PANEVĖŽIO KOLEGIJA ISSN 2029-1280. Taikomieji tyrimai studijose ir praktikoje – Applied Research in Studies and Practice, 2017, 13.

SQL UŽKLAUSŲ OPTIMIZAVIMO TYRIMAS

Jurgita Lieponienė

Panevėžio kolegija, Lietuva

Anotacija. Informacinėms sistemoms sparčiai tobulėjant, didėja ir jų apdorojamų duomenų kiekiai, o kartu mažėja vienas iš svarbiausių informacinių sistemų rodiklių – greitis. Kadangi informacinių sistemų naudotojui svarbu kuo greičiau gauti jų pateiktų užklausų rezultatus, todėl informacinių sistemų, duomenų bazių programuotojai, kurdami ir modernizuodami informacines sistemas turi įvertinti visus parametrus įtakojančius informacinių sistemų greitį, tarp kurių ir naudojamų SQL užklausų efektyvumas. Vykdomo tyrimo tikslas: atlikti SQL užklausų optimizavimo, optimizuojant SQL kodą, naudojant duomenų bazės indeksus ir denormalizuojant duomenų bazės struktūrą, tyrimą. Straipsnyje pristatomi šio tyrimo rezultatai ir formuluojamos apibendrinančios išvados.

Raktiniai žodžiai: SQL užklausa; duomenų bazės indeksas; duomenų bazės normalizavimas; duomenų bazės denormalizavimas.

ĮVADAS

Informacinių sistemų sparta yra toks pats svarbus informacinių sistemų atrankos parametras kaip ir funkcionalumas, kaina, patikimumas, gamintojo reputacija, integracijos galimybės, technologijų naujumas, plėtros galimybės, versijų atnaujinimas, pritaikomumas, naudojimo paprastumas, lankstumas, moduliškumas (Ratkevičius, 2011). Vienas iš parametrų, įtakojančių informacinių sistemų atsako laiką, yra informacinėje sistemoje naudojamų SQL užklausų efektyvumas (Habimana, 2015). SQL užklausų efektyvumas didinamas jas optimizuojant.

Mokslinėje literatūroje nagrinėjami įvairūs SQL užklausų optimizavimo būdai. Habimana (2015) analizavo SQL užklausų optimizavimą, optimizuojant SQL kodą. Costel, Luca ir Teodor (2014) nagrinėjo Microsoft SQL Server siūlomas SQL optimizavimo technikas. Mithani, Machchhar ir Jasdanwala (2016) apibendrino SQL kodo optimizavimo taisykles ir pasiūlė SQL užklausų konvertavimo į optimizuotas užklausas modelį, užtikrinantį trumpesnį SQL užklausų vykdymo laiką. Bhajipaleir kt. (2016) analizavo SQL užklausų optimizavimą, projektuojant efektyvesnę duomenų bazės struktūrą, optimizuojant duomenų bazės indeksus bei SQL kodą, analizuojant užklausų vykdymo planus. Oktavia ir Sujarwo (2014) tyrė vidinių SQL užklausų naudojimo būdus ir duomenų bazės indeksavimo strategiją, kuri teigiamai įtakoja vidinių SQL užklausų vykdymo trukmę. Mokslinėje literatūroje taip pat pateikiama visa eilė tyrimų, kuriuose nagrinėjamas NoSQL duomenų bazių užklausų efektyvumas.

Mokslininkai, analizuojantys SQL užklausų efektyvumą, akcentuoja, kad nors tuos pačius rezultatus galima gauti sudarant kelias skirtingas SQL užklausas, tačiau šiandieniniam naudotojui aktualios tik tos užklausos, kurios leidžia gauti rezultatus per kuo trumpesnį laiką. Neoptimizuotų užklausų vykdymo kaina yra labai didelė (Gupta ir Chandra, 2011). Neefektyvios SQL užklausos neigiamai įtakoja verslo sistemų greitį ir tuo pačių mažina verslo efektyvumą (Oktavia ir Sujarwo, 2014).

Tyrimo aktualumas ir naujumas. Nors mokslinėje literatūroje SQL užklausų optimizavimas tiriamas įvairiais aspektais, tačiau didžioji dalis tyrimų apsiriboja teorine problemų analize. Trūksta tyrimų, eksperimentinio testavimo rezultatais pagrindžiančių SQL užklausų optimizavimo metodų efektyvumą. Todėl vykdomu tyrimu eksperimentinio testavimo būdu buvo vertinama SQL kodo optimizavimo, duomenų bazės indeksų, duomenų bazės struktūros svarba, optimizuotų SQL užklausų kūrime. Šio tyrimo rezultatai svarbūs informacinių sistemų, duomenų bazių programuotojams ir kitiems IT specialistams, kuriantiems informacines sistemas bei analizuojantiems duomenų bazių duomenis. Eksperimentinio tyrimo rezultatai taip pat gali būti naudojami aukštosiose mokyklose, dėstant duomenų bazes.

Tyrimo objektas: SQL užklausų optimizavimas.

Tyrimo tikslas – atlikti SQL užklausų optimizavimo, optimizuojant SQL kodą, naudojant duomenų bazės indeksus ir denormalizuojant duomenų bazės struktūrą, tyrimą.

Tyrimo uždaviniai:

1. Įvertinti SQL kodo įtaką, SQL užklausos efektyvumui.
2. Nustatyti, kaip duomenų bazės indeksai įtakoja SQL užklausų vykdymo trukmę.

3. Įvertinti duomenų bazės struktūros denormalizavimo įtaką SQL užklausų efektyvumui.

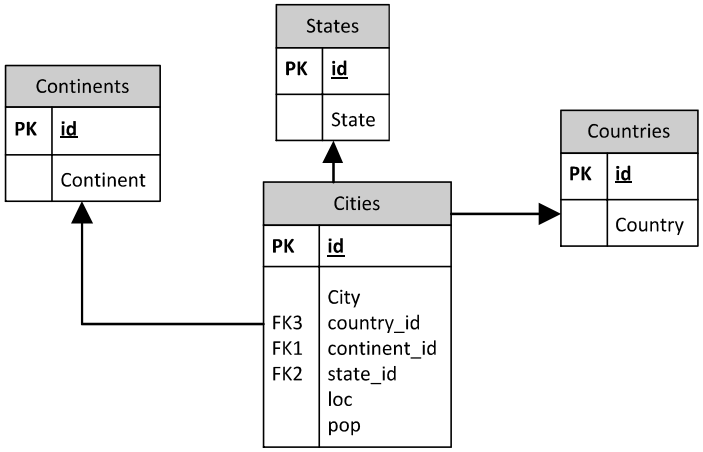
Tyrimo metodai: SQL kodo, duomenų bazės indeksų ir duomenų bazės denormalizavimo įtaka SQL užklausų efektyvumui vertinta eksperimentinio testavimo būdu. Eksperimentinis testavimas atliktas Oracle XE duomenų bazių valdymo sistemoje, vykdant testus skirtingo dydžio duomenų bazėse, jas indeksuojant ir denormalizuojant.

Atliktas tyrimas svarbus tiek teoriniu, tiek praktiniu aspektu. Tyrimo rezultatai svarbūs duomenų bazių programuotojams ir kitiems IT specialistams, vykdantiems SQL užklausas. Atlikto tyrimo rezultatai gali būti integruojami į aukštosiose mokyklose dėstomą duomenų bazių dalyką, pagrindžiant SQL užklausų optimizavimo svarbą ir apibrėžiant gerąsias SQL kodo rašymo praktikas. Pateikta tyrimo metodika gali būti taikoma vertinant užklausų efektyvumą ir NoSQL duomenų bazėse.

EKSPERIMENTINIO TYRIMO METODOLOGIJA

Tyrimui taikomas eksperimentinio testavimo metodas. Eksperimentinis testavimas vykdomas Oracle XE duomenų bazių valdymo sistemoje. Eksperimentui atlikti sugeneruoti 4 testinių duomenų rinkiniai, naudojant *XML Data Generator* įrankį. Pirmąjį testinių duomenų rinkinį sudaro 10000 įrašų, antrąjį – 50000 įrašų, trečiąjį – 250000 įrašų, o ketvirtąjį – 1250000 įrašų, pateikiamų XML formatu. Automatiškai sugeneruoti duomenys aprašo miestų informaciją: pateikia miesto pavadinimą, miesto geografines koordinates, gyventojų skaičių, valstiją, šalį, žemyną kuriam priklauso miestas.

Eksperimentui atlikti Oracle XE duomenų bazių valdymo sistemoje sukurtos keturios testinės schemos į kurias importuoti automatiškai sugeneruoti testiniai duomenys. Testinių schemų duomenų struktūra pateikta 1 paveiksle.



1 pav. **Testinės schemos duomenų struktūra**

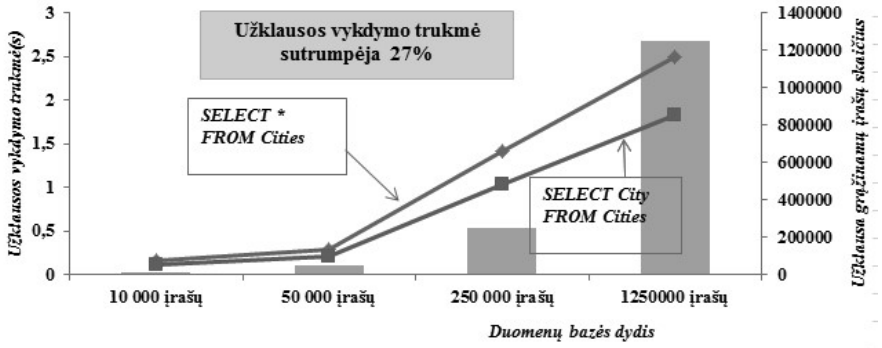
Eksperimentinis tyrimas sudarytas iš trijų dalių. Pirmąja ekperimentinio tyrimo dalimi, naudojant keturis testus, vertinamos SQL užklausų optimizavimo galimybės, optimizuojant užklausos kodą. Kiekvieną testą sudaro originali ir pakeista (optimizuoto kodo) SQL užklausa. Kiekvieno testo užklausos vykdomos po 10 kartų Oracle XE testinėse duomenų bazėse. Užklausų vykdymo trukmė fiksuojama užklausų vykdymo *log* faile. Apibendrinant eksperimentinio tyrimo rezultatus, skaičiuojami užklausų vykdymo trukmės vidurkiai.

Antrąja ekperimentinio tyrimo dalimi, naudojant du testus, vertinama duomenų bazės indeksų įtaka SQL užklausų efektyvumui. Kiekvienam testui sukurta užklausa vykdoma tiek indeksuotose, tiek neindeksuotose duomenų bazėse po 10 kartų. Eksperimento tyrimo rezultatai apibendrinami skaičiuojant užklausų vykdymo trukmės vidurkį.

Paskutiniąja eksperimentinio tyrimo dalimi vertinama duomenų bazės schemos denormalizavimo įtaka vykdomų SQL užklausų efektyvumui. Vertinant duomenų bazės denormalizavimo įtaką užklausos vykdymo greičiui sukurti trys testai. Kiekvienam testui sukurta originali ir optimizuota SQL užklausa, pritaikyta denormalizuotai duomenų bazės schemai. Testavimo metu originali užklausa vykdoma normalizuotose, o optimizuoto kodo užklausa – denormalizuotose duomenų bazėse. Eksperimentinio tyrimo statistinė duomenų analizė atlikta naudojant Ms Excel 2016 programą. Tyrimo duomenys apibendrinti skaičiuojant užklausų vykdymo trukmės aritmetinį vidurkį.

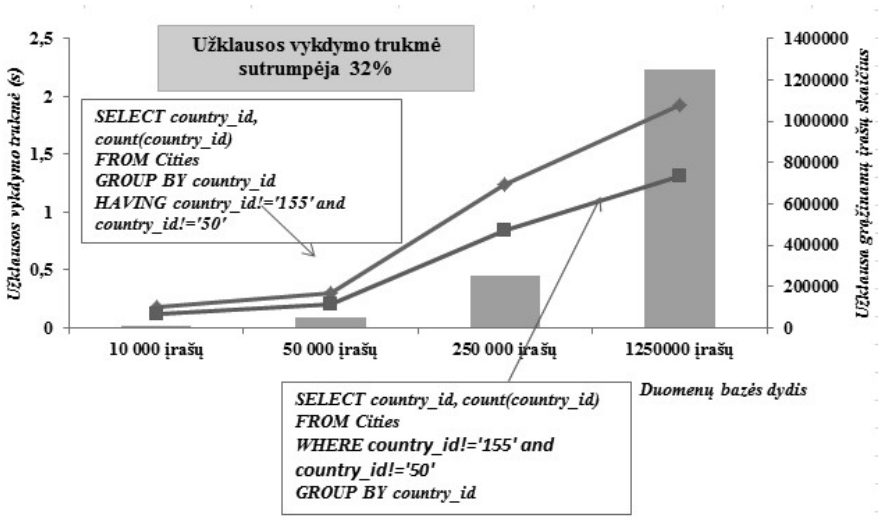
EKSPERIMENTINIO TYRIMO REZULTATAI IR JŲ ANALIZĖ

Pirmąja ekperimentinio tyrimo dalimi vertinamos SQL užklausų optimizavimo galimybės, optimizuojant užklausos kodą. Eksperimentas pradedamas užklausos, kuri atrenka duomenis iš lentelės *Cities*, vykdymu. Originalios užklausos SELECT frazėje nėra įvardijami tikslūs duomenų bazės laukai, bet naudojamas „*“ simbolis. Optimizuotojo užklausoje konkretizuojamas atrenkamas duomenų bazės lentelės laukas. Testavimo rezultatai pateikti 2 paveiksle. Nors testinės užklausos grąžina vienodą įrašų skaičių, tačiau jų vykdymo trukmės skiriasi. Originalios ir optimizuoto kodo SQL užklauso vykdymo trukmės skirtumas auga, didėjant testuojamų duomenų bazių įrašų skaičiui. Testavimo rezultatai rodo, kad SQL užklausos optimizavimas, nurodant tikslius duomenų bazės laukus, gali sutrumpinti užklausos vykdymo trukmę iki 27,0 proc..



2 pav. **„*“ simbolio naudojimo SELECT frazėje įtaka SQL užklausos efektyvumui**

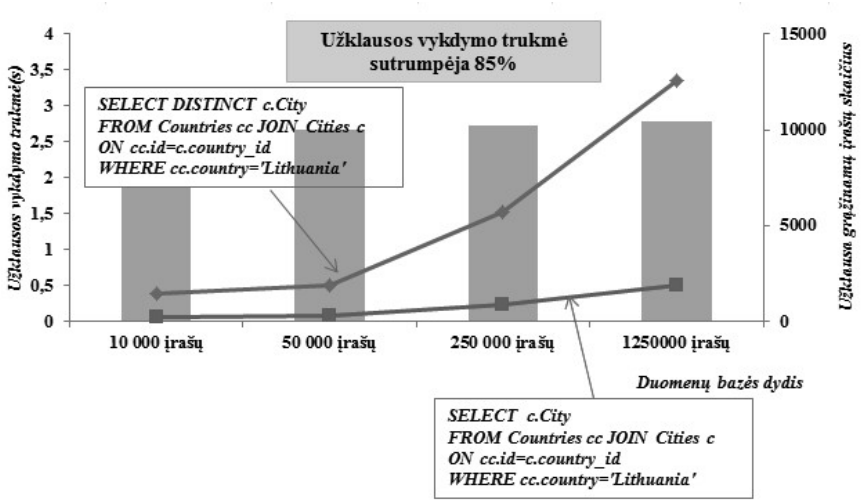
Antruoju testu tikrinama HAVING frazės naudojimo SQL sakinyje įtaka užklausos efektyvumui. HAVING frazė SQL sakiniuose naudojama, apibrėžiant grupių atrankos kriterijams. Šią frazę dažnai galima pakeisti WHERE fraze, kuri vykdoma prieš atliekant grupavimą ir taip atrenkamos tik tam tikros įrašų grupės. Tiek originali, tiek optimizuoto kodo SQL užklausa atrenka tuos pačius lentelės Cities įrašus, tačiau optimizuoto SQL kodo užklausoje HAVING frazė keičiama WHERE fraze. Testavimo rezultatai pateikti 3 paveiksle rodo, kad HAVING frazės naudojimas SQL užklausoje prailgina užklausos vykdymo trukmę 32,0 proc..



3 pav. **HAVING frazės įtaka SQL užklausos efektyvumui**

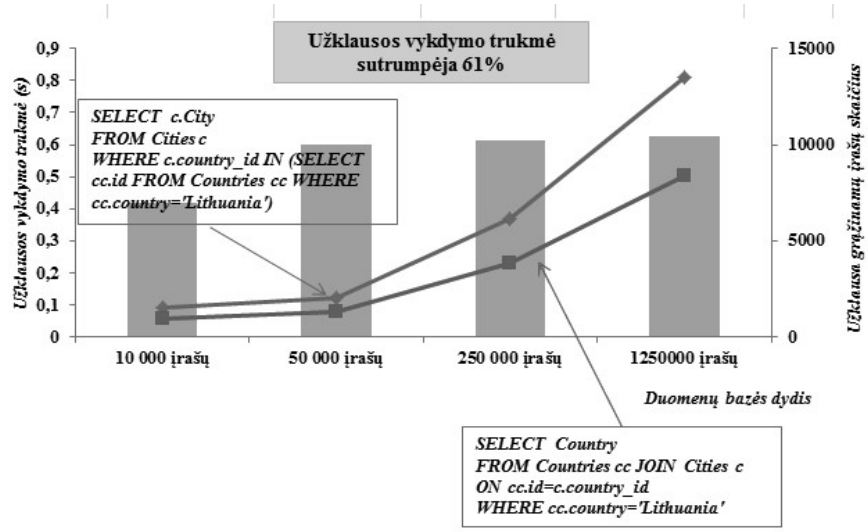
Tęsiant eksperimentą tikrama, kaip SQL užklausos vykdymo laiką įtakoja operatoriaus *Distinct* naudojimas. Naudojant šį operatorių, SQL užklausose užtikrinama nepasikartojančių įrašų atranka. Šis operatorius ypač dažnai naudojamas programuotojų net neįvertinus jų naudojimo būtinumo (Ordonez, 2010). Trečiojo testo originalioje užklausoje pertekliškai naudojamas operatorius *Distinct*, o optimizuoto kodo SQL sakinyje šio operatoriaus atsisakoma. Eksperimentinio tyrimo rezultatai rodo (žr. 4 pav.), kad perteklinis operatoriaus *Distinct* naudojimas užklausose gali prailginti jų vykdymo trukmę net iki 85,0 proc.. Originalios

ir optimizuoto kodo SQL užklauso vykdymo trukmės skirtumas ypač išryškėja didėjant užklausa grąžinamų įrašų skaičiui.



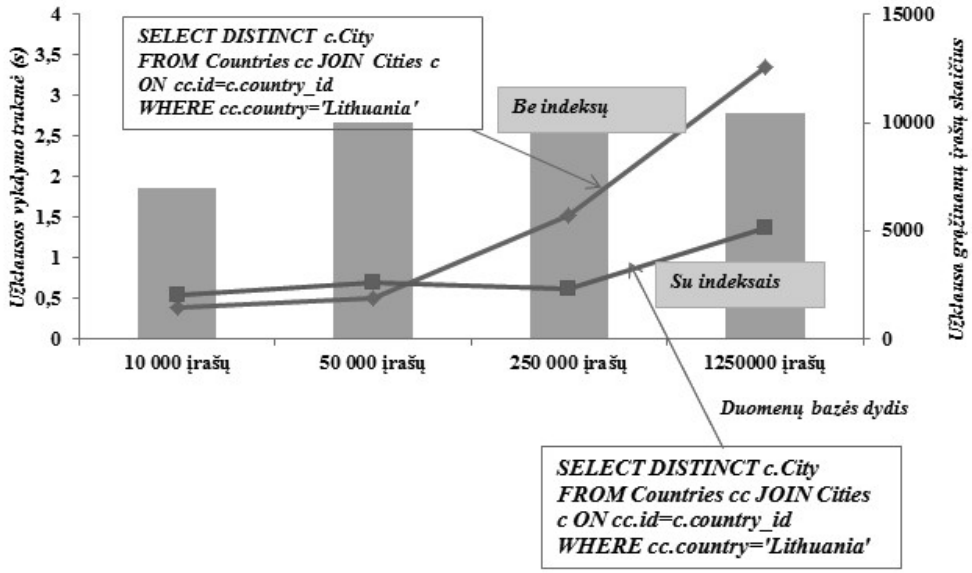
4 pav. **Operatoriaus Distinct įtaka SQL užklausos efektyvumui**

SQL užklausų įrašų atrankos kriterijų apibrėžimui dažnai naudojamos vidinės užklausos (Oktavia, Sujarwo, 2014). Ketvirtuoju testu vertinama vidinių užklausų įtaka SQL užklausų vykdymo trukmei. Originalioje užklausoje, atrenkant Lietuvos miestus, atrankos kriterijai formuojami, naudojant vidinę užklausą. Optimizuoto kodo užklausoje tie patys rezultatai gaunami, atrenkant įrašus iš dviejų lentelių ir WHERE frazėje nurodant elementarius apribojimus. Eksperimentinio tyrimo rezultatai (žr. 5 pav.) rodo, kad vidinių užklausų naudojimas SQL sakiniuose gali prailginti užklausos vykdymo laiką iki 61,0 proc. didėjant užklausa grąžinamų įrašų skaičiui, ryškėja originalios ir optimizuotos užklausos vykdymo laiko skirtumas.



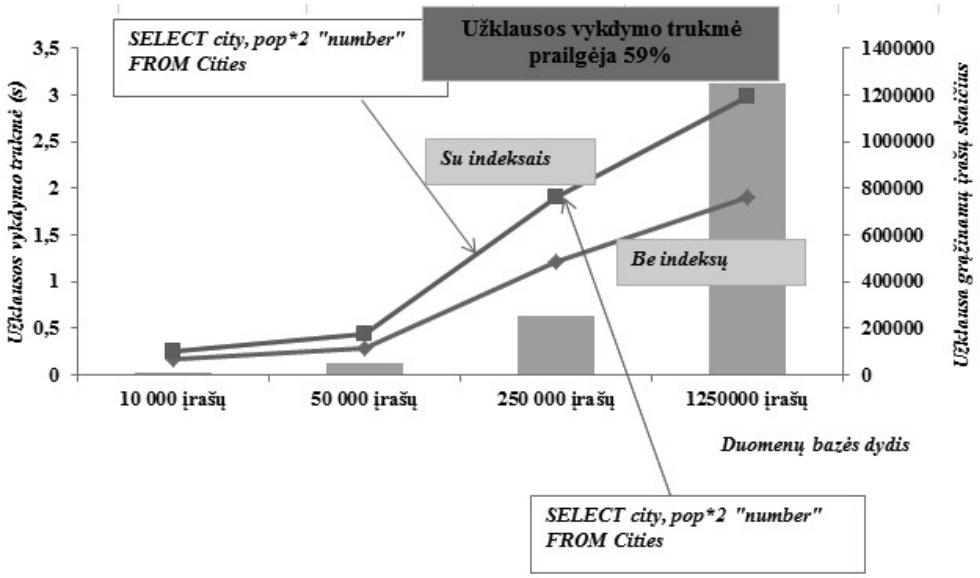
5 pav. **Vidinės užklausos įtaka SQL užklausos efektyvumui**

Pasak Chunxia (2016) duomenų bazių indeksai yra svarbūs, vykdant paiešką didelėse duomenų bazėse. Antrąja ekperimentinio tyrimo dalimi vertinama, indeksų nauda užklausų efektyvumo didinimui. Šiai eksperimentinio tyrimo daliai buvo sukurti du testai, kiekvienam testui parengiant po vieną užklausą. Skirtingai nuo pirmosios eksperimentinio tyrimo dalies, testo užklausa vykdoma tiek indeksuotose, tiek neindeksuotose duomenų bazėse. Pirmuoju testu vykdomos Lietuvos miestų atrankos užklausos apribojimai apibrėžiami WHERE frazėje. Testinė užklausa pirmiausiai vykdoma neindeksuotose duomenų bazėse, o po to kartojama duomenų bazėse indeksuotose pagal atrankos kriterijų formuojantį lauką *Country*. Eksperimentinio testavimo rezultatai pateikti 6 paveiksle. Atliktu eksperimentu nustatyta, kad duomenų bazės pagal atrankos lauką indeksavimas yra efektyvus tik tuomet, kai vykdoma užklausa grąžina mažiau nei 10,0 proc. visų duomenų bazės įrašų.



6 pav. **Indeksų įtaka SQL užklausos efektyvumui**

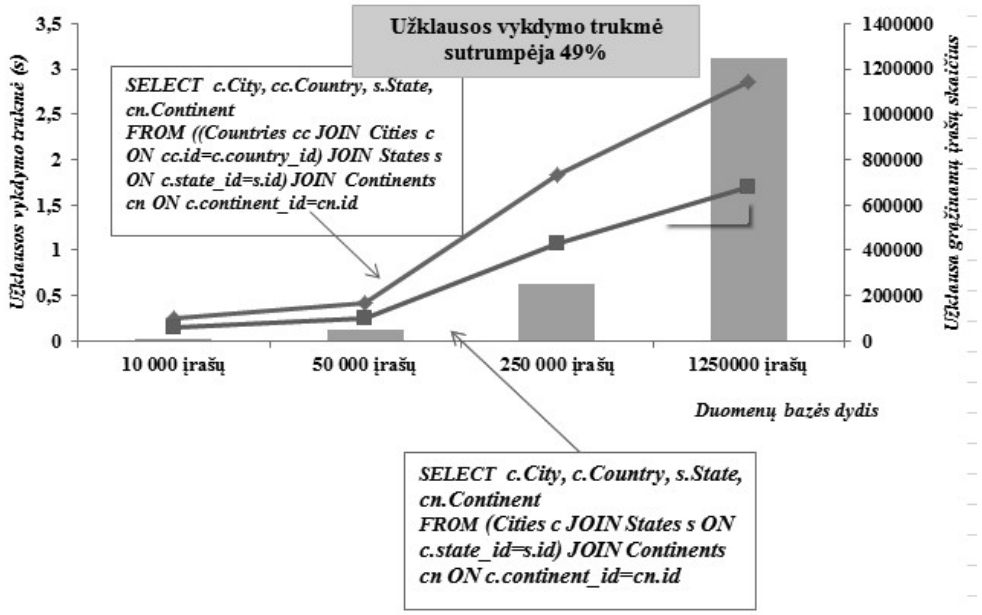
Antruoju testu vertinamas duomenų bazės indeksų efektyvumas SQL užklausų optimizavimui, indeksuojant skaičiuojamus laukus. Sukurtos testinės užklausos SELECT frazėje apskaičiuojamas padvigubintas miestų gyventojų skaičius. Testo užklausa vykdoma tiek indeksuotose, tiek neindeksuotose duomenų bazėse. Testinių duomenų bazių indeksavimas vykdomas pagal skaičiavimuose naudojamą lauką *Pop*. Atlikto eksperimentinio tyrimo rezultatai (žr. 7 pav.) rodo, kad duomenų bazės indeksavimas pagal skaičiuojamą lauką neigiamai įtakoja vykdomos SQL užklausos efektyvumą. Netinkamas duomenų bazės indeksavimas prailgina SQL užklausos vykdymo laiką iki 59,0 proc..



7 pav. **Skaičiuojamų laukų indeksų įtaka SQL užklausos efektyvumui**

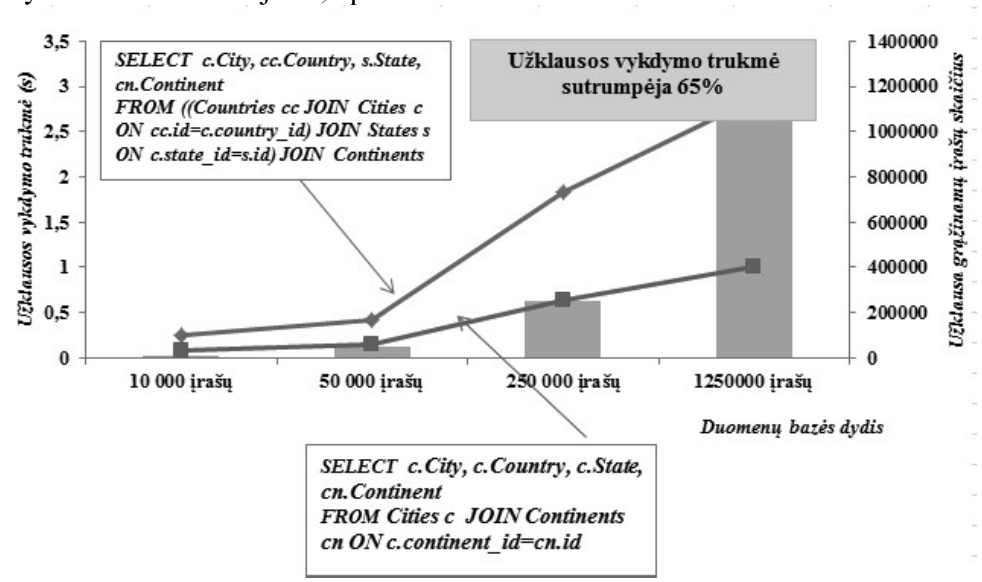
Paskutiniąja eksperimentinio tyrimo dalimi vertinama duomenų bazės schemos denormalizavimo įtaka vykdomų SQL užklausų efektyvumui. Projektuojant reliacines duomenų bazes, vykdomas duomenų bazės schemos normalizavimas, siekiant užtikrinti saugomų duomenų neperteklišumą, integralumą, vientisumą. Tačiau ar duomenų bazės schemos normalizavimas yra efektyvus vykdomoms SQL užklausoms? Vertinant duomenų bazės schemos denormalizavimo įtaką užklausos vykdymo greičiui, sukurti trys testai. Kiekvienam testui sukurta originali ir optimizuota SQL užklausa, pritaikyta denormalizuotai duomenų bazės schemai. Testavimo metu originali užklausa vykdoma normalizuotose duomenų bazėse, o optimizuotos – denormalizuotose duomenų bazėse. Pirmojo testo originali užklausa atrenka duomenis iš keturių lentelių, o optimizuota užklausa tuos pačius įrašus atrenka iš denormalizuotų duomenų bazių, apjungdama tris lenteles. Atlikus eksperimentą nustatyta (žr. 8 pav.), kad denormalizavus duomenų bazės

schemą ir SQL užklausą optimizavus, sumažinant sąryšių skaičių vienetu, užklausos vykdymo trukmė sutrumpėja 49,0 proc..



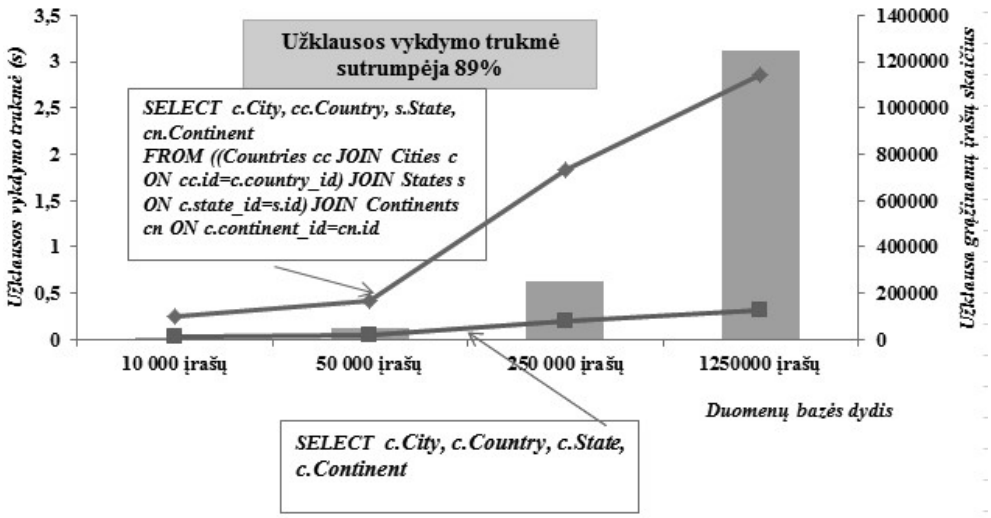
8 pav. **Duomenų bazės schemos denormalizavimo įtaka SQL užklausos efektyvumui** ***(sąryšių skaičius sumažintas iki dviejų sąryšių)***

Tęsiant eksperimentą eksperimentinės duomenų bazės denormalizuojamos, siekiant dar labiau sumažinti optimizuoto kodo užklausoje naudojamų sąryšių skaičių. Antrojo testo originali užklausa atrenka duomenis iš keturių lentelių, o optimizuoto kodo užklausa tuos pačius įrašus atrenka iš denormalizuotų duomenų bazių, apjungdama tik dvi lenteles. Apibendrinti eksperimentinio testavimo rezultatai pateikti 9 paveiksle. Optimizavus SQL užklausą, sumažinant užklausoje naudojamų sąryšių skaičių iki vieno sąryšio, užklausos vykdymo trukmė sumažėja 65,0 proc..



9 pav. **Duomenų bazės denormalizavimo įtaka SQL užklausos efektyvumui** ***(sąryšių skaičius sumažintas iki vieno sąryšio)***

Paskutiniuoju testavimo atveju optimizuoto kodo SQL užklausoje nėra naudojami sąryšiai. Testinės duomenų bazės maksimaliai denormalizuojamos, siekiant išvengti sąryšių optimizuotoje SQL užklausoje naudojimo. Atlikus eksperimentą nustatyta, kad SQL užklausoje nenaudojant sąryšių galima sutrumpinti užklausos vykdymo trukmę iki 89,0 proc. (žr. 10 pav.).



10 pav. **Duomenų bazės denormalizavimo įtaka SQL užklausos efektyvumui *(optimizuoto kodo užklausoje nenaudojami sąryšiai)***

Apibendrinant eksperimentinio tyrimo rezultatus galima teigti, kad SQL užklausos efektyvumą įtakoja tinkamai parašytas SQL kodas, teisingai naudojami duomenų bazės indeksai ir duomenų bazės schemos denormalizavimas, siekiant eliminuoti sąryšių naudojimą SQL užklausose.

IŠVADOS

1. Vertinant SQL kodo įtaką užklausos efektyvumui, nustatyta, kad tinkamai parašytas SQL kodas sumažina SQL užklausos vykdymo trukmę iki 85,0 proc. Užklausos vykdymo trukmę neigiamai įtakoja vidinių užklausų, HAVING frazės, operatoriaus Distinct naudojimas. Atrenkamų užklausos laukų įvardijimas SELECT frazėje gali sumažinti užklausos vykdymo trukmę iki 27,0 proc..
2. Atliktu eksperimentiniu tyrimu nustatyta, kad skaičiavimuose naudojamų duomenų bazės laukų indeksavimas neigiamai įtakoja užklausos vykdymo trukmę. Atrankos kriterijus formuojančių užklausos laukų indeksavimas efektyvus tik tuomet, kai vykdoma atrankos užklausa grąžina mažiau nei 10,0 proc. duomenų bazės įrašų.
3. Atliktas eksperimentinis tyrimas parodė, kad duomenų bazės denormalizavimas turi teigiamą įtaką SQL užklausų efektyvumui. Denormalizuojant duomenų bazę, siekiant išvengti sąryšių SQL užklausose naudojimo, galima sumažinti užklausų vykdymo trukmę iki 89,0 proc..

LITERATŪROS SĄRAŠAS

Bhajipale, R., Bisen, P., Meshram, A., & Thakur, S. (2016). SQL Tuner // *International Journal of Computer Trends and Technology (IJCTT)*, 33(1), 29-32.

Chunxia, Qi. (2016). On index-based query in SQL Server database // *Control Conference (CCC)*, 56-65.

Costel, G., Luca, V., & Teodor, O. (2014). Query Optimization Techniques in Microsoft SQL Server // *Database Systems Journal*, 2, 33-48.

Gupta, M., & Chandra, P. (2011). An Empirical Evaluation of LIKE Operator in Oracle // *International Journal of Information Technology*, 55-65.

Habimana, J. (2015). Query Optimization Techniques – Tips For Writing Efficient And Faster SQL Queries // *International Journal of Scientific & Technology Research*, 4(10), 22-26.

Mithani, F., Machchhar, S., & Jasdanwala, F. (2016). A novel approach for SQL query optimization // *Computational Intelligence and Computing Research (ICCIC)*, 15-25.

Oktavia, T., & Sujarwo, S. (2014). Evaluation of Sub Query Performance in SQL Server // *EPJ Web of Conferences*, 16-22.

Ordonez, C. (2010). Optimization of Linear Recursive Queries in SQL // *IEEE Transactions on Knowledge & Data Engineering*, 22(2), 264-277.

Ratkevičius, D. (2011). Neprograminiai verslo valdymo sistemų atrankos kriterijai // *Socialinių mokslų studijos*, 13(1), 1359–1374.

Summary
SQL query optimization research

Fast improvement of information systems causes the increase of data quantity processed by them. Databases are an integral part of information systems that store billions of information system records. Given such quantity of data and taking into account its future growth it can be said that the time of information system operations increases not linearly but exponentially, whereas one of the most important information system indexes – speed – decreases. As users of information systems are eager to get the results of their queries as quickly as possible the programmers who create and upgrade information systems should assess all the parameters that have influence on information system speed, including the effectiveness of used SQL queries. Scientific literature studies different ways of SQL query optimization. Habimana (2015) analyzed SQL query optimisation optimising SQL codeCostel, Luca, and Teodor (2014) studied SQL optimization techniques offered by Microsoft SQL Server. Mithani, Machchhar, and Jasdanwala (2016) summarised SQL code optimization rules and suggested the model of SQL query converting into optimized query that ensures shorter time of SQL query execution. Bhajipale, et al. (2016) analyzed SQL query optimization designing more effective structure of data base, optimizing data base indexes and SQL code, analyzing query execution plans. Oktavia and Sujarwo (2014) explored the methods of internal SQL queries se and data base indexing strategy which makes positive impact on the duration of internal SQL queries. Besides, scientific literature provides a wide range of researches that study the effectiveness of NoSQL data base queries. Scientists who analyze the effectiveness of SQL queries emphasise that although the same results can be received making several different queries, a modern user needs only those queries that allow obtaining results in record-breaking time. The price of unoptimized query execution is very high (Gupta & Chandra, 2011). Ineffective SQL queries have a negative effect on the speed of business systems and reduce business effectiveness (Oktavia & Sujarwo, 2014).

Research topicality and novelty. Although scientific literature studies different aspects of SQL query optimization the major part of researches is limited to the theoretical analysis of problems. There is a lack of researches that ground the effectiveness of SQL query optimization methods on the results of experimental testing. In light of this the aim of the current research is to assess the importance of SQL code optimization, data base indexes and structure in the creation of optimized SQL queries by means of experimental testing. The results of this research are important for the programmers of information systems, databases and other IT specialist who create information systems and analyze the data of databases. The results of experimental research can be used in high schools teaching databases.

Research object is SQL query optimization.

Research objective is to perform the research of SQL query optimization optimizing SQL code, using data base indexes and denormalizing data base structure.

Research tasks:

1. To assess the impact of SQL code on the effectiveness of SQL query.
2. To ascertain how data base indexes influence the duration of SQL query execution.
3. To assess the impact of data base denormalization on the effectiveness of SQL queries.

Research methods: the impact of SQL code, data base indexes and data base denormalization on the effectiveness of SQL queries was assessed my means of experimental testing. The experimental testing was implemented in Oracle XE data base management system conducting tests in various in size databases, indexing them and denormalizing.

The conducted research is important both in theoretical and practical aspects. Research results are important for data base programmers and other IT specialists who execute SQL queries. The results of the conducted research can be integrated into the data base subject teaching in higher schools grounding the importance of SQL query optimization and defining good practice of SQL code writing. The presented research method can be also used assessing the effectiveness of queries in NoSQL databases.

Conclusions:

1. During the assessment of the impact of SQL code on the effectiveness of query it was established that properly written SQL code reduces the duration SQL query execution by 85,0 percent. The duration of query execution is negatively influenced by internal queries, phrase HAVING and use of operator Distinct. The designation of query field in SELECT phrase can reduce the duration of query execution by 27,0 percent.
2. The conducted experimental research showed that the indexing of query fields used in calculations has negative impact on the duration of query execution. The indexing of query fields that form selection criteria is effective only when selection query returns less than 10,0 percent of data base records.
3. The conducted experimental research showed that the denormalization of data base has positive impact on the effectiveness of SQL queries. Denormalizing data base on purpose to avoid connections in SQL query use it is possible to reduce the time of query execution by 89,0 percent.

