas. Todėl norint įgalinti jų sąveiką reikia tiksliai žinoti:

- PHP versiją, jos architektūros tipą, kompiliatoriaus pavadinimą
- Oracle duomenų bazės versiją ir tipą

Norint įgalinti PHP bendrauti su Oracle reikia tik konfigūraciniam „php.ini“ faile įtraukti šias bylas:

```
extension=php_oci8.dll
extension=php_pdo_oci.dll
```

Kadangi testavimui buvo pasirinkta Oracle 11g x64 duomenų bazės versija reikėjo atlikti pakeitimui sistemoje [12], nes PHP papildinys neatpažino sąsajos bylų tarp Oracle ir PHP.

Atlikus šios veiksmus reikia iš naujo paleisti Oracle servisą.

2.3.5. PostgreSQL ir PHP sąsajos tvarkyklės suderinimas

Kaip ir MySQL sąsajos su PHP atveju bylos kuriomis PostgreSQL bendrauja su PHP yra įdiegtos pagal nutylėjimą. Tačiau papildomai reikia Apache serverio konfigūraciniam „httpd.conf“ faile nurodyti kelią iki „libpq.dll“ bibliotekos.

Norint įgalinti PHP bendrauti su PostgreSQL reikia tik konfigūraciniam „php.ini“ faile įtraukti šias bylas:

```
extension=php_pgsql.dll
extension=php_pdo_pgsql.dll
```

Atlikus šios veiksmus reikia iš naujo paleisti PostgreSQL servisą.

2.3.6. MSSQL ir PHP sąsajos tvarkyklės suderinimas

MSSQL ir PHP sąsajos suderinamumas labai priklauso nuo jų versijų. Testavimui buvo pasirinkta MSSQL 2008 R2 versija yra suderinama tik iki PHP 5.4.X versijos. Didesnės versijos PHP suderinta su naujesne MSSQL versija.

Microsoft palaikymams yra išleidę dvi versijas 2.0 ir 3.0, kurių viena skirta 2008 versijai, o kita 2012 MSSQL versijai. Šioje sistemoje yra naudojama 2.0 versija, nes tik ji palaiko sąryšį tarp PHP ir MSSQL 2008 R2 duomenų bazės. Priklausomai nuo PHP versijos ir kompiliatoriaus tipo kuriuo buvo sukompiliuotos PHP bibliotekos buvo pasirinkta „53_ts_vc6“ sąsajos versija.

Norint įgalinti PHP bendrauti su PostgreSQL reikia tik konfigūraciniam „php.ini“ faile įtraukti šias bylas:

```
extension= php_sqlsrv_53_ts_vc6.dll
extension= php_pdo_sqlsrv_53_ts_vc6.dll
```

Atlikus šios veiksmus reikia iš naujo paleisti MSSQL servisą.

2.4. Apibendrinančios išvados

Suprojektavus sistemą eksperimentams atlikti, buvo sukurtos keturios duomenų bazės su MySQL, MsSQL, PostgreSQL ir Oracle duomenų bazių valdymo sistemomis. Duomenų bazės buvo užpildytos septyniomis lentelėmis, susietomis ryšiais bei bendrai užpildytos virš 17 milijonų įrašų. Duomenų bazėse buvo sukurtos vidinės procedūros, skirtos duomenų išrinkimui (žr. 3 Priedą). Kuriant duomenų bazės lenteles didesnio skirtumo tarp duomenų bazių neaptikta, išskyrus tai, kad duomenų tipo pavadinimų užrašymas nežymiai skyrėsi tarp skirtingų duomenų bazių. Pats duomenų užpildymas į duomenų bazes ir vidinių procedūrų aprašymas pagal sudėtingumo tvarką (nuo lengviausio iki sunkiausio) yra: MySQL, PostgreSQL, MsSQL, Oracle.

3. EKSPERIMENTINIS TYRIMAS IR REZULTATŲ ANALIZĖ

Šiame skyriuje nustatoma eksperimento tyrimo specifikacija bei jo atlikimo būdai ir glaustai aprašoma eksperimentinio tyrimo eiga ir gaunami rezultatai. Tyrimui yra naudojama sistema suprojektuota pagal antroje dalyje aprašytą projektą. Atlikus tyrimą bus galima nustatyti geriausią plėtinių darbui su PHP skirtą naudoti vidutinės apimties duomenų bazę.

3.1. Eksperimentų tyrimų nustatymai ir specifikacija

Eksperimentiniams tyrimams naudojamas vienas kompiuteris su šia aparatūrine modifikacija:

- Procesorius: Mobile DualCore Intel Core i3-370M, 2000 MHz;
- Operatyvinė atmintis: 6GB DDR3, 557Mhz;
- Kietasis diskas: 500 GB, SATA2, 5400 RPM 8MB Cache;
- Tinklo plokštė: Atheros AR8152 PCI-E Fast Ethernet Controller;
- Operacinė sistema: Windows 7 x64 Home Premium.

Tyrimo metu naudojamos programos serverio ir duomenų bazių programų pavadinimai ir versijos :

- Apache 2.2.21 su PHP 5.3.5
- MySQL 5.5.8
- MsSQL 2008 R2
- PostgreSQL 9.2
- Oracle 11g

Eksperimentinio tyrimo vykdymas:

- Generuojamas 300 per sekundę kreipinių srautas, kuris vykdomas lygiagrečiai ir didžiausias veikiančių procesų skaičius yra 100
- Vienam eksperimentiniam tyrimui skirtas laikas yra 21 minutė.
- Kiekvienas eksperimentinis tyrimas pakartotinai atliekamas 10 kartų.

Tyrimo metu yra vykdomi du eksperimentai:

- Su išrinkimu iš duomenų bazės eksperimentu yra simuliuojama atsitiktinis produktų užsakymo numeris ir gražinama penkiasdešimt vartotojų užsakytų produktų sąrašas nuo sugeneruoto numerio.
- Su įterpimu susijusiame eksperimente įterpiamas vienas užsakytas produktas į produktų užsakymo lentelę.

3.2. Eksperimentas su duomenų išrinkimu

3.2.1 Eksperimento tikslas

Šis eksperimentas atliekamas tam, kad ištirti duomenų išrinkimo iš duomenų bazės spartą, atsakų į užklausą laiką, bei operacijų kiekį ir spartą atliekamą eksperimento metu, naudojant skirtingus PHP plėtinius ir užklausas.

3.2.2 Eksperimento atlikimo tvarka

Eksperimento tikslui įgyvendinti yra būtina apskaičiuoti:

- Įvykdytų užklausų kiekį;
- Duomenų srautą;
- Užklausų įvykdymo laiką;
- Transakcijų įvykdymo spartą.

Eksperimento atlikimo tvarka:

- Įjungiame reikalingą eksperimentui duomenų bazės valdymo sistemą;
- Užduotą eksperimentą vykdome LoadUI programa 21 minutę;
- Eksperimentą kartojame dešimt kartų;
- Gautus rezultatus eksportuojame į Excel bylas;
- Excel bylas suformatuojame atitinkamomis formulėmis ir sukuriame eksperimento vykdymo grafikus;
- Toliau aprašytomis formulėmis (1-3) apskaičiuojame rezultatus.

Vidutinio duomenų srautui (toliau BPS) apskaičiuoti naudosime šią formulę:

$$BPS = \frac{Įvykdytų\ užklausų\ kiekis \times Užklausos\ dydis}{Vykdymo\ laikas} \quad (1)$$

Čia:

1. Įvykdytų užklausų kiekis – tai kiekis užklausų, kurios įvykdytos eksperimento metu.
2. Užklausos dydis – užklausos rezultato dydis, kuris gražinamas iš serverio. Mato vienetas – baitas.
3. Vykdymo laikas – eksperimento vykdymo laikas. Mato vienetas – sekundė.

Priklausomai nuo duomenų kiekio dydžio vidutinis duomenų srautas matuojamas Kb/s (kilobaitai per sekundę) arba Mb/s (megabitai per sekundę).

Užklausų atsakų vidutinius laikus apskaičiuojame pagal formulę (2):

$$Vidutinis\ užklausos\ laikas = \frac{Vykdymo\ laikas \times Procesų\ kiekis}{Įvykdytų\ užklausų\ kiekis} \quad (2)$$

Čia:

1. Vykdymo laikas – eksperimento vykdymo laikas. Mato vienetas – sekundė.

2. Procesų kiekis – tai lygiagrečiai vykdomų užklausų kiekis. Šiame eksperimente lygiagrečiai vykdomų procesų kiekis – 100.

3. Įvykdytų užklausų kiekis – tai kiekis užklausų, kurios įvykdytos eksperimento metu.

Vidutinį transakcijų skaičių (toliau TPS) galima paskaičiuoti pagal formulę (3):

$$TPS = \frac{Įvykdytų\ užklausų\ kiekis}{Vykdymo\ laikas} \quad (3)$$

Čia:

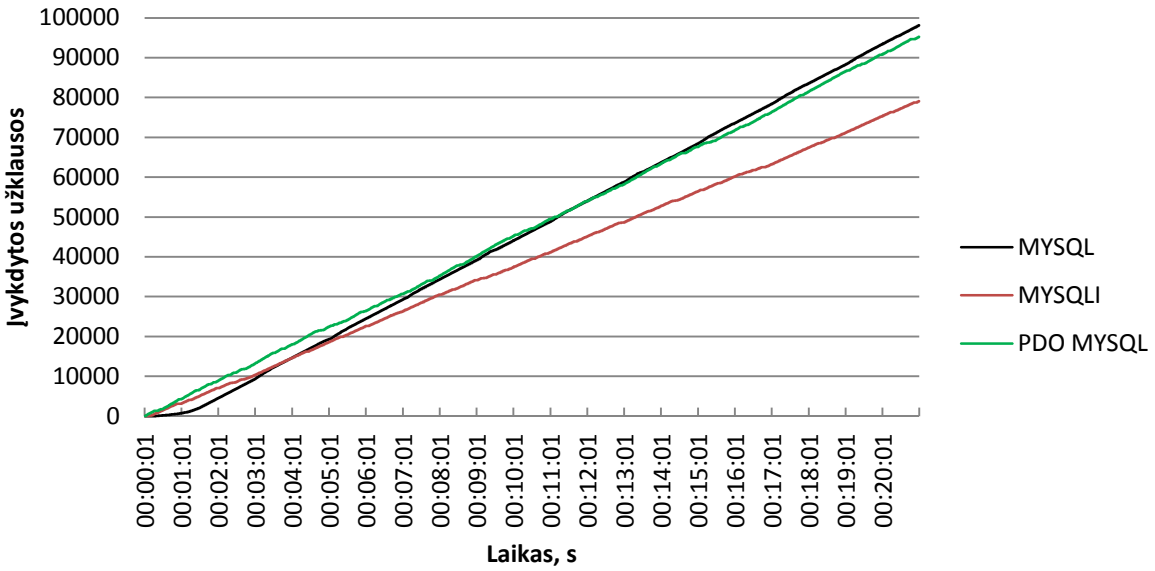
1. Vykdymo laikas – eksperimento vykdymo laikas. Mato vienetas – sekundė.
2. Įvykdytų užklausų kiekis – tai kiekis užklausų, kurios įvykdytos eksperimento metu.

Šiame eksperimente išrenkami duomenys iš duomenų bazės pasinaudojant ir paprastą užklausas (žr. 2 Priedą), ir užklausas, kurios yra vidinėje procedūroje (žr. 3 Priedą). Šios užklausos suformuojama penkiasdešimt įrašų apie vartotojų pirktus produktus, bei pirkusiojo vartotojo duomenis.

Eksperimentą kartojame dešimt kartų [16, 17] tam, kad padidintume duomenų imtį ir gautume tikslesnius rezultatus. Eksperimento metu veikia vienas duomenų bazės serveris, likę serveriai yra išjungiami.

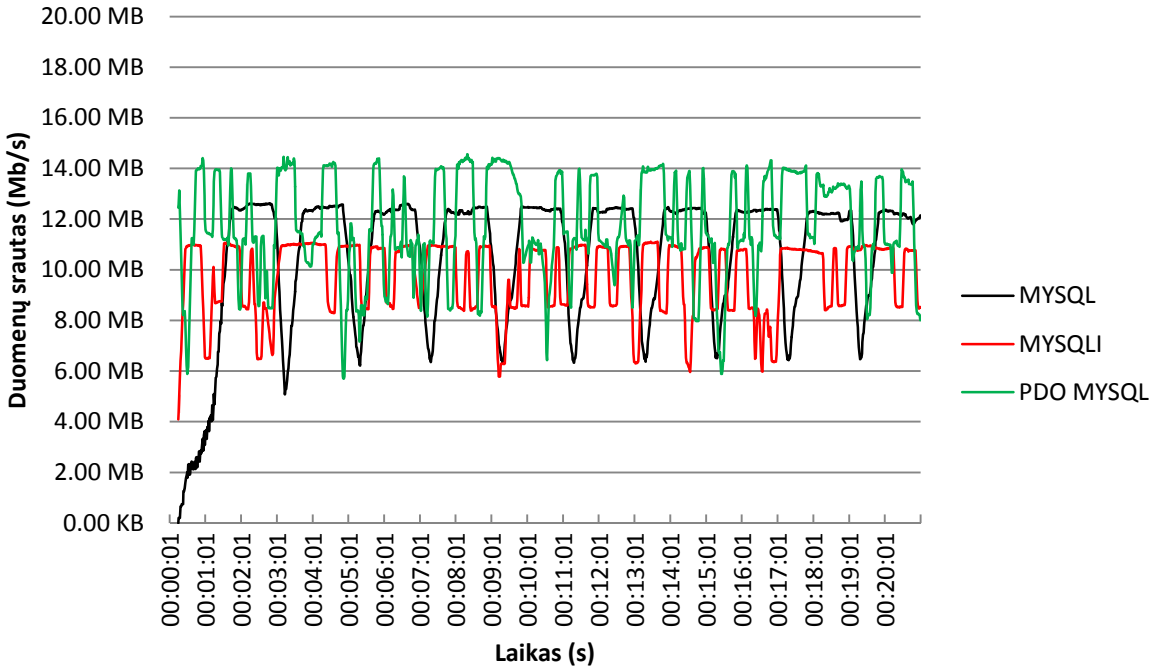
3.2.3 Eksperimento rezultatai

Eksperimentas su MySQL duomenų baze naudojant paprastą užklausą (žr. 2 Priede, 38 pav.) įvykdytas dešimt kartų ir didžiausią įvykdytų užklausų kiekį įvykdė mysql sąsaja įvykdžius vidutiniškai 98 tūkst. užklausų kiekį per eksperimento vykdymo laiką – 21 minutę. Eksperimento rezultatų grafiką matome 17 pav. Su mysqli sąsaja įvykdyta tik 79 tūkst. užklausų - tai net 19 tūkst. mažiau užklausų lyginant su mysql sąsaja. PDO mysql sąsaja įvykdė vidutiniškai apie 95,2 tūkst. užklausų eksperimentų metu.



17 pav. Įvykdytų užklausų kiekis su *MySQL*, naudojant paprastąją užklausą

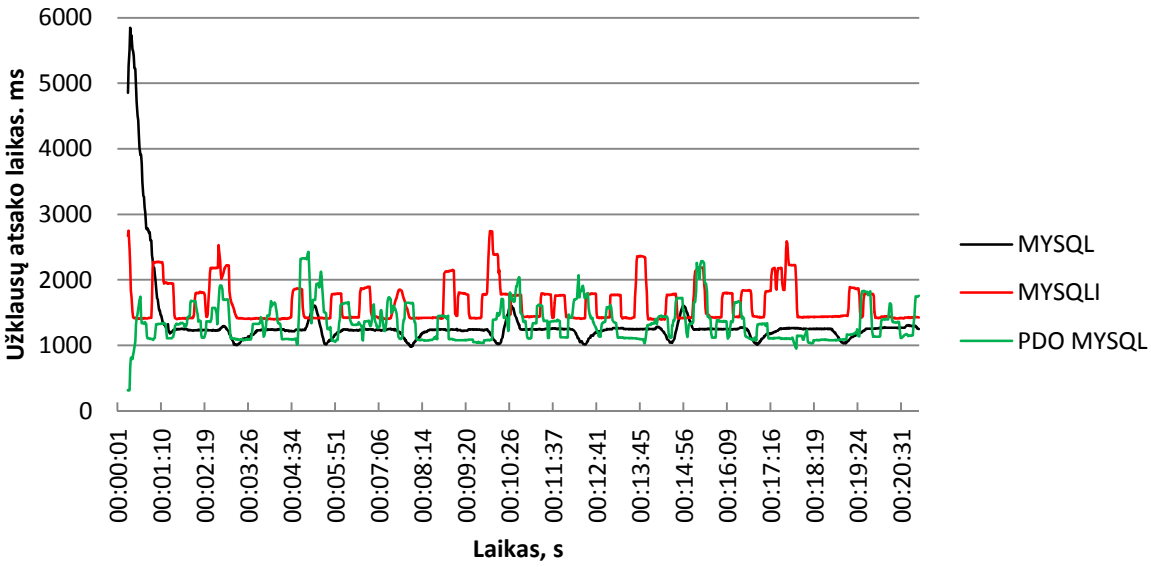
Atlikus eksperimentus su MySQL duomenų baze nustatyta, kad didžiausia duomenų perdavimo sparta buvo su mysql sąsaja. Iš įvykdytų užklausų ir žinant, kad vienos užklausos generuojamas duomenų srautas yra 159 Kb, galima apskaičiuoti vidutinius duomenų kiekio srautus pasinaudojus formule (1). Apskaičiavus eksperimento rezultatus gauname: vidutinis duomenų srautas su mysql plėtiniu yra 12,07 Mb/s, su mysqli – 9,73 Mb/s, o su PDO mysql – 11,73 Mb/s. Viso eksperimento metu kintantis duomenų srautas yra atvaizduotas 18 pav. Toks didžiulis amplitudės kitimas su PDO mysql sąsaja yra įtakotas Apache serverio, serveriui nespėjant susidoroti su užklausomis.



18 pav. Duomenų srautas su *MySQL*, naudojant paprastąją užklausą

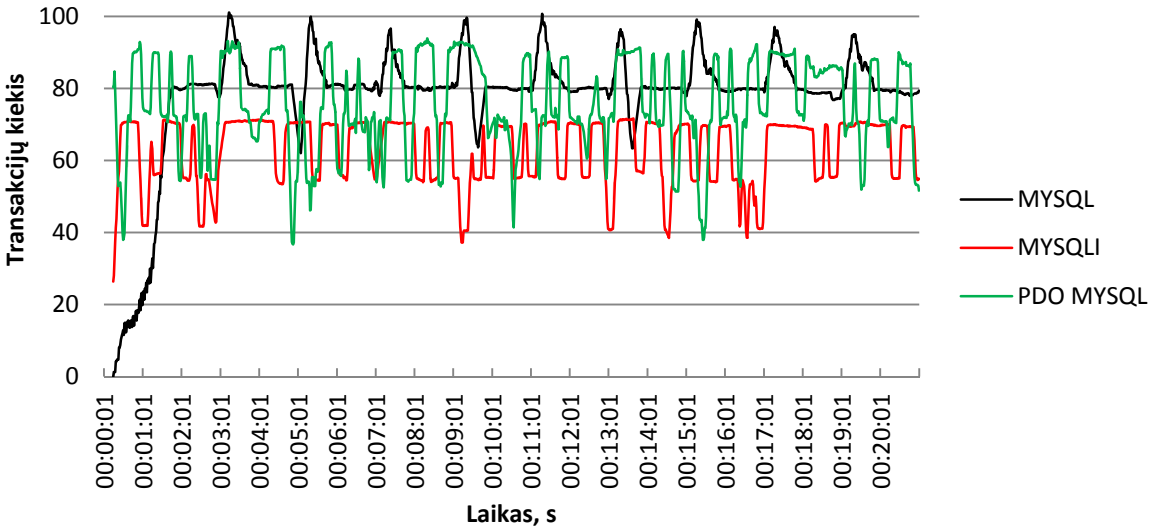
Eksperimento metu taip pat matuojama ir užklausų įvykdymo laikas, kuris matuojamas nuo užklausos nusiuntimo į serverį iki užklausos duomenų atsakymo atvaizdavimo testavimo įrankio modulyje.

Užklausų atsakų vidutinius laikus apskaičiuojame pagal formulę (2). Gautieji rezultatai su mysql sąsaja yra 1,28 sekundės kiekvienai užklausai, su mysqli sąsaja atsakų laikas gerokai išaugęs net iki 1,59 sekundės, o su PDO mysql sąsaja – 1,32 sekundės. Iš paveikslo 19 pav. matyti, kad su mysql sąsaja eksperimento pradžioje yra išaugęs užklausų atsako laikas. Tačiau maždaug po minutės vykdymo laiko jisai tampa beveik pastovus. Skirtingai nei mysql sąsaja, likusios dvi mysqli ir PDO sąsajos užklausų atsakų laikai laike kito nepastoviai. Tai įtakojo galutinį rezultatą.



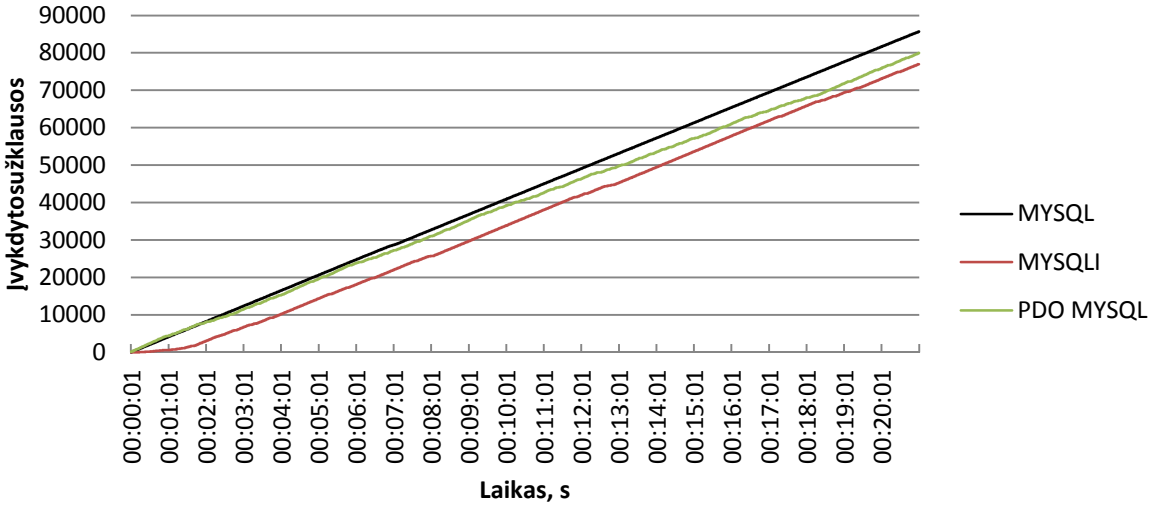
19 pav. Užklausų atsako laikas su *MySQL*, naudojant paprastąją užklausą

Paskutinis parametras kuris eksperimento metu buvo matuojamas yra transakcijų skaičius per sekundę. Kadangi ir įvykdytų užklausų kiekis ir atsakų kiekis su mysql sąsaja buvo geriausiai įvertintas eksperimento metu, tai ir ši sąsaja pasiekė didžiausią transakcijų kiekį per sekundę. Transakcijų kitimą eksperimento metu parodo 20 pav. Eksperimentų pradžiose mysql sąsajos transakcijų kiekis palyginus su mysqli ir PDO mysql sąsajomis, tačiau po 2 minučių eksperimento transakcijų kiekis išaugo. Vidutinį transakcijų skaičių galima paskaičiuoti pagal formulę (3). Gautieji rezultatai rodo, kad su mysql sąsaja buvo įvykdyta maždaug 77,7 transakcijos per sekundę, su mysqli sąsaja net 15 transakcijų per sekundę mažiau – 62,7 transakcijos per sekundę, o su PDO mysql pasiekta tik viena transakcija mažesnis rezultatas nei su mysql sąsaja, t.y. 76,6 transakcijos per sekundę.



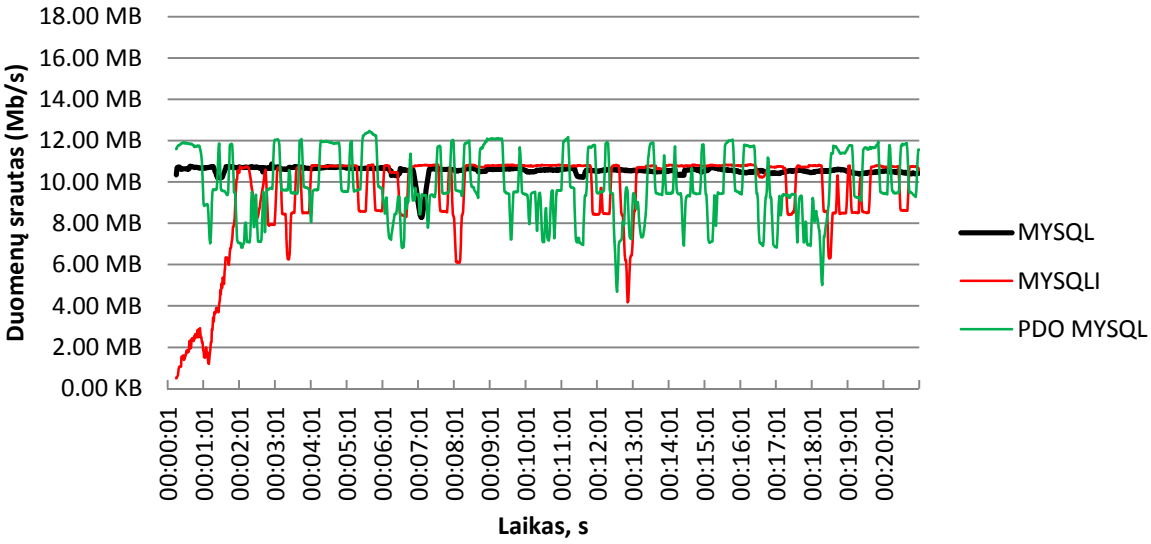
20 pav. Transakcijų kiekis su *MySQL*, naudojant paprastąją užklausą

Toliau eksperimentas su Mysql duomenų baze vykdomas su tomis pačiomis sąsajomis, mysql, mysqli ir PDO mysql, tik šį kartą vietoje paprastos užklausos yra naudojama vidinė Mysql procedūra (žr. 4 priede, 42 pav.) identiškai gražinanti tą patį duomenų modelį. Šiame eksperimente daugiausiai įvykdė užklausų su mysql sąsaja vidutiniškai 85,6 tūkst. per 21 minutę. Su mysqli sąsaja įvykdytą beveik 8,5 tūkst. užklausų mažiau nei su mysql sąsaja – 77 tūkst., o su PDO buvo įvykdoma vidutiniškai 80 tūkst. užklausų. Visas įvykdytų užklausų kitimas laike pavaizduotas 21 pav. grafike.



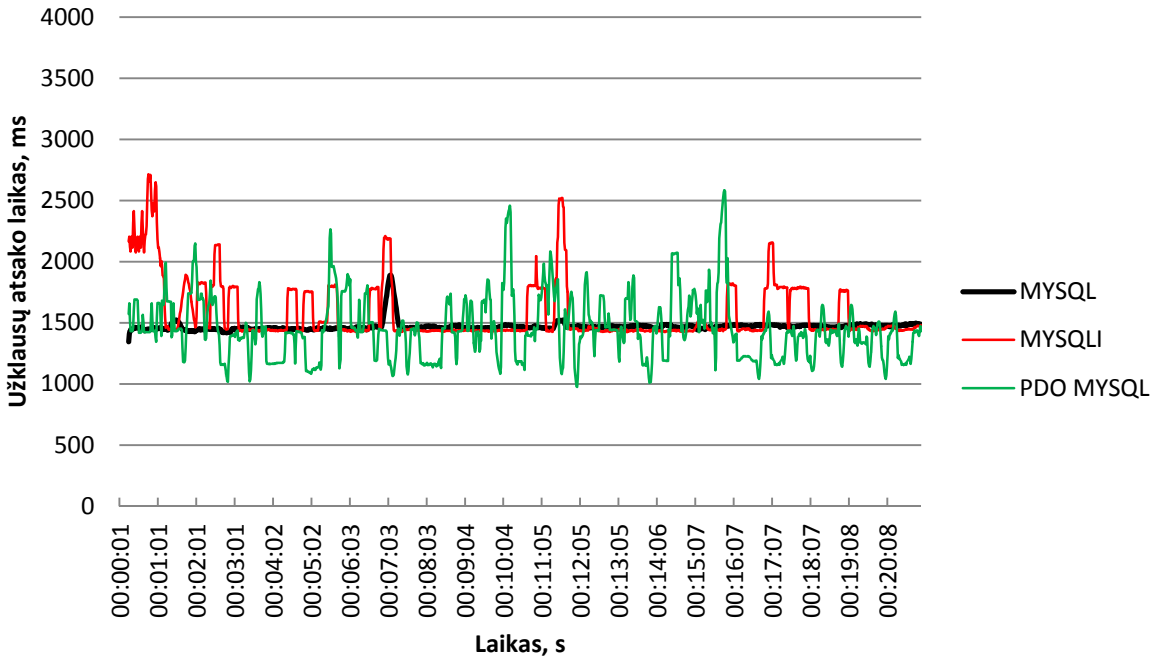
21 pav. Įvykdytų užklausų kiekis su *MySQL*, naudojant vidinę procedūrą

Duomenų srautas kintantis eksperimento laike matomas 22 pav. grafike. Su mysql sąsaja duomenų srautas vykdant eksperimentus buvo beveik nekintantis ir atlikus skaičiavimus pagal (1) formulę, vidutinė reikšmė – 10,54 Mb/s. Apskaičiavus su mysqli sąsaja duomenų srautas gautas – 9,48 Mb/s, o su PDO mysql sąsaja vidutiniškas duomenų srautas – 9,85 Mb/s.



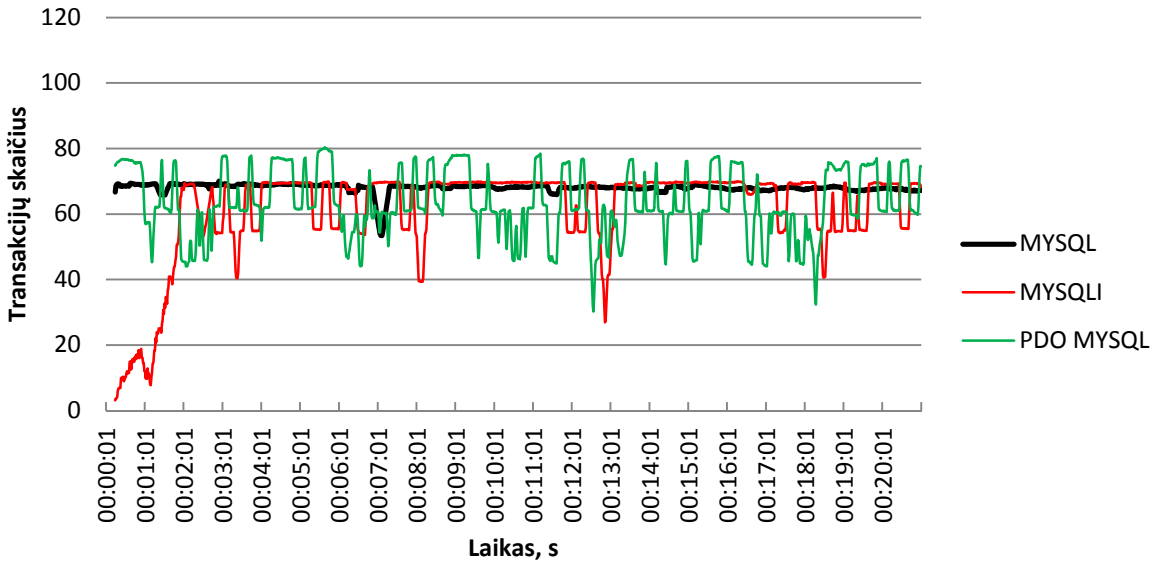
22 pav. Duomenų srautas su *MySQL*, naudojant vidinę procedūrą

Taip pat apskaičiuojame užklausų atsakų laikus pagal (2) formulę. Pagal užklausų atsako laikus geriausią rezultatą pasiekė mysql sąsaja kurios atsako laikas yra 1,47 sekundės. Mysqli sąsajos atsako laikas į užklausą vidutiniškai yra 1,63 sekundės, o su PDO mysql sąsaja 1,57 sekundės. Visą eksperimento metu kitusį užklausų atsakų laikas pavaizduotas 23 pav. grafike.



23 pav. Užklausų atsako laikas su *MySQL*, naudojant vidinę procedūrą

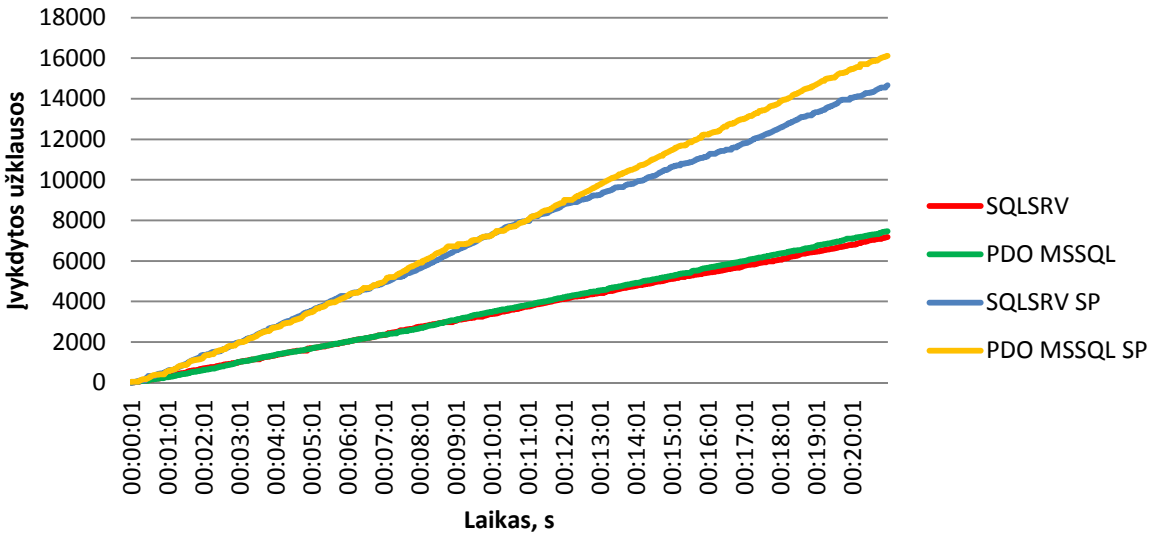
Galiausiai apskaičiuojame vidutines transakcijų per sekundę reikšmes pagal (3) formulę. Didžiausią transakcijų kiekį per sekundę pasiekė mysql sąsaja su 67,93 transakcijos per sekundę greičiu. Mysqli sąsajos greitis siekė 61,11, o PDO mysql sąsajos – 63,49 transakcijos per sekundę.



24 pav. Transakcijų kiekis su *MySQL*, naudojant vidinę procedūrą

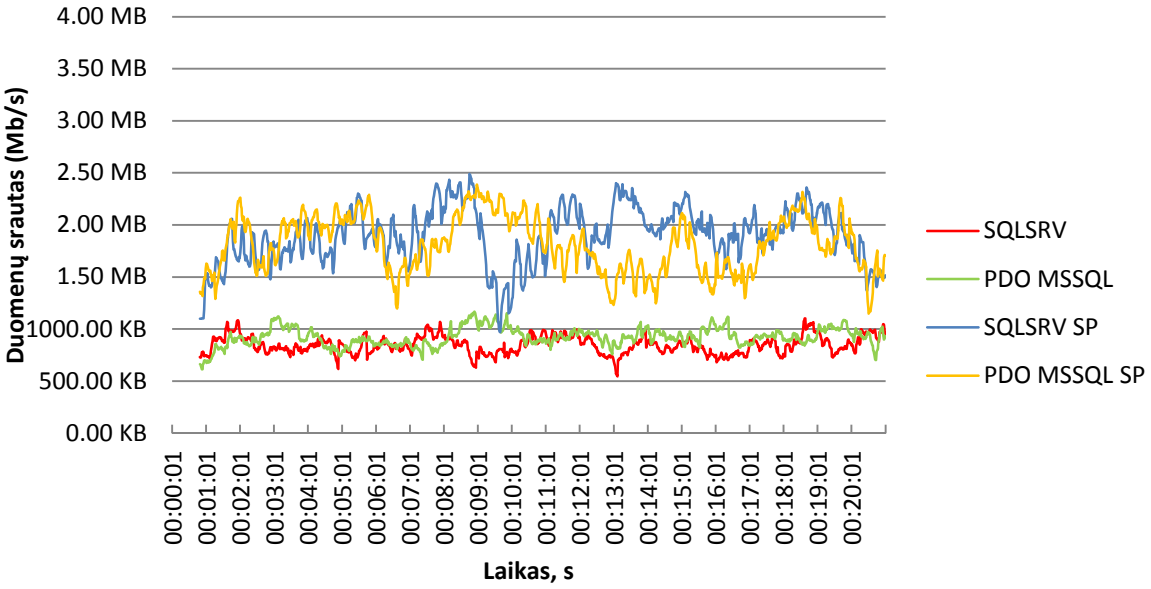
Atlikus visus eksperimentus su MySQL duomenų baze, ji yra išjungiama ir įjungiama MsSQL duomenų bazės valdymo sistema. Eksperimentas atliekamas su paprasta užklausa (žr. 2 priede, 39 pav.) ir su vidine procedūra (žr. 4 priede, 43 pav.).

Vykdant eksperimentus gauta, kad paprastos užklausos metu rezultatai nežymiai skyrės.



25 pav. Įvykdytų užklausų kiekis su *MsSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

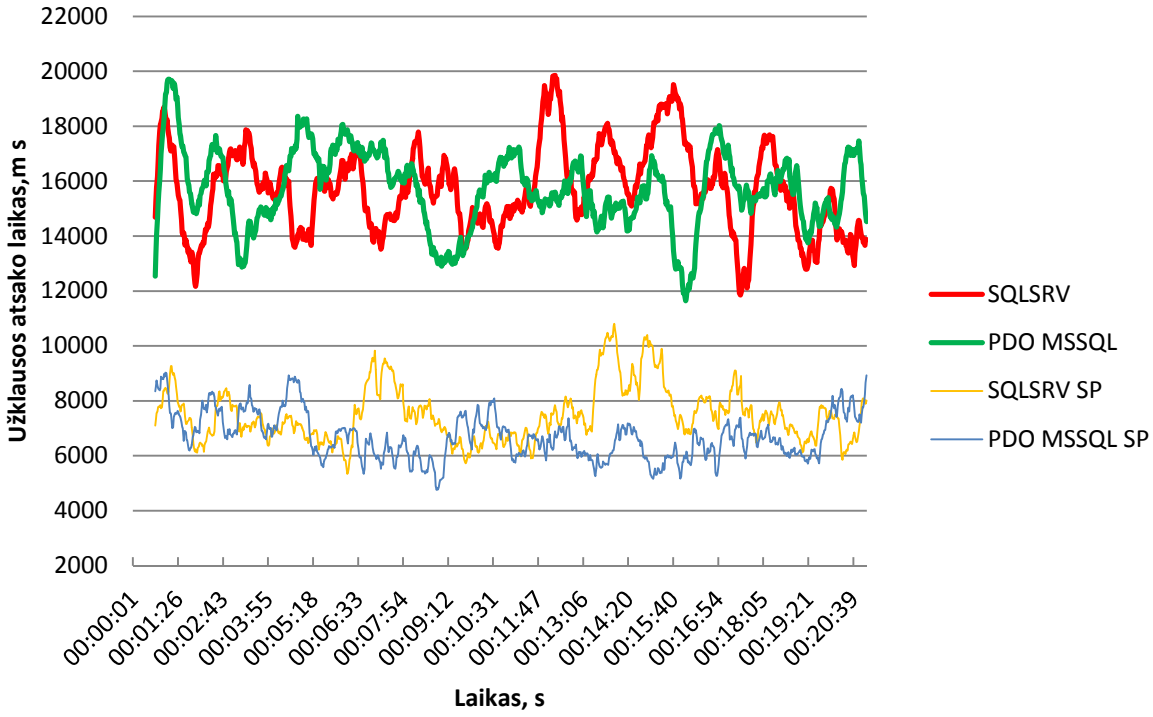
Iš grafiko (25 pav.) matyti, kad su sqlsrv sąsaja įvykdyta 7175 užklausos, o su PDO mssql sąsaja – 7467 užklausos per tą patį 21 minučių eksperimento vykdymo laiką. Tačiau vykdant eksperimentą su vidine procedūra gautuose rezultatuose matomas žymus įvykdytų užklausų didėjimas išaugęs beveik du kartus. Su sqlsrv sąsaja buvo įvykdyta vidutiniškai 14460 užklausų, o su PDO mssql sąsaja net 16110 užklausų. Nors ir įvykdytų užklausų kiekis žymiai skiriasi nuo MySQL duomenų bazės gautų rezultatų, tačiau tarp paprastų užklausų ir vidinių procedūros užklausos įvykdytų santykis žymiai didesnis.



26 pav. Duomenų srautas su *MsSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

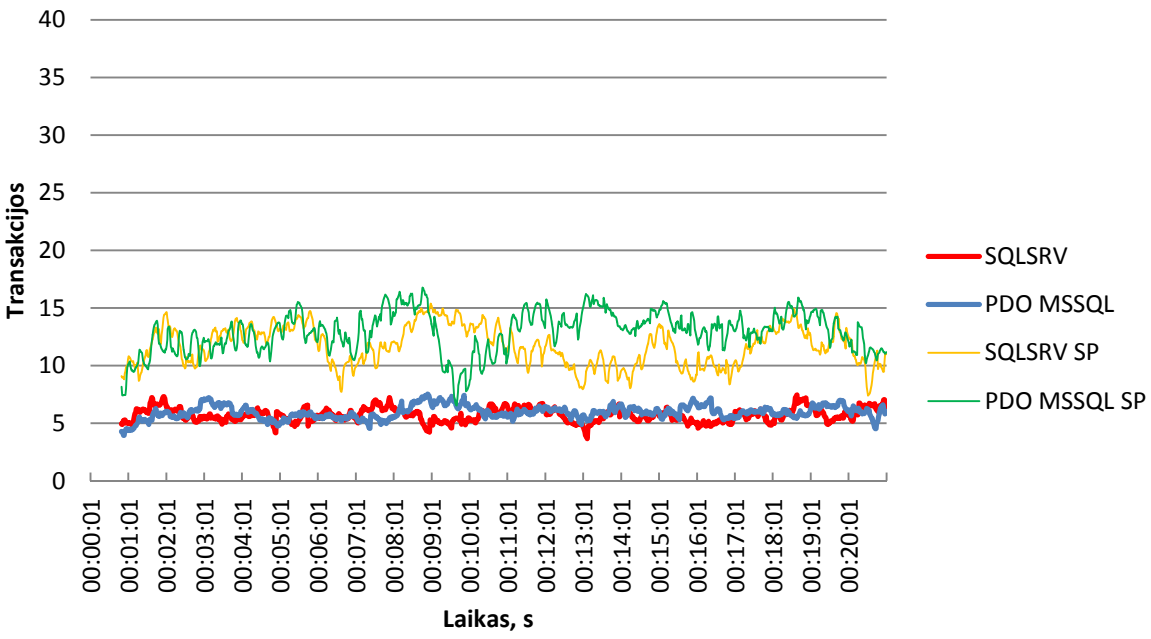
Duomenų srautas pavaizduotas 26 pav. grafike. Rezultatai yra maži, nes ir įvykdytų užklausų kiekis yra mažas. Naudojant (1) formulę apskaičiuojame vidutines duomenų srauto reikšmes. Vykdant paprastą užklausą duomenų srautas su sqlsrv sąsaja yra 0,88 Mb/s, o su PDO mssql – 0,92 Mb/s. Užklausos įvykdytos per vidinę procedūrą duomenų srautas yra su sqlsrv sąsaja yra 1,80 Mb/s, o su PDO mssql – 1,98 Mb/s.

Užklausų atsakų laikai eksperimento metu, naudojant vidinę procedūrą, yra beveik du kart mažesni, nei su paprastomis užklausos. Apskaičiavus vidutinius užklausų laikus gauta, kad su sqlsrv sąsaja vidutinis užklausų atsako laikas yra 17,5 sekundės, o naudojant vidinę procedūrą atsako laikas tik 8,5 sekundės. Naudojant PDO mssql sąsają gauta: su vidinės procedūros užklausa atsako laikas yra vidutiniškai lygus 7,8 sekundės, o naudojant paprastą užklausą – 16,8 sekundės. Atsakų į užklausas grafikas pavaizduotas 27 pav. grafike.



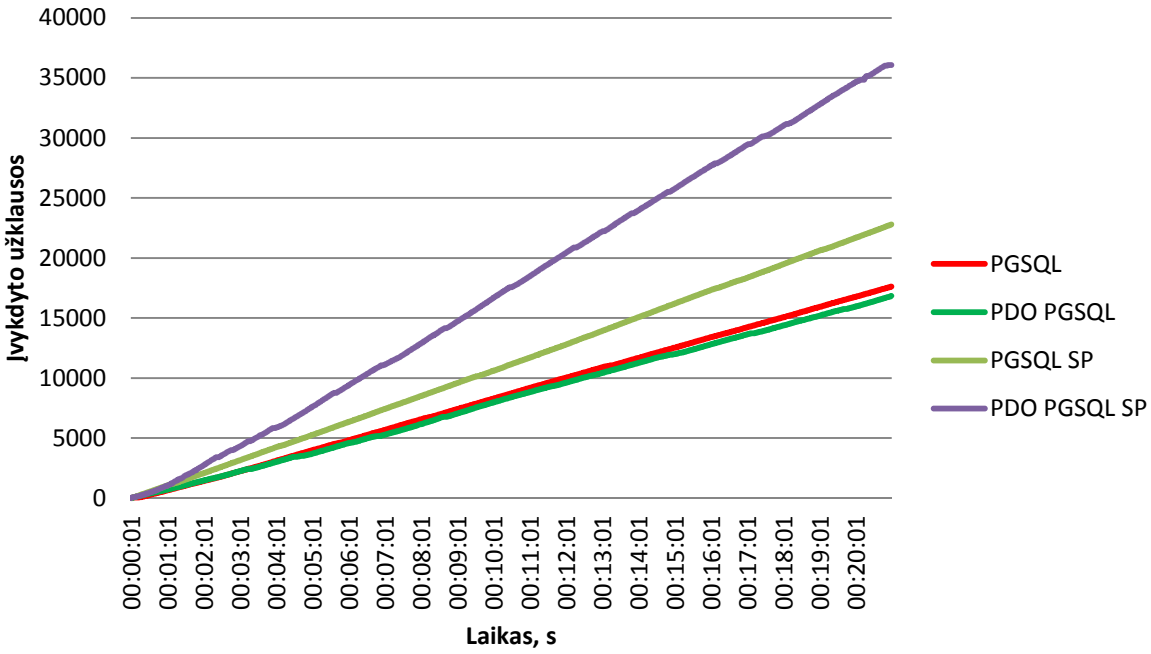
27 pav. Užklausų atsako laikas su *MsSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Paskutinis parametras matuotas atliekant eksperimentą su MsSQL duomenų baze yra transakcijų kiekis per sekundę. Paprastos užklausos eksperimento metu užklausų kiekis, apskaičiuotas pagal (3) formulę skyrėsi nežymiai. Su sqlsrv sąsaja vidutiniškai įvykdoma 5,69 transakcijos per sekundę, o su PDO mssql sąsaja – 5,92 transakcijos per sekundę. Vykdant eksperimentą su vidinėm užklausomis vidutinis transakcijų greitis naudojant sqlsrv sąsaja yra 11,63 transakcijos per sekundę, o su PDO mssql – 12,78 transakcijos per sekundę. Visas transakcijų kitimas pavaizduotas 28 pav. grafike.



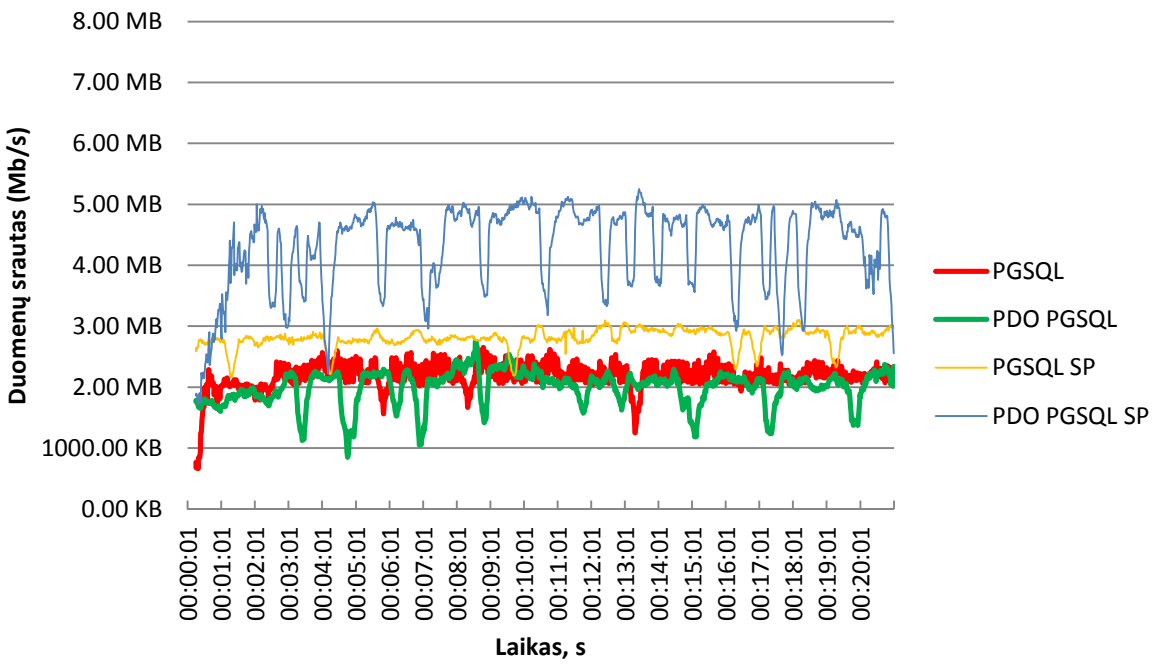
28 pav. Transakcijų kiekis su *MsSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Trečia eksperimente naudojama duomenų bazės valdymo sistema yra PostgreSQL. Atlikus eksperimentus nustatyta, kad rezultatai naudojant paprastą užklausą skiriasi nežymiai, mažiau nei tūkstantis užklausų skirtumu pgsql sąsaja aplenkė PDO pgsql sąsają. Vidutiniškai su pgsql sąsaja įvykdyta 17,6 tūkst. užklausų, o su PDO pgsql – 16,9 tūkst. užklausų. Naudojant vidinę procedūrą nustatyta, kad su pgsql sąsaja vidutiniškai įvykdyta 22,8 tūkst. užklausų, o su PDO pgsql net 36 tūkst. užklausų. Užklausų įvykdymo eksperimento metu grafikas pavaizduotas 29 paveiksle.



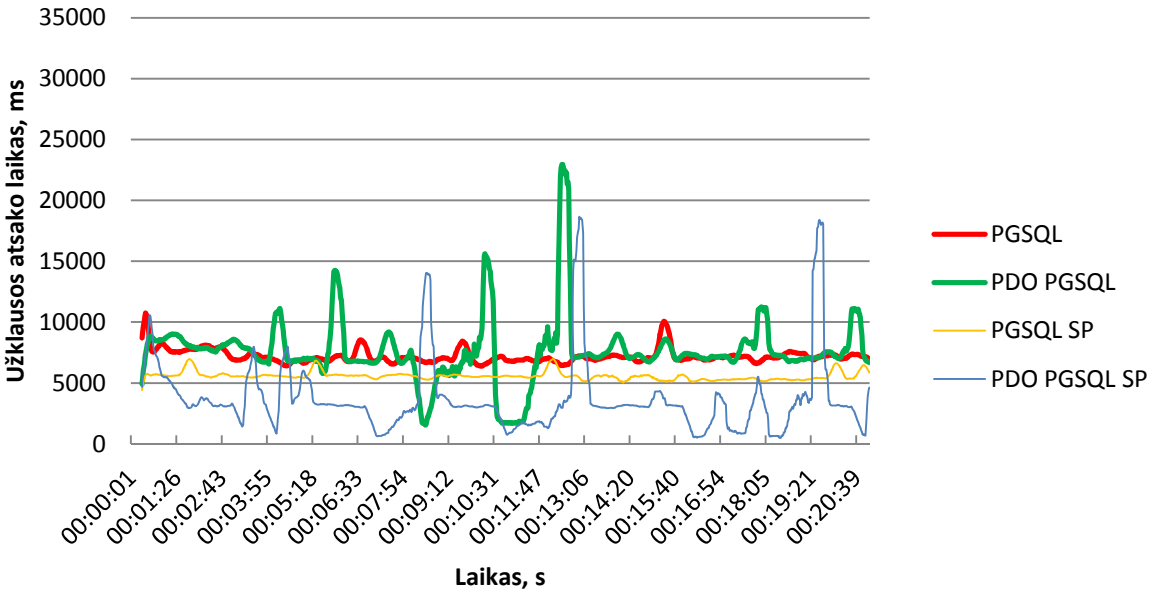
29 pav. Įvykdytų užklausų kiekis su *PostgreSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Duomenų srauto 30 pav. grafike matyti labai dideli amplitudės svyravimai atliekant eksperimentus su PDO pgsql sąsaja. Tai įtakojo didelis Apache serverio apkrovimas, beveik viso eksperimento metu serverio procesorius buvo naudojamas šimtu procentu. Iš (1) formulės apskaičiuojame vidutinius duomenų srauto reikšmes. Gauta, kad su pgsql sąsaja naudojant paprastąją užklausą duomenų perdavimo sparta yra 2,17 Mb/s, o su PDO pgsql – 2,07 Mb/s. Eksperimente naudojant vidinę procedūrą gautos reikšmės: su pgsql sąsaja yra 2,81 Mb/s, o su PDO pgsql – 4,45 Mb/s.



30 pav. Duomenų srautas su *PostgreSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Užklausų atsakų laikai eksperimente su PostgreSQL sąsajomis yra gerokai mažesnis nei atliekant eksperimentą su MsSQL duomenų baze. Atsakų laikai eksperimento metu matyti 31 pav. grafike. Vidutines atsakų į užklausas reikšmes apskaičiuojame pagal (2) formulę. Gauta, kad su pgsql sąsaja naudojant paprastą užklausą vidutinis atsako laikas yra 7,1 sekundės, o su PDO pgsql sąsaja – 7,4 sekundės. Naudojant vidinę procedūrą atsakų laikai yra sumažėję ir su pgsql sąsaja vidutinis atsako laikas yra 5,5 sekundės, o su PDO pgsql atsako laikas sumažėja iki 3,4 sekundės.

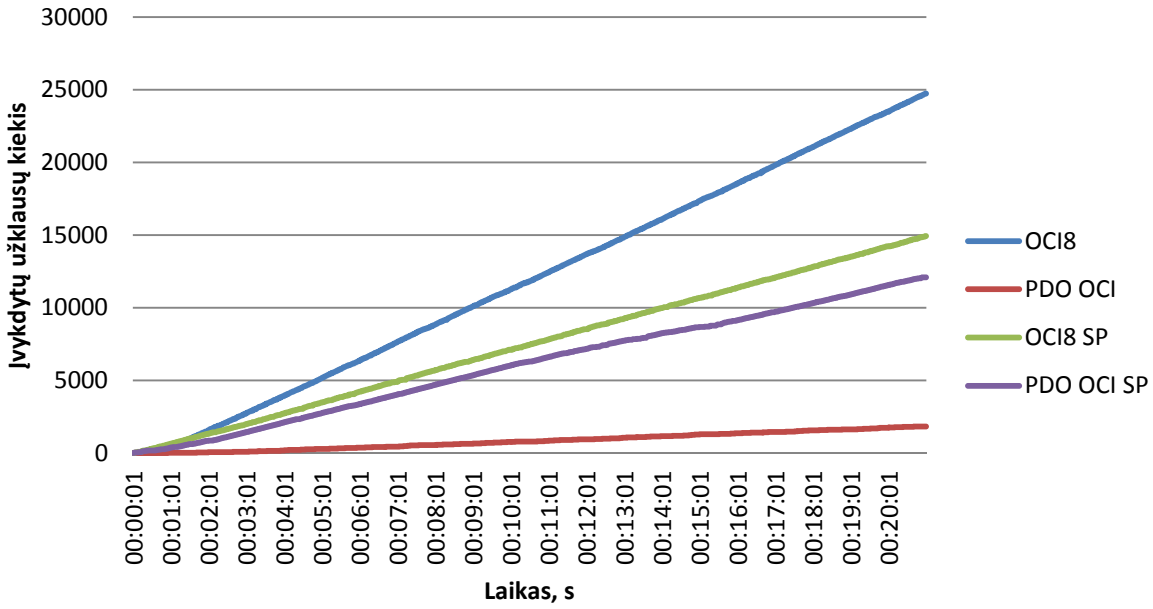


31 pav. Užklausų atsako laikas su *PostgreSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Galiausiai eksperimente su PostgreSQL duomenų baze išnagrinėjame transakcijų kitimą eksperimento metu. Visas eksperimento metu transakcijų kitimas pavaizduotas 32 pav. grafike. Didžiausią spartą pasiekė užklausos su vidine procedūra ir naudojant PDO pgsql sąsają, kurios vidutinis rezultatas apskaičiavus pagal (3) formulę yra 28,65 transakcijos per sekundę. Antrą pagal spartą rezultatą parodė eksperimentas su pgsql sąsaja naudojant vidinę procedūrą ir siekė 18,09 transakcijos per sekundę. O naudojant paprastąją užklausą ir pgsql sąsaja gautas 13,97 transakcijų per sekundę rezultatas. Su PDO pgsql rezultatas yra nežymiai mažesnis – 13,36 transakcijos per sekundę.

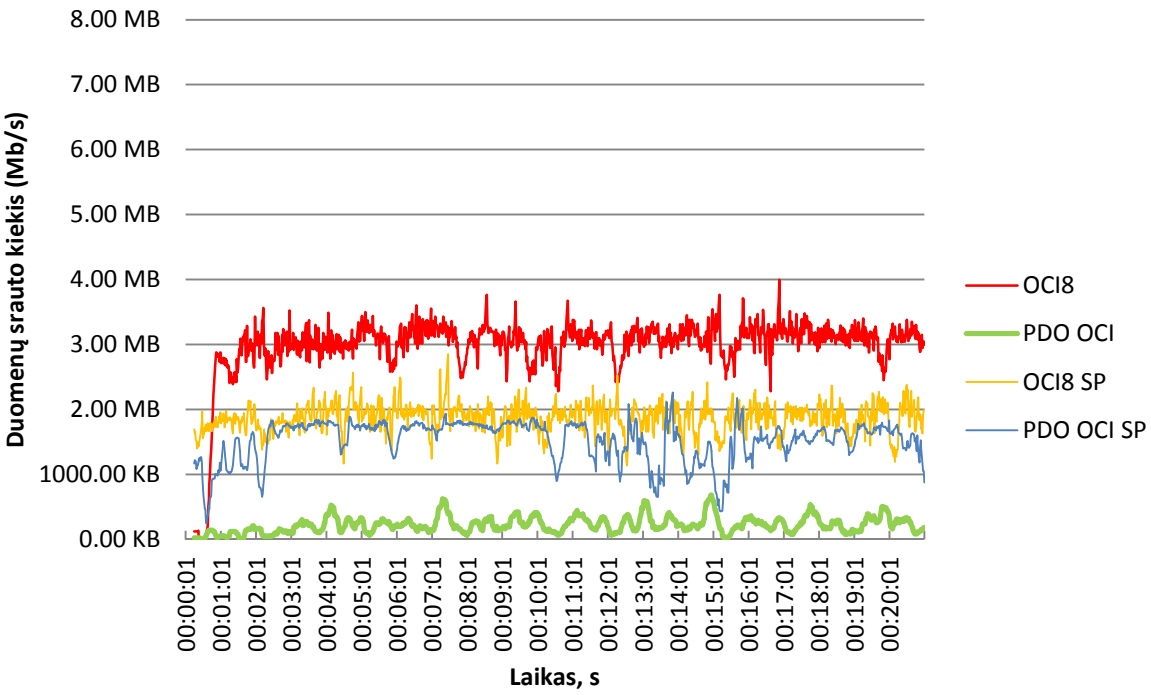
32 pav. Transakcijų kiekis su *PostgreSQL*, naudojant paprastąją užklausą ir vidinę procedūrą

Paskutinė eksperimente naudota duomenų bazė – Oracle. Eksperimento rezultatai parodė, kad didžiausi užklausų įvykdymo skirtumai naudojant oci8 sąsają, bei PDO oci sąsają. Naudojant PDO oci sąsają ir atliekant paprastąją užklausą eksperimento metu per 21 min. įvykdoma tik 1850 užklausų, kai tuo tarpu su oci8 sąsaja įvykdyta 24,8 tūkst. užklausų per tą patį laiką. Vykdant eksperimentą su vidine procedūra rezultatų skirtumo santykis yra žymiai mažesnis. Taigi, su oci8 sąsaja ir naudojant vidinę procedūrą vidutiniškai įvykdytą 15 tūkst. užklausų, o su PDO oci – 12 tūkst. užklausų. Užklausų įvykdymo grafikas pavaizduotas 33 paveiksle. Taip pat kaip ir eksperimente naudojant PostgreSQL duomenų bazę procesoriaus apkrovimas buvo pilnai išnaudojamas.



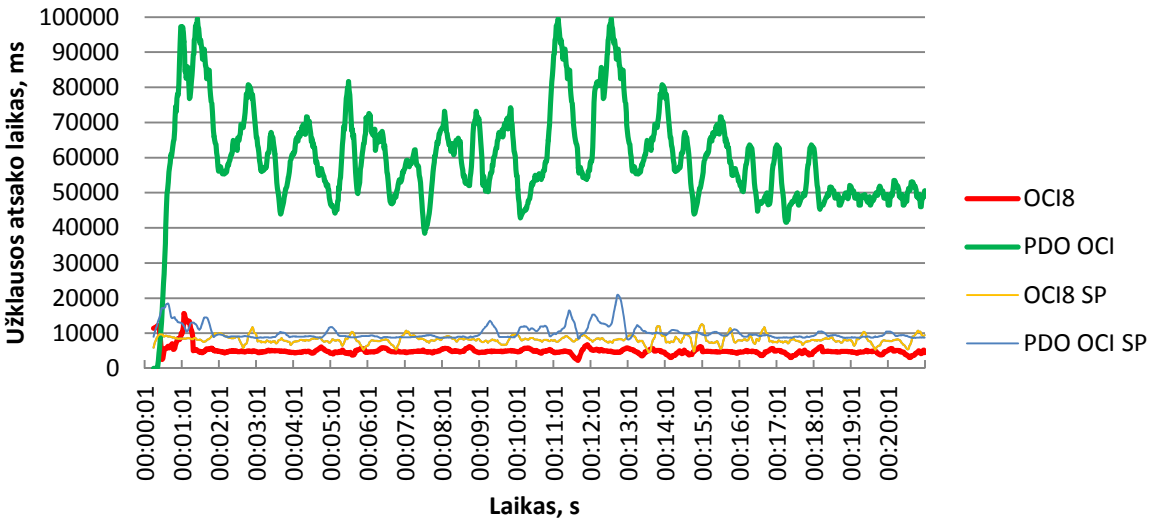
33 pav. Įvykdytų užklausų kiekis su *Oracle*, naudojant paprastąją užklausą ir vidinę procedūrą

Duomenų srautas eksperimento metu pavaizduotas 34 pav. grafike. Su oci8 sąsaja duomenų srautas yra didžiausias ir su paprastąja užklausa ir su vidine procedūra. Vidutiniai duomenų srauto rezultatus apskaičiuojame pagal (1) formulę. Gauta, kad su oci8 sąsaja ir naudojant paprastąją užklausą duomenų vidutinis srautas yra 3,05 Mb/s, o su PDO oci tik 0,23 Mb/s. Eksperimente su vidinėmis procedūromis ir oci8 sąsaja duomenų sparta yra 1,85 Mb/s, o su PDO oci – 1,45 Mb/s.



34 pav. Duomenų srautas su *Oracle*, naudojant paprastąją užklausą vidinę procedūrą

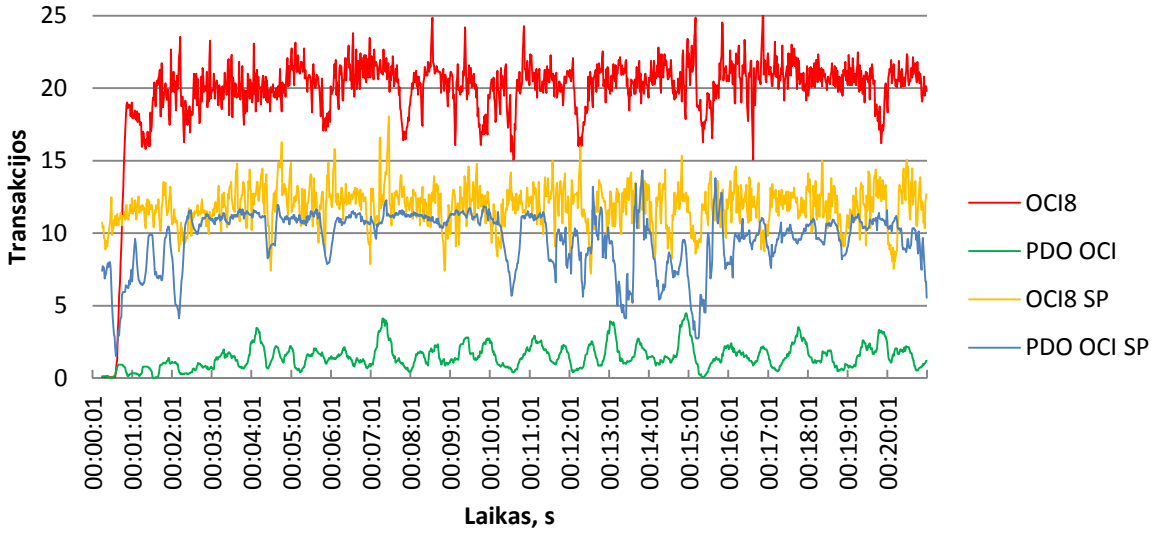
Užklausų atsakų laikų žymiai skyrėsi. Didžiausias užklausų atsako laikas apskaičiavus pagal (2) formulę yra su PDO oci sąsaja naudojant paprastąją užklausą ir vidutinis atsako laikas yra 68 sekundės. Tuo tarpu naudojant oci8 sąsają ir naudojant paprastąją užklausą vidutinis atsako laikas yra 5,1 sekundės. Naudojant vidinę procedūrą vidutinis atsako laikas su oci8 sąsaja yra 8,4 sekundės, o su PDO oci sąsaja – 10,4 sekundės. Eksperimento metu kitęs užklausų atsakų laikas pavaizduotas 35 paveiksle.



35 pav. Užklausų atsako laikas su *Oracle*, naudojant paprastąją užklausą ir vidinę procedūrą

Transakcijų greitaveikos matavime taip pat blogiausią rezultatą parodė PDO oci sąsaja naudojant paprastąja užklausą ir vidutiniškai atliekanti 1,46 transakcijas per sekundę, apskaičiavus pagal (3) formulę. Didžiausią transakcijų spartą pasiekė oci8 sąsajos ir

paprastosios užklausos eksperimentas, kurio metu yra įvykdyta 19,46 transakcijos per sekundę. Su vidinėmis procedūromis eksperimentai ir oci8 sąsaja transakcijų skaičius yra 11,91 transakcijos per sekundę, o su PDO oci sąsaja – 9,6 transakcijos per sekundę. Transakcijų kitimo eksperimento metu grafikas pavaizduotas 36 paveiksle.



36 pav. Transakcijų kiekis su *Oracle*, naudojant paprastąją užklausą ir vidinę procedūrą

3.2.4 Analizė susijusi su eksperimento tikslu ir gautų rezultatų apibendrinimas

Atlikus eksperimentus su MySQL, MsSQL, PostgreSQL ir Oracle duomenų bazėmis apibendrinti rezultatai surašyti 6 lentelėje.

6 Lentelė. Apibendrinti su duomenų išrinkimu eksperimento rezultatai

	MySQL			**MsSQL**		**PostgreSQL**		**Oracle**	
Rezultato tipas *	mysql	mysqli	PDO mysql	sqlsrv	PDO mssql	pgsql	PDO pgsql	oci	PDO oci
1	12,07	9,73	11,73	0,88	0,92	2,17	2,07	3,05	0,23
2	10,54	9,48	9,85	1,80	1,98	2,81	4,45	1,85	1,45
3	98000	79000	95200	7175	7467	17610	16840	24750	1850
4	85600	77000	80000	14660	16110	22800	36100	15000	12100
5	1,28	1,59	1,32	17,5	16,8	7,1	7,4	5,1	68,1
6	1,47	1,63	1,57	8,5	7,8	5,5	3,4	8,4	10,4
7	77,7	62,7	76,6	5,69	5,92	13,97	13,36	19,64	1,46
8	67,93	61,11	63,49	11,63	12,78	18,09	28,65	11,91	9,60

* Rezultatų tipo lauko reikšmės: 1 – Baitų kiekis per sekundę (matavimo vienetai - Mb/s); 2 – Baitų kiekis per sekundę naudojant vidinę procedūrą (Mb/s); 3 – Įvykdytos užklausos (Užklausos); 4 – Įvykdytos užklausos naudojant vidinę procedūrą (Užklausos); 5 – Užklausos atsakų laikas (s); 6 – Užklausos atsakų laikas naudojant vidinę procedūrą (s); 7 – Transakcijų kiekis per sekundę (Transakcijos/sekundę); 8 – Transakcijų kiekis per sekundę naudojant vidinę procedūrą (Transakcijos/sekundę).

Pasinaudojus gautus rezultatus iš 6 lentelės sudarome PHP plėtinių tarpusavio palyginimo lentelę (žr. 7 lentelė), kurioje kiekvieno PHP plėtinio palyginamajame rezultate pirmoje eilutėje yra santykis tarp plėtinių naudojant paprastąją užklausą, o antroji eilutė – tai santykinis palyginimas užklausose naudojant vidinę procedūrą. Žalios lentelės reikšmės laukas reiškia geresnį rezultatą su lyginamąja sąsaja, raudonas – blogesnį rezultatą, o geltonas – su vienu iš rezultatų geresnį ir blogesnį rezultatų įvertinimo procentą.

7 Lentelė. PHP duomenų bazių plėtinių palyginimo lentelė pagal duomenų išrinkimo eksperimentą

	mysql	mysqli	PDO mysql	sqlsrv	PDO mssql	pgsql	PDO pgsql	oci	PDO oci
mysql		24% 11%	2% 7%	1265% 483%	1212% 431%	456% 275%	481% 137%	295% 470%	5197% 607%
mysqli	-24% -11%		-20,5% -3,8%	1001,1% 425,2%	1001,0% 377,9%	348,6% 237,7%	369,1% 113,2%	219,2% 413,3%	4170,2% 536,3%
PDO mysql	-2% -7%	20,5% 3,8%		*1226,8%* *445,7%*	1174,9% 396,5%	440,6% 250,8%	465,3% 121,6%	284,6% 433,3%	5045,9% 561,1%
sqlsrv	-1265% -483%	-1000,1% -425,2%	-1226,8% -445,7%		-4% -9,8%	-145,4% -55,5%	-134,7% -146,2%	-244,9% -2,3%	287,8% 21,1%
PDO mssql	-1212% -431%	-1001,0% -377,9%	-1174,9% -396,5%	4% 9,8%		-135,8% -41,5%	-125,5% -124,0%	-231,4% 7,4%	303,6% 33,1%
pgsql	-456% -275%	-348,6% -237,7%	-440,6% -250,8%	145,4% 55,5%	135,8% 41,5%		4,5% -58,3%	-40,5% 52,0%	851,8% 88,4%
PDO pgsql	-481% -137%	-369,1% -113,2%	-465,3% -121,6%	134,7% 146,2%	125,5% 124,0%	-4,5% 58,3%		-46,9% 198,3%	810,2% 198,3%
oci	-295% -470%	-219,2% -413,3%	-284,6% -433,3%	244,9% 2,3%	231,4% -7,4%	40,5% -52,0%	46,9% -198,3%		1237,8% 23,9%
PDO oci	-5197% -607%	-4170,2% -536,3%	-5045,9% -561,1%	-287,8% -21,1%	-303,6% -33,1%	-851,8% -88,4%	-810,2% -198,3%	-1237,8% -23,9%	

Atlikus visus eksperimentus nustatyta, kad geriausi rezultatai yra su MySQL duomenų bazės PHP plėtiniais. Geriausias rezultatas yra su mysql sąsaja, tačiau nors ir 2% geresnis rezultatas vykdant su paprastąja užklausą bei 7% greitesnis su vidine procedūrą, nei su PDO mysql sąsaja, visgi jau analizės dalyje paminėta, kad mysql sąsaja nuo 5.5 PHP versijos yra laikoma pasenusia ir nepalaiko didesnių duomenų bazės ypatumų kaip transakcijos ir vidinių procedūrų kursorių, todėl vertėtų kuriant naujus projektus naudoti PDO mysql sąsają.

Jeigu projekte nenaudojama MySQL duomenų bazės, o ieškoma alternatyvų, tai antra pagal geriausią rezultatą parodžiusių duomenų bazių yra PostgreSQL. Su PostgreSQL duomenų baze vykdytame eksperimente pgsql ir PDO pgsql sąsajos skirtumai: su paprastąja užklausa yra greitesnė pgsql sąsaja 4,5% nei PDO pgsql, bet naudojant vidinę procedūrą didesnį rezultatą parodė PDO pgsql, kurios įvykdymo sparta yra 58,3% greitesnė nei pgsql.

Jeigu projekto uždaviniuose nenaudotume nei MySQL, nei PostgreSQL tai iš likusių tyrime nagrinėtų duomenų bazių sistemų tinkamiausias yra Oracle. Eksperimente su Oracle ir PHP sąsajomis, prastesnį rezultatą parodė PDO oci sąsaja. Kaip ir analizės dalyje buvo minėta kad PDO oci sąsaja yra eksperimentinė ir nepatariama naudot. Tyrimo rezultatuose matome, kad su paprastąja užklausa PDO oci sąsaja yra lėtesnė: už MySQL sąsaja maždaug apie 50 kartų, už PostgreSQL sąsajas apie 10 kartų, o už MsSQL sąsaja 4 kartus lėtesnė. Nors

skirtumas su oci8 sąsaja ir PDO oci sąsaja net 13 kartų skiriasi oci8 sąsajos naudai. Naudojant oci sąsaja tyrime su paprastąja užklausą rezultatai nusileido tik MySQL sąsajoms, o likusias duomenų bazių sąsajas aplenkė.

Paskutinė duomenų bazės sistema yra MsSQL. Rezultatai rodo, kad MsSQL sąsajos visuose eksperimentuose kaip ir Oracle PDO oci sąsaja yra prasčiausi. Su MsSQL duomenų bazės valdymo sistema PHP geriausiai bendravo vidinių procedūrų pagalba daugiau nei du kartus greičiau nei su praprastosiomis užklausomis.

3.3.Eksperimentas su duomenų įterpimu

3.3.1 Eksperimento tikslas

Eksperimentas atliekamas tam, kad ištirti kiek įterpimo užklausų įvykdoma eksperimento metu kiekį ir naudojant skirtingus PHP ir duomenų bazių susijungimo plėtinius.

3.3.2 Eksperimento atlikimo tvarka

Eksperimento tikslui įgyvendinti yra būtina apskaičiuoti:

- Įvykdytų kreipinių kiekį;
- Kreipinių įvykdymo laiką;
- Įterpimo kreipinių įvykdymo spartą.

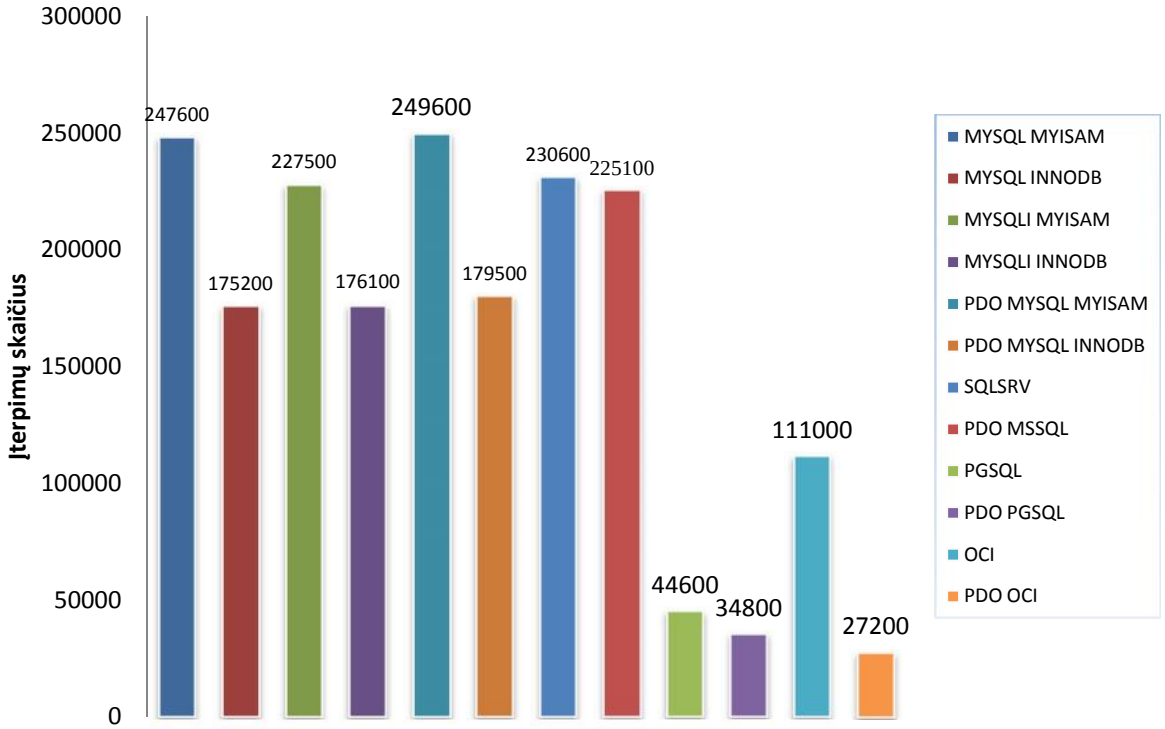
Eksperimento atlikimo tvarka:

- Įjungiame reikalingą eksperimentui duomenų bazės valdymo sistemą;
- Užduotą eksperimentą vykdome LoadUI programa 21 minutę;
- Eksperimentą kartojame dešimt kartų;
- Gautus rezultatus eksportuojame į Excel bylas;
- Excel bylas suformatuojame atitinkamomis formulėmis ir sukuriame eksperimento vykdymo grafikus;
- Formulėmis (2-3) apskaičiuojame rezultatus.

3.3.3 Eksperimento rezultatai

Įvykdžius visus įterpimo eksperimentus gautieji rezultatai pavaizduoti 37 paveiksle. Daugiausiai įterpimo kreipinių įvykdė PDO mysql sąsaja ir įterpė vidutiniškai apie 250 tūkst. įrašų į duomenų bazę su MYISAM varikliu. Nedaug nuo PDO mysql sąsajos atsiliko ir mysql sąsaja kuri eksperimentų metu vidutiniškai įterpė 248 tūkst. įrašų į MySQL duomenų bazės lentelę su MYISAM varikliu. Treti pagal spartą yra įterpimai į MsSQL duomenų bazę, su sqlsrv sąsaja įvykdyta 231 tūkst. užklausų, o su PDO mssql 225 tūkst. užklausų. Mysqli

sąsajos įterpimai beveik panašūs kaip ir įterpimai į MsSQL duomenų bazę. Tačiau įterpimai į MySQL duomenų bazę naudojant InnoDB variklį žymiai skyrėsi nuo rezultatų naudojant MyISAM ir sudarė 175-180 tūkst. įterpimų per 21 minutę.



37 pav. Įterpimo operacijų pasiskirstymas

Daugiau nei du kartus, lyginant su pirmomis sąsajomis, mažiau įterpimų nuo lyderių buvo įterpta su Oracle oci8 sąsaja ir siekė 111 tūkst. įterpimų. Su PostgreSQL sąsajomis įterpta net apie 4-5 kartus mažiau įrašų į duomenų bazę per ta patį eksperimento laiką ir siekė 45 tūkst. įrašų naudojant pgsql sąsają ir apie 35 tūkst. įrašų naudojant PDO pgsql sąsają. Visų prasčiausią rezultatą vėl parodė PDO oci sąsaja kur eksperimento metu vidutiniškai buvo įterpiama vos daugiau nei 27 tūkst. įrašų per 21 minutę. Ši sąsaja vėl parodė blogiausią rezultatą kaip ir eksperimente su duomenų išrinkimu.

Įterpimo operacijų užklausos atsakų laikus apskaičiuojame pagal (2) formulę ir gauname, kad: su mysql (MYISAM variklis) vidutinis užklausos įterpimo laikas – 0,508 s, mysql (INNODB) – 0,719 s, mysqli (MYISAM) – 0,553 s, mysqli (INNODB) – 0,715 s, PDO mysql (MYISAM) – 0,504 s, PDO mysql (INNODB) – 0,701 s, sqlsrv – 0,546 s, PDO mssql – 0,559 s, pgsql – 2,825 s, PDO pgsql – 3,620 s, oci8 – 1,135 s, PDO oci – 4,632 s.

Taip pat dar apskaičiuojame kiek vidutiniškai įterpimų per sekundę yra atlikta eksperimentinio tyrimo metu. Pasinaudojant (3) formulę gauname: su mysql (MYISAM variklis) 196,60 įterpimų per sekundę, su mysql (INNODB) – 139,04 įterpimų per sekundę, mysqli (MYISAM) – 180,55 įterpimų per sekundę, su mysqli (INNODB) – 139,76 įterpimų per sekundę, su PDO mysql (MYISAM) – 198,09 įterpimų per sekundę, su PDO mysql

(INNODB) – 142,46 įterpimų per sekundę, su sqlsrv – 183,01 įterpimų per sekundę, su PDO mssql – 178,65 įterpimų per sekundę, su pgsql – 35,39 įterpimų per sekundę, su PDO pgsql – 27,62 įterpimų per sekundę, su oci8 – 88,09 įterpimų per sekundę, su PDO oci –21,58 įterpimų per sekundę.

3.3.4 Analizė susijusi su eksperimento tikslu ir gautų rezultatų apibendrinimas

Atlikus eksperimentą su duomenų įterpimų gavome rezultatus ir susisteminus pateikti kiekvieno PHP plėtinio procentinis santykis su kitais, kurie pavaizduoti 8 lentelėje. Žalios lentelės reikšmės laukas reiškia geresnį rezultatų santykį su lyginamąja sąsaja, raudonas – blogesnį santykį.

8 Lentelė. PHP plėtinių palyginimo lentelė pagal įterpimo eksperimento rezultatus

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
1		41%	9%	40%	-1%	38%	7%	10%	455%	611%	123%	810%
2	-41%		-30%	-0,5%	-42%	-2%	-32%	-29%	293%	403%	58%	544%
3	-9%	30%		29%	-10%	27%	-1%	-1%	410%	554%	105%	736%
4	-40%	0,5%	-29%		-42%	-2%	-31%	-28%	295%	406%	58%	547%
5	1%	42%	10%	42%		39%	8%	11%	460%	617%	125%	818%
6	-38%	2%	-27%	2%	-39%		-28%	-25%	302%	416%	62%	560%
7	-7%	32%	1%	31%	-8%	28%		2%	417%	563%	107%	748%
8	-10%	29%	1%	28%	-11%	25%	-2%		404%	547%	103%	728%
9	-455%	-293%	-410%	-295%	-460%	-302%	-417%	-404%		28%	-149%	64%
10	-611%	-403%	-554%	-406%	-617%	-416%	-563%	-547%	-28%		-219%	28%
11	-123%	-58%	-105%	-58%	-125%	-62%	-107%	-103%	149%	219%		308%
12	-810%	-544%	-736%	-547%	-818%	-560%	-748%	-728%	-64%	-28%	-308%	

8 lentelės stulpelių ir eilučių pavadinimai yra:

1 – įterpimai su Mysql (MYISAM variklis) sąsaja

2 – įterpimai su Mysql (InnoDB) sąsaja

3 – įterpimai su Mysqli (MYISAM) sąsaja

4 – įterpimai su Mysqli (InnoDB) sąsaja

5 – įterpimai su PDO mysql (MYISAM) sąsaja

6 – įterpimai su PDO mysql (InnoDB) sąsaja

7 – įterpimai su Sqlsrv sąsaja

8 – įterpimai su PDO mssql sąsaja

9 – įterpimai su Pgsql sąsaja

10 – įterpimai su PDO pgsql sąsaja

11 – įterpimai su Oci8 sąsaja

12 – įterpimai su PDO oci sąsaja

Iš rezultatų matyti, kad daugiausiai teigiamų įvertinimų įterpimo eksperimente gauta su PDO mysql sąsaja naudojant MYISAM lentelės variklį bei mažiau nei per procentą atsilikusi su mysql sąsaja ir taip pat naudojant MYISAM variklį. Tačiau naudojant su tomis pačiomis sąsajomis lentelę su InnoDB varikliu rezultatai yra prastesnis 40 procentų, nes kiekvieną kartą įterpiant įrašą perskaičiuojamos indeksų reikšmės.

Jeigu projekte naudosime MySQL duomenų bazės lenteles su InnoDB varikliais, tai greitesnės už jas yra MsSQL duomenų bazės sąsajos ir įterpimo jų greitis yra 30 procentų didesnis nei MySQL.

PostgreSQL duomenų bazės sąsajos įterpimo greičiai yra net 300 – 600 procentų prastesni nei su MySQL ar MsSQL ir beveik tris kart lėtesni nei su oci8 sąsaja. Pgsql sąsaja yra 28 procentais greitesnė nei PDO pgsql sąsaja.

Oracle duomenų bazės įterpimai su oci8 ir PDO oci skyrėsi per 308 procentus arba daugiau nei keturis kartus. Nuo MySQL ir MsSQL duomenų bazės sąsajų oci8 sąsaja 2 kartus yra lėtesnė, o už PostgreSQL beveik 2 kartus greitesnė.

3.4. Apibendrinančios eksperimentų išvados

Atlikus eksperimentinius tyrimus su duomenų išrinkimu ir įterpimu nustatyta, kad Windows Apache serveryje geriausias PHP duomenų bazės plėtinys yra PDO mysql.

Duomenų išrinkimo eksperimente nustatyta, kad greičiausiai duomenis gauti iš MySQL duomenų bazės naudojant mysql ir PDO mysql sąsajomis, o lėčiausiai su PDO oci sąsaja. Eksperimente naudojant vidinę procedūrą didžiausias santykis nei vykdant su paprastąją užklausa gautas vykdant su MsSQL duomenų baze ir gautas net trijų kartų pagreitėjimas naudojant vidines procedūras kreipiniuose. Taip pat nustatyta, kad su Oracle duomenų baze PDO oci naudojimas sumažina greitaveiką paprastosiose užklausose net trylika kartų, bet naudojant vidines procedūras kreipiniuose skirtumas yra tik 24 procentais geriau su oci8 sąsaja.

Duomenų įterpimo eksperimente geriausi rezultatai buvo su MySQL duomenų baze naudojant MYISAM lentelės variklį, tačiau naudojant InnoDB lentelės variklį rezultatai yra prastesni už MsSQL duomenų bazės sąsajas net 30 procentų. Toliau likusios PostgreSQL ir Oracle didelės spartos nepasiekė, tačiau su oci8 sąsaja rezultatai tik du kartus prastesni nei MySQL ar MsSQL.

Nustatyta, kad ir su MsSQL ir su PostgreSQL duomenų bazėmis geriau naudoti PDO bibliotekos plėtinius, o su Oracle – oci8 plėtinį. Galima teigti, kad PDO nors ir tinka visas duomenų bazėms susieti su PHP, tačiau Oracle atveju geriau PDO oci plėtinio nenaudoti.

IŠVADOS

1. Atlikus duomenų bazių valdymo sistemų analizę gauta, kad MySQL duomenų bazė su PHP turi daugiausia susijungimo sąsajų. Išanalizuota, kad žemesnėje nei PHP5 versijoje galimos tik dvi susijungimo sąsajos su MySQL ir PostgreSQL duomenų bazėmis, o aukštesnėje nei PHP5 versijoje yra plėtiniai visoms analizuojamoms duomenų bazėms. Su PHP PDO biblioteka galimi susijungimai su visomis duomenų bazėmis apžvelgtomis analizės metu.
2. Suprojektavus sistemą eksperimentams atlikti, buvo sukurtos keturios duomenų bazės, kurios buvo užpildytos septyniomis lentelėmis, susietomis ryšiais bei užpildytos virš 17 milijonų įrašų. Duomenų bazėse buvo sukurtos vidinės procedūros, skirtos duomenų išrinkimui. Kuriant duomenų bazės lenteles didesnio skirtumo tarp duomenų bazių neaptikta, išskyrus tai, kad duomenų tipo pavadinimų užrašymas nežymiai skyrėsi tarp skirtingų duomenų bazių.
3. Atlikus eksperimentinius tyrimus su duomenų išrinkimu ir įterpimu nustatyta, kad Windows Apache serveryje geriausias PHP duomenų bazės plėtinys yra PDO mysql. Nustatyta, kad su MsSQL duomenų baze tinkamiausia naudoti PDO mssql plėtinį, su PostgreSQL – PDO pgsql plėtinį, o su Oracle – oci8 plėtinį.

LITERATŪROS SĄRAŠAS

1. Usage statistics and market share of PHP for websites – [žiūrėta 2013-05-21]. Prieiga per internetą: <http://w3techs.com/technologies/details/pl-php/all/all>
2. Website Developing: Why Learn PHP Programming – [žiūrėta 2013-05-21]. Prieiga per internetą: <http://wikiwebpedia.com/2013/04/09/website-developing-learn-php-programming>
3. Google Trends – [žiūrėta 2013-05-09]. Prieiga per internetą: < http://bit.ly/193A48l >
4. MYSQL 5.5 Reference Manual – [žiūrėta 2013-05-21]. Prieiga per internetą: <http://dev.mysql.com/doc/refman/5.5/en/>
5. Sun Picks Up MySQL For $1 Billion; Open Source Is A Legitimate Business Model – [žiūrėta 2013-05-21]. Prieiga per internetą: <http://techcrunch.com/2008/01/16/sun-picks-up-mysql-for-1-billion-open-source-is-a-legitimate-business-model/>
6. PHP Database Extensions - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://www.php.net/manual/en/refs.database.php>
7. InnoDB Full-Text Search is in MySQL 5.6.4 - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://blogs.innodb.com/wp/2011/12/innodb-full-text-search-in-mysql-5-6-4/>
8. Introduction to mysql - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://www.php.net/manual/en/intro.mysql.php>
9. James W. Denton, A. Graham Peace, Selection and Use of MySQL in a Database Management Course, Journal of Information Systems Education, Vol. 14(4), 401-407p.
10. JCC’s SQL Standards Page - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://www.jcc.com/sql.htm#Status>
11. Oracle Database Documentation Library - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://www.oracle.com/pls/db112/homepage>
12. Using PHP OCI8 with 32-bit PHP on Windows 64-bit - [žiūrėta 2013-05-21]. Prieiga per internetą: <https://blogs.oracle.com/opal/entry/using_php_oci8_with_32-bit_php>
13. PostgreSQL 9.2 Documentation - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://www.postgresql.org/docs/9.2/interactive/index.html>
14. SQL Server 2008 R2 Documentation - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://technet.microsoft.com/en-us//library/ms130214(SQL.105).aspx>
15. Dell DVD Store Database Test Suite - [žiūrėta 2013-05-21]. Prieiga per internetą: <http://linux.dell.com/dvdstore/>
16. Natalia Juristo, Ana M. Moreno, Basics of software engineering experimentation, Universidad Politéncnica de Madrid, 2001 364p.
17. Stephanie Bell, A Beginner’s Guide to Uncertainty of Measurement. Issue 2, 2001. 33p. ISSN 1368-6550.

PRIEDAI

1. Priedas. Duomenų bazės lentelių atributai skirtingose DBVS

9 Lentelė. categories lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
category	tinyint(4)	NUMBER	serial	tinyint
categoryname	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)

10 Lentelė. cust_hist lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
customerid	int(11)	NUMBER	integer	int
orderid	int(11)	NUMBER	integer	int
prod_id	int(11)	NUMBER	integer	int

11 Lentelė. customers lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
customerid	int(11)	NUMBER	serial	int
firstname	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
lastname	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
address1	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
address2	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
city	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
state	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
zip	int(11)	NUMBER	integer	int
country	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
region	tinyint(4)	NUMBER	smallint	tinyint
email	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
phone	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
creditcardtype	int(11)	NUMBER	integer	tinyint
creditcard	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
creditcardexpiration	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
username	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
password	varchar(50)	VARCHAR2(50)	character varying(50)	varchar(50)
age	tinyint(4)	NUMBER	smallint	tinyint
income	int(11)	NUMBER	integer	int
gender	varchar(1)	VARCHAR2(1)	character varying(1)	varchar(1)

12 Lentelė. inventory lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
prod_id	int(11)	NUMBER	integer	int
quan_in_stock	int(11)	NUMBER	integer	int
sales	int(11)	NUMBER	integer	int

13 Lentelė. orderlines lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
orderlineid	smallint(6)	NUMBER	smallint	smallint
orderid	int(11)	NUMBER	integer	int
prod_id	int(11)	NUMBER	integer	int
quantity	smallint(6)	NUMBER	smallint	smallint
orderdate	date	DATE	date	datetime

14 Lentelė. orders lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
orderid	int(11)	NUMBER	serial	int
orderdate	date	DATE	date	datetime
customerid	int(11)	NUMBER	integer	int
netamount	decimal(12,2)	NUMBER(12,2)	numeric	money
tax	decimal(12,2)	NUMBER(12,2)	numeric	money
totalamount	decimal(12,2)	NUMBER(12,2)	numeric	money

15 Lentelė. products lentelės struktūra

Atributo vardas	MySQL	Oracle	PostgreSQL	MSSQL
prod_id	int(11)	NUMBER	serial	int
category	tinyint(4)	NUMBER	smallint	tinyint
title	varchar(50)	VARCHAR2(50)	text	varchar(50)
actor	varchar(50)	VARCHAR2(50)	text	varchar(50)
price	decimal(12,2)	NUMBER(12,2)	numeric	money
special	tinyint(4)	NUMBER	smallint	tinyint
common_prod_id	int(11)	NUMBER	integer	int

2. Priedas. Užklausos

```
SELECT
    PRODUCTS.TITLE,PRODUCTS.ACTOR,PRODUCTS.PRICE,PRODUCTS.SPECIAL,
    CATEGORIES.CATEGORYNAME,CUSTOMERS.FIRSTNAME,CUSTOMERS.LASTNAME,
    CUSTOMERS.ADDRESS1,CUSTOMERS.ADDRESS2,CUSTOMERS.CITY,
    CUSTOMERS.STATE,CUSTOMERS.COUNTRY,CUSTOMERS.REGION,
    CUSTOMERS.ZIP,CUSTOMERS.EMAIL,CUSTOMERS.PHONE,
    CUSTOMERS.CREDITCARDTYPE,CUSTOMERS.CREDITCARD,
    CUSTOMERS.CREDITCARDEXPIRATION,CUSTOMERS.USERNAME,
    CUSTOMERS.PASSWORD,CUSTOMERS.AGE,CUSTOMERS.INCOME,CUSTOMERS.GENDER
FROM
    CUST_HIST
    LEFT JOIN CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID
    LEFT JOIN PRODUCTS ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
    LEFT JOIN CATEGORIES ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
WHERE CUST_HIST.ORDERID > 50 AND CUST_HIST.ORDERID <= 100
LIMIT 50
```

38 pav. MySQL užklausa

```
SELECT TOP 50
PRODUCTS.TITLE, PRODUCTS.ACTOR, PRODUCTS.PRICE, PRODUCTS.SPECIAL,
CATEGORIES.CATEGORYNAME, CUSTOMERS.FIRSTNAME, CUSTOMERS.LASTNAME,
CUSTOMERS.ADDRESS1, CUSTOMERS.ADDRESS2, CUSTOMERS.CITY, CUSTOMERS.STATE,
CUSTOMERS.COUNTRY, CUSTOMERS.REGION, CUSTOMERS.ZIP,CUSTOMERS.EMAIL,
CUSTOMERS.PHONE, CUSTOMERS.CREDITCARDTYPE, CUSTOMERS.CREDITCARD,
CUSTOMERS.CREDITCARDEXPIRATION,CUSTOMERS.USERNAME, CUSTOMERS.PASSWORD,
CUSTOMERS.AGE, CUSTOMERS.INCOME, CUSTOMERS.GENDER
FROM
    CUST_HIST
    LEFT JOIN CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID
    LEFT JOIN PRODUCTS LEFT JOIN CATEGORIES
        ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
        ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
WHERE CUST_HIST.ORDERID > $a AND CUST_HIST.ORDERID <= ($a+50)
```

39 pav. MSSQL užklausa

```
SELECT
    PRODUCTS.TITLE, PRODUCTS.ACTOR, PRODUCTS.PRICE, PRODUCTS.SPECIAL,
    CATEGORIES.CATEGORYNAME,CUSTOMERS.FIRSTNAME, CUSTOMERS.LASTNAME,
    CUSTOMERS.ADDRESS1, CUSTOMERS.ADDRESS2, CUSTOMERS.CITY, CUSTOMERS.STATE,
    CUSTOMERS.COUNTRY, CUSTOMERS.REGION, CUSTOMERS.ZIP, CUSTOMERS.EMAIL,
    CUSTOMERS.PHONE, CUSTOMERS.CREDITCARDTYPE, CUSTOMERS.CREDITCARD,
    CUSTOMERS.CREDITCARDEXPIRATION, CUSTOMERS.USERNAME, CUSTOMERS.PASSWORD,
    CUSTOMERS.AGE, CUSTOMERS.INCOME, CUSTOMERS.GENDER
FROM
    CUST_HIST
    LEFT JOIN CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID
    LEFT JOIN PRODUCTS ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
    LEFT JOIN CATEGORIES ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
WHERE CUST_HIST.ORDERID > $a AND CUST_HIST.ORDERID < ($a+50)
LIMIT 50
```

40 pav. PostgreSQL užklausa

```
SELECT DS2.CUSTOMERS.CUSTOMERID, DS2.CUSTOMERS.FIRSTNAME, DS2.CUSTOMERS.LASTNAME,
    DS2.CUSTOMERS.ADDRESS1, DS2.CUSTOMERS.ADDRESS2,DS2.CUSTOMERS.CITY,
    DS2.CUSTOMERS.STATE,DS2.CUSTOMERS.COUNTRY,DS2.CUSTOMERS.REGION,
    DS2.CUSTOMERS.ZIP, DS2.CUSTOMERS.PHONE, DS2.CUSTOMERS.EMAIL,
    DS2.CUSTOMERS.CREDITCARD,DS2.CUSTOMERS.CREDITCARDTYPE,
    DS2.CUSTOMERS.CREDITCARDEXPIRATION, DS2.CUSTOMERS.AGE,DS2.CUSTOMERS.INCOME,
    DS2.CUSTOMERS.GENDER,DS2.CUSTOMERS.USERNAME,DS2.CATEGORIES.CATEGORYNAME,
    DS2.PRODUCTS.TITLE,DS2.PRODUCTS.ACTOR,DS2.PRODUCTS.PRICE,DS2.PRODUCTS.SPECIAL
FROM DS2.CUST_HIST
    LEFT JOIN DS2.CUSTOMERS ON DS2.CUSTOMERS.CUSTOMERID = DS2.CUST_HIST.CUSTOMERID
    LEFT JOIN DS2.PRODUCTS ON DS2.PRODUCTS.PROD_ID = DS2.CUST_HIST.PROD_ID
    LEFT JOIN DS2.CATEGORIES ON DS2.CATEGORIES.CATEGORY = DS2.PRODUCTS.CATEGORY
    WHERE DS2.CUST_HIST.ORDERID > $a AND DS2.CUST_HIST.ORDERID < ($a+50)
AND rownum < 50
```

41 pav. Oracle užklausa

4. Priedas. Vidinių procedūrų kodai

```
CREATE PROCEDURE `select_random`(IN `a` INT, IN `b` INT)
BEGIN
SELECT
    PRODUCTS.TITLE,PRODUCTS.ACTOR,PRODUCTS.PRICE,PRODUCTS.SPECIAL,
    CATEGORIES.CATEGORYNAME,CUSTOMERS.FIRSTNAME,CUSTOMERS.LASTNAME,
    CUSTOMERS.ADDRESS1,CUSTOMERS.ADDRESS2,CUSTOMERS.CITY,
    CUSTOMERS.STATE,CUSTOMERS.COUNTRY,CUSTOMERS.REGION,CUSTOMERS.ZIP,
    CUSTOMERS.EMAIL,CUSTOMERS.PHONE,CUSTOMERS.CREDITCARDTYPE,
    CUSTOMERS.CREDITCARD,CUSTOMERS.CREDITCARDEXPIRATION,CUSTOMERS.USERNAME,
    CUSTOMERS.PASSWORD,CUSTOMERS.AGE,CUSTOMERS.INCOME,
    CUSTOMERS.GENDER
FROM
    CUST_HIST
        LEFT JOIN CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID
        LEFT JOIN PRODUCTS ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
        LEFT JOIN CATEGORIES ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
WHERE CUST_HIST.ORDERID > a AND CUST_HIST.ORDERID <= b LIMIT 50;
END
```

42 pav. MySQL vidinė procedūra

```
USE [DS2]
GO
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
-- =============================================
-- Author:      Gediminas Kuliavas
-- Create date: 2013.04.17
-- Description: get 50 customers orders
-- =============================================
ALTER PROCEDURE [dbo].[getCustom]
    @a INT = 0
AS

SELECT TOP 50 PRODUCTS.TITLE, PRODUCTS.ACTOR, PRODUCTS.PRICE, PRODUCTS.SPECIAL, CATEGORIES.
CATEGORYNAME, CUSTOMERS.FIRSTNAME, CUSTOMERS.LASTNAME, CUSTOMERS.ADDRESS1, CUSTOMERS.ADDRESS2,
CUSTOMERS.CITY, CUSTOMERS.STATE, CUSTOMERS.COUNTRY, CUSTOMERS.REGION, CUSTOMERS.ZIP, CUSTOMERS.
EMAIL,CUSTOMERS.PHONE, CUSTOMERS.CREDITCARDTYPE, CUSTOMERS.CREDITCARD, CUSTOMERS.
CREDITCARDEXPIRATION,CUSTOMERS.USERNAME, CUSTOMERS.PASSWORD, CUSTOMERS.AGE, CUSTOMERS.INCOME,
CUSTOMERS.GENDER
FROM        CUST_HIST INNER JOIN
                CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID INNER JOIN
            PRODUCTS INNER JOIN
                CATEGORIES ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
                           ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
WHERE CUST_HIST.ORDERID > @a AND CUST_HIST.ORDERID <= @a+50
```

43 pav. MSSQL vidinė procedūra

```
CREATE OR REPLACE FUNCTION selectrandom(IN c integer)
  RETURNS TABLE(title text, actor text, price numeric, special smallint, categoryname character varying,
                firstname character varying, lastname character varying, address1 character varying,
                address2 character varying, city character varying, state character varying,
                country character varying, region smallint, zip character varying, email character varying,
                phone character varying, creditcardtype integer, creditcard character varying,
                creditcardexpiration character varying, username character varying, password character varying,
                age smallint, income integer, gender character varying) AS
$BODY$
 BEGIN
    RETURN QUERY
     SELECT PRODUCTS.TITLE, PRODUCTS.ACTOR, PRODUCTS.PRICE, PRODUCTS.SPECIAL, CATEGORIES.CATEGORYNAME,
     CUSTOMERS.FIRSTNAME, CUSTOMERS.LASTNAME, CUSTOMERS.ADDRESS1, CUSTOMERS.ADDRESS2, CUSTOMERS.CITY,
     CUSTOMERS.STATE, CUSTOMERS.COUNTRY, CUSTOMERS.REGION, CUSTOMERS.ZIP, CUSTOMERS.EMAIL, CUSTOMERS.PHONE,
     CUSTOMERS.CREDITCARDTYPE, CUSTOMERS.CREDITCARD, CUSTOMERS.CREDITCARDEXPIRATION, CUSTOMERS.USERNAME,
     CUSTOMERS.PASSWORD, CUSTOMERS.AGE, CUSTOMERS.INCOME, CUSTOMERS.GENDER
    FROM CUST_HIST
    INNER JOIN CUSTOMERS ON CUST_HIST.CUSTOMERID = CUSTOMERS.CUSTOMERID
    INNER JOIN PRODUCTS ON CUST_HIST.PROD_ID = PRODUCTS.PROD_ID
    INNER JOIN CATEGORIES ON PRODUCTS.CATEGORY = CATEGORIES.CATEGORY
    WHERE CUST_HIST.ORDERID > c AND CUST_HIST.ORDERID < c+50
    LIMIT 50;
END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100
  ROWS 50;
```

44 pav. PostgreSQL vidinė procedūra

```
CREATE OR REPLACE
PROCEDURE selectRandom
(  C IN NUMBER, p_rc OUT sys_refcursor)
AS
BEGIN
 OPEN p_rc
 for
  SELECT DS2.CUSTOMERS.CUSTOMERID, DS2.CUSTOMERS.FIRSTNAME, DS2.CUSTOMERS.LASTNAME,
  DS2.CUSTOMERS.ADDRESS1,DS2.CUSTOMERS.ADDRESS2,DS2.CUSTOMERS.CITY,DS2.CUSTOMERS.STATE,
  DS2.CUSTOMERS.COUNTRY,DS2.CUSTOMERS.REGION,DS2.CUSTOMERS.ZIP,DS2.CUSTOMERS.PHONE,
  DS2.CUSTOMERS.EMAIL,DS2.CUSTOMERS.CREDITCARD,DS2.CUSTOMERS.CREDITCARDTYPE,
  DS2.CUSTOMERS.CREDITCARDEXPIRATION,DS2.CUSTOMERS.AGE,DS2.CUSTOMERS.INCOME,
  DS2.CUSTOMERS.GENDER,DS2.CUSTOMERS.USERNAME,DS2.CATEGORIES.CATEGORYNAME,
  DS2.PRODUCTS.TITLE,DS2.PRODUCTS.ACTOR,DS2.PRODUCTS.PRICE,DS2.PRODUCTS.SPECIAL
FROM DS2.CUST_HIST
  JOIN DS2.CUSTOMERS ON DS2.CUSTOMERS.CUSTOMERID = DS2.CUST_HIST.CUSTOMERID
  JOIN DS2.PRODUCTS ON DS2.PRODUCTS.PROD_ID = DS2.CUST_HIST.PROD_ID
  JOIN DS2.CATEGORIES ON DS2.CATEGORIES.CATEGORY = DS2.PRODUCTS.CATEGORY
WHERE DS2.CUST_HIST.ORDERID > c AND DS2.CUST_HIST.ORDERID < c+50 AND rownum < 50;
END;
```

45 pav. Oracle vidinė procedūra