Skaidymo metodų vertinimas skirtingiems duomenų bazių tipams

Anton Zagzin, Pijus Zlatkus, Vasilij Savin

Vilniaus Universitetas, Matematikos ir informatikos fakultetas,
Informatikos institutas,
Didlaukio g. 47, Vilnius
anton.zagzin@mif.stud.vu.lt, pijus.zlatkus@mif.stud.vu.lt,
vasilij.savin@mif.vu.lt

Santrauka. Straipsnyje vertinami maišos (angl. *hash-based*) ir diapazoninio (angl. *range-based*) duomenų skaidymo metodai PostgreSQL (su Citus plėtiniu) ir MongoDB duomenų bazėse. Eksperimentai atlikti automatizuojant duomenų įkėlimą bei vykdant įvairaus sudėtingumo užklausas naudojant JMeter įrankį. Rezultatai parodė, jog maišos metodas yra efektyvesnis izoliuotoms užklausoms, tuo tarpu diapazoninis skaidymas geriau tinka intervalinėms užklausoms. Pastebėtas ryškus PostgreSQL su Citus pranašumas automatizuojant klasterio operacijas, lyginant su MongoDB. Tyrimo rezultatai gali būti paveikti pasirinktų duomenų schemos specifikos ir riboto duomenų kiekio.

Raktiniai žodžiai: PostgreSQL, MongoDB, Citus, duomenų bazių skaidymas, JMeter, maišos skaidymas, diapazoninis skaidymas.

1 Įvadas

Augant duomenų kiekiui ir naudotojų skaičiui, didėja efektyvaus duomenų valdymo poreikis. Vienas iš pagrindinių būdų padidinti duomenų bazių našumą ir mastelį yra horizontalusis skaidymas (angl. *sharding*) – duomenų padalinimas tarp kelių serverių pagal tam tikrą raktą, užtikrinantis apkrovos balansavimą ir galimybę sistemą plėsti horizontaliai [1], [2].

Tyrime pasirinktos dvi skirtingų tipų duomenų bazių valdymo sistemos: PostgreSQL – atvirojo kodo **reliacinė** DBVS, pasižyminti patikimumu ir SQL suderinamumu [2]; MongoDB – **dokumentinė** NoSQL duomenų bazė, palaikanti horizontalų mastelio didinimą ir replikaciją [3]. Duomenų paskirstymui PostgreSQL aplinkoje naudotas Citus plėtinys, leidžiantis realizuoti skaidymą ir paskirstytas užklausas [4]. Visos sistemos paleistos virtualiose mašinose, naudojant Docker konteinerizacijos platformą – tai užtikrina lengvą komponentų izoliavimą ir konfigūracijos perkeliamumą [5].

DOI: https://doi.org/10.15388/LMITT.2025.30

2 Tyrimo metodologija

Eksperimentams buvo parengta OpenNebula virtualizacijos platforma. Eksperimentinė aplinka sudaryta iš virtualių mašinų (VM), naudojančių Ubuntu 24.04 LTS operacinę sistemą. Naudoti du skirtingi duomenų bazių tipai – PostgreSQL (17 versija su Citus 13.0.3 plėtiniu [4]) ir MongoDB (4.4.29 versija) [6].

Virtualių mašinų charakteristikos:

- PostgreSQL (vieno serverio konfigūracija): 2 CPU, 8 vCPU branduoliai, 10 GB RAM, 100 GB vietos saugykloje (ext4 failų sistema).
- PostgreSQL (klasterio pagrindinis serveris): 2 CPU, 8 vCPU branduoliai, 10 GB RAM, 100 GB vietos saugykloje.
- PostgreSQL (klasterio mazgai): keturios VM, kiekviena su 1 CPU, 4 vCPU branduoliais, 5 GB RAM ir 50 GB vietos saugykloje.
- MongoDB (vieno serverio konfigūracija): 2 CPU, 8 vCPU branduoliai, 10 GB RAM, 100 GB vietos saugykloje.
- MongoDB (klasterio konfigūracija): maršrutizatorius (1 CPU, 4 vCPU, 5 GB RAM, 50 GB vietos), konfigūraciniai serveriai (1 CPU, 4 vCPU, 5 GB RAM, 50 GB vietos), du duomenų mazgai (po 2 CPU 8 vCPU, 10 GB RAM, 100 GB vietos)

Visos VM naudoja Docker 26.1.3 versiją, skirtą automatizuoti programinės įrangos diegimą ir valdymą, kas leidžia efektyviai valdyti eksperimentines aplinkas, užtikrinant konfigūracijų vientisumą ir atkuriamumą [5], [2].

MongoDB klasterio architektūrai pasirinktas išskaidytas dizainas, atitinkantis tipinį replikacijos ir skaidymo modelį, kuris plačiai aprašytas mokslinėje literatūroje kaip tinkamas našumo ir atsparumo pusiausvyrai [7].

3 Duomenų struktūra ir schema

Duomenų rinkiniui pasirinkta Point-of-Sale (PoS) tipo duomenų schema. PostgreSQL lentelių struktūra išliko tokia pati tiek vieno serverio, tiek klasterio atveju, skirtumas tik tas, kad klasteryje lentelės buvo paskirstytos naudojant skaidymo metodus. MongoDB sistemai buvo sukurta analogiška dokumentų struktūra, laikantis dokumentų orientuotos paradigmos ir pritaikius JSON schemos validatorių. Esminių duomenų modelio pakeitimų, išskyrus formos keitimą iš lentelės į dokumentus, atlikta nebuvo.

Skaidymo metodai:

- Maišos (angl. *hash*) metodas: Duomenys skaidomi pagal įmonės identifikatorių (*tenant_id*). Šis metodas buvo naudotas abiejose sistemose (PostgreSQL su Citus ir MongoDB klasteryje).

- Diapazoninis (angl. *range*) metodas: Duomenys skaidomi pagal įmonės identifikatorių (*tenant_id*).

Raktas *tenant_id* pasirinktas dėl savo gebėjimo užtikrinti pakankamą entropiją (skirtingų reikšmių pasiskirstymą) tarp įrašų – tai rekomenduojama literatūroje kaip viena iš pagrindinių skaidymo rakto savybių [1], [7].

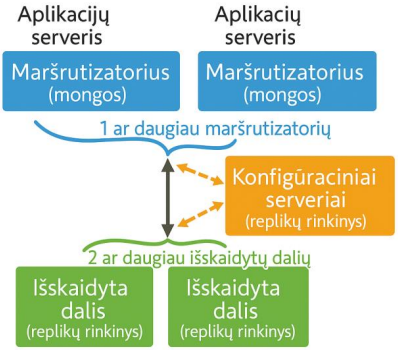
Citus klasterį sudaro koordinatoriaus mazgas, kuris valdo metaduomenis apie duomenų fragmentų (išskaidytų dalių) išdėstymą ir nukreipia užklausas, bei vienas ar daugiau darbininkų mazgų. Darbininkų mazgai saugo faktinius išskaidytų lentelių duomenis šiose išskaidytose dalyse ir atlieka didžiąją dalį duomenų apdorojimo bei skaičiavimų. [4]

PostgreSQL atveju duomenų lentelės sukurtos pagal standartinius SQL DDL sakinius su atitinkamais apribojimais ir sąsajomis. Maišos skaidymo atveju lentelės buvo paskirstytos klasteryje naudojant komandą *create_distributed_table* pagal *tenant_id.* Diapazoninio skaidymo atveju buvo panaudotas schema pagrįstas skaidymas (angl. *Schema-based sharding*), leidžiantis paskirstyti duomenis pagal schemas, kurios buvo paskirstytos pagal *tenant_id.* Taip Citus klasteryje pavyko imituoti diapazoninį skaidymą.

MongoDB išskaidytą klasterį sudaro:

- Išskaidytos dalys (angl. *Shards*): Kiekviena saugo dalį visų duomenų. Būtinai turi veikti kaip replikų rinkinys (angl. *replica set*).
- Maršrutizatorius (mongos): Veikia kaip užklausų nukreipėjas, per kurį aplikacijos jungiasi prie viso telkinio.
- Konfigūraciniai serveriai: Laiko telkinio metaduomenis ir nustatymus. Būtinai turi veikti kaip replikų rinkinys (CSRS).

1 paveikslėlis rodo, kaip šie komponentai sąveikauja tarpusavyje [6].



1 pav. MongoDB išskaidyto klasterio komponentų sąveika

MongoDB dokumentai buvo saugomi atskirose kolekcijose, naudotas JSON schemos validatorius duomenų vientisumui užtikrinti. Duomenų klasteryje naudojamas MongoDB maišos skaidymo mechanizmas pagal *tenant_id* ir diapazoninio skaidymo mechanizmas pagal *tenant_id.*

4 Duomenų įkėlimas

Eksperimentinių duomenų įkėlimas į PostgreSQL ir MongoDB duomenų bazes buvo automatizuotas naudojant Apache JMeter įrankį [8]. Naudojant parengtą testavimo scenarijų, buvo užpildytos tiek atskiros, tiek skaidytos (maišos ir diapazoniniu būdu) duomenų bazių konfigūracijos. Duomenų rinkinį sudarė lentelės ir kolekcijos, imituojančios realius Point-of-Sale (PoS) sistemos duomenis, įskaitant užsakymus, klientus, mokėjimus ir kitus susijusius objektus. Įrašų kiekis ir dydis lentelėse bei kolekcijose skyrėsi priklausomai nuo pasirinktos skaidymo strategijos ir duomenų bazės tipo.

Lentelių dydžiai PostgreSQL duomenų bazėje (vieno serverio konfigūracija) svyravo nuo kelių kilobaitų (mažiausios lentelės, pvz., *discounts* – 8 kB, *loyalties* – 1,5 MB) iki kelių gigabaitų (didžiausia lentelė *order_items* – 8,9 GB). MongoDB vieno serverio konfigūracijoje didžiausia kolekcija buvo tenants, kurios bendras dydis siekė apie 2 GB (225 MB duomenų ir 149 MB indeksų), tačiau dėl ryšio sutrikimų kai kurios lentelės (pvz., *bookings*, *order_items*, *discounts*) liko tuščios ir buvo testuojama su užpildytomis lentelėmis.

PostgreSQL su Citus maišos skaidymas sumažino didžiausios lentelės *order_items* dydį iki 5,85 GB, paskirstant apie 79 mln. įrašų. Analogiškai diapazoninis skaidymas lentelės *order_items* dydį padidino iki 6,41 GB dėl papildomų schemų administravimo duomenų.

MongoDB skaidyto klasterio maišos strategija didžiausią *order_items* kolekciją sumažino iki 610 MB, išskaidant apie 4,2 mln. dokumentų. Diapazoninis skaidymas šią kolekciją išlaikė panašaus dydžio (apie 611 MB), tačiau įrašai taip pat buvo paskirstyti tarp mazgų.

Bendras duomenų įkėlimo greitis buvo didžiausias PostgreSQL vieno serverio konfigūracijoje, tuo tarpu MongoDB vieno serverio duomenų įkėlimas vyko pastebimai lėčiau dėl ryšio stabilumo problemų ir lėtesnio dokumentų apdorojimo greičio.

Preliminarūs duomenų įkėlimo testai parodė, kad:

- PostgreSQL (vienas serveris) ir Citus klasteris buvo efektyviausi pagal duomenų įkėlimo greitį ir stabilumą.

- MongoDB įkėlimas vykdėsi lėčiau, ypač vieno serverio aplinkoje, dėl dokumentinio formato specifikos ir periodinių ryšio sutrikimų.

Tolimesniuose tyrimuose rekomenduojama detaliau analizuoti ir optimizuoti MongoDB duomenų įkėlimo parametrus bei tinklo stabilumą, siekiant išvengti aptiktų ribojimų.

5 Užklausų rezultatų analizė

Norint palyginti skirtingus skaidymo metodus buvo pasirinkta palyginti, kaip skirtingos išskaidytos duomenų bazės efektyviai veikia atliekant įvairius užklausų, apkrovos testus. Taip pat buvo palyginta ir neskaidytų duomenų bazių efektyvumas. Kiekvienas tyrimas buvo atliktas prieš tai kiekvieną duomenų bazę užpildžius apie 150 milijonų įrašų, kad būtų galima imituoti kuo realesnius atvejus, kaip siūloma skaidymo ir apkrovos analizės tyrimuose [7]. Šį tyrimą išskaidėme į šias pagrindines dalis:

5.1 Duomenų nuskaitymo, rašymo ir keitimo užklausų analizė

Šioje dalyje buvo atlikta duomenų skaitymo, rašymo ir keitimo užklausų apdorojimo greitis ir kiekis, taip pat ir klaidų kiekis per 10 minučių. Testai buvo atliekami imituojant 100, 400, 800 ir 1200 virtualių naudotojų, ir skirtingo tipo užklausomis.

1 lentelė. Bendras užklausų ir klaidų santykis skirtingose duomenų bazėse

	PostgreSQL vieno serverio	**PostgreSQL maišos funkcija išskaidyto klasterio**	**PostgreSQL diapazoniniu skaidymu išskaidyto klasterio**	**MongoDB vieno serverio**	**MongoDB maišos funkcija išskaidyto klasterio**	**MongoDB diapazoniniu skaidymu išskaidyto klasterio**
Sėkmingos užklausos	88,22 %	90,19 %	94,85 %	90,21 %	94,27 %	96,45 %
Neįvykdytos užklausos	11,78 %	9,81 %	5,15 %	9,79 %	5,83 %	3,55 %

1 lentelėje matyti, kuri dalis užklausų atitinkamose konfigūracijose yra sėkminga ir kuri buvo klaidinga dėl serverio atsako. Vieno serverio konfigūracijos (tiek PostgreSQL, tiek MongoDB) demonstruoja didesnį klaidų skaičių esant didesniam apkrovimui. Taip nutinka dėl ribotų resursų: užklau-

soms daugėjant, vienas mazgas tampa tinklo silpna vieta (angl. *bottleneck*), dėl kurio auga klaidos ir atsako vėlinimas. Klasterinėse duomenų bazėse klaidingų užklausų procentas ryškiai mažesnis, nes sistemai padeda duomenų išskaidymas tarp kelių mazgų. Tiek maišos, tiek diapazoninis skaidymas gerina apkrovos paskirstymą.

2 lentelė. PostgreSQL duomenų bazių užklausų kiekis ir atsako laikas

	Visų užklausų kiekis	**Atsako laikas t <=500 ms**	**Atsako laikas 500 ms < t <= 1500 ms**	**Atsako laikas t > 1500 ms**
Vieno neskaidyto serverio	~30,231 mln.	~26,912 mln. (89 %)	~1,901 mln. (6 %)	~1,418 mln. (5 %)
Citus maišos funkcija išskaidyto klasterio	~17,698 mln.	~15,401 mln. (87 %)	~1,402 mln. (8 %)	~0,895 mln. (5 %)
Citus diapazoniniu skaidymu išskaidyto klasterio	~45,160 mln.	~42,576 mln. (94 %)	~1,672 mln. (4 %)	~0,912 mln. (2 %)

3 lentelė. MongoDB duomenų bazių užklausų kiekis ir atsako laikas

	Visų užklausų kiekis	**Atsako laikas t <=500 ms**	**Atsako laikas 500 ms < t <= 1500 ms**	**Atsako laikas t > 1500 ms**
Vieno neskaidyto serverio	~63,672 mln.	~58,578 mln. (92 %)	~3,184 mln. (5 %)	~1,910 mln. (3 %)
Maišos funkcija išskaidyto klasterio	~70,039 mln.	~65,837 mln. (94 %)	~2,802 mln. (4 %)	~1,412 mln. (2 %)
Diapazoniniu skaidymu išskaidyto klasterio	~103,785 mln.	~99,634 mln. (96 %)	~1,925 mln. (2 %)	~2,226 mln. (2 %)

Pirmiausia šioje dalyje buvo analizuojama, kaip skirtingos duomenų bazių konfigūracijos – vieno serverio ir išskaidytos (maišos ar diapazoninio) – veikia tada, kai atliekamos skaitymo, rašymo ir keitimo užklausos. Analizuojant 2 ir 3 lenteles, galima pastebėti reikšmingus skirtumus tarp skirtingų duomenų bazių konfigūracijų efektyvumo. 2 lentelėje matome, kad PostgreSQL diapazoniškai skaidytas klasteris pasiekė didžiausią bendrą užklausų kiekį – apie 45,160 mln., iš kurių net 94% (42,576 mln.) buvo apdorotos greičiau nei per 500 ms. Tuo tarpu vieno neskaidyto serverio konfi-

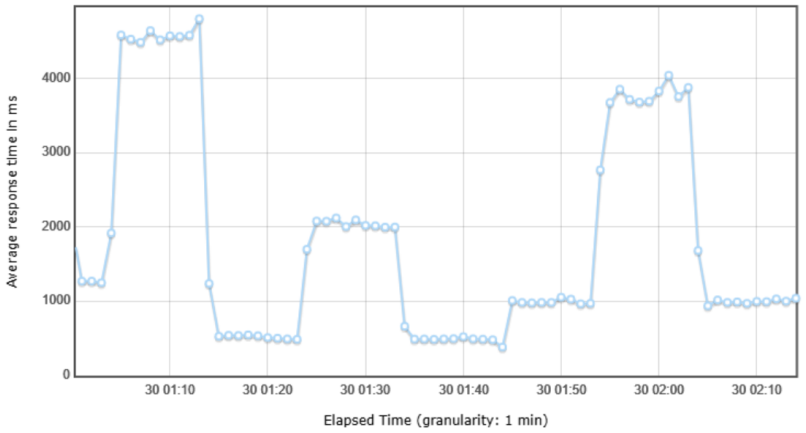
gūracija, nors ir apdorojo apie 30,231 mln. užklausų, tačiau pasiekė tik 89% užklausų su atsako laiku iki 500 ms.

Pastebėtina anomalija 2 lentelėje yra ta, kad Citus maišos funkcija išskaidyto klasterio bendras užklausų kiekis (~17,698 mln.) yra ženkliai mažesnis nei vieno neskaidyto serverio (~30,231 mln.). Toks rezultatas prieštarauja teorinėms prielaidoms, kad skaidymas turėtų padidinti bendrą sistemos pralaidumą. Šis nukrypimas gali būti susijęs su specifine maišos skaidymo implementacija Citus plėtinyje ir jos pritaikymu esamam 79 mln. įrašų duomenų kiekiui.

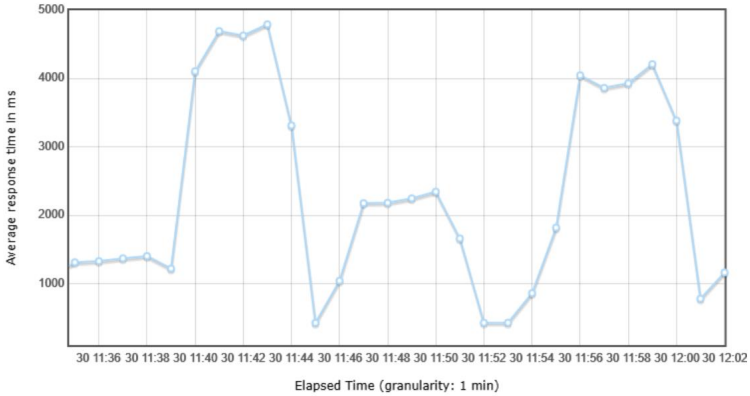
3 lentelėje pastebimos panašios, bet dar ryškesnės tendencijos MongoDB sistemoje. Diapazoniškai skaidytas klasteris pasiekė įspūdingą 103,785 mln. užklausų kiekį, iš kurių 96% (99,634 mln.) turėjo atsako laiką mažesnį nei 500 ms. Maišos funkcija išskaidytas klasteris taip pat demonstravo puikius rezultatus – 70,039 mln. užklausų su 94% (65,837 mln.) greitų atsakymų. Vieno serverio konfigūracija, nors ir parodė gerą našumą (63,672 mln. užklausų), tačiau šis rezultatas yra ženkliai mažesnis nei skaidytų sprendimų.

5.2 Didelės užklausų apkrovos testavimas

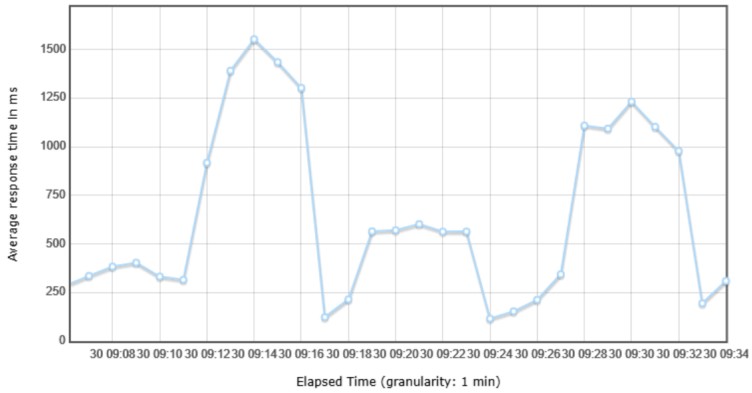
Šis testas buvo pasirinktas imituoti realią pastovią apkrovą ir kaip skirtingai išskaidytos duomenų bazės elgiasi staigiai atsirandant didėliai apkrovai. Iš pradžių testai buvo atlikti imituojant didėjantį naudotojų srautą iki 5000. Ir kartu buvo 500 ir 1000 virtualių naudotojų kiekį ir staigiai jį pakeičiant į 4 kartais didesnę apkrovą (2000 ir 4000 vartotojų).



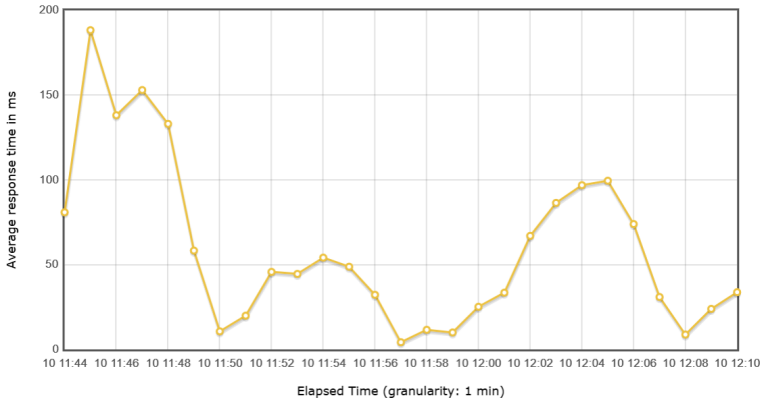
2 pav. PostgreSQL vieno serverio apkrovos testavimas



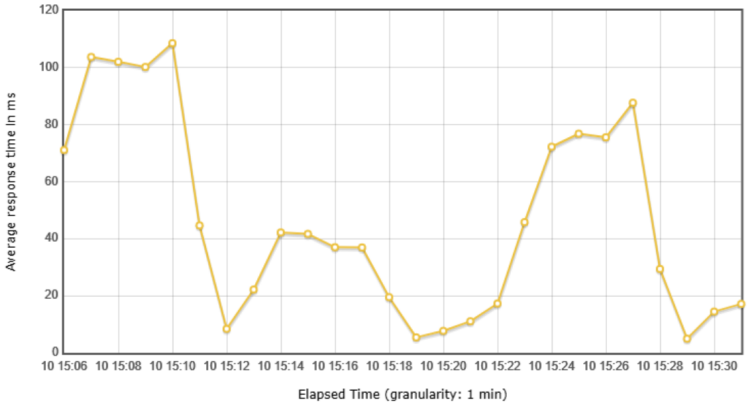
3 pav. PostgreSQL Citus maišos funkcija išskaidyto klasterio apkrovos testavimas



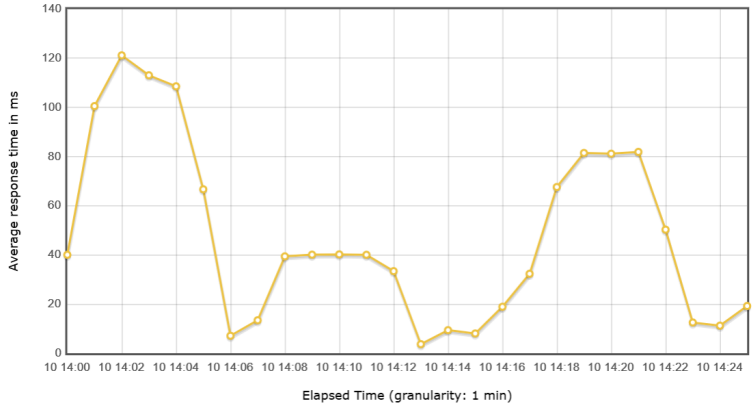
4 pav. PostgreSQL Citus diapazoniniu skaidymu išskaidyto klasterio apkrovos testavimas



5 pav. MongoDB vieno serverio apkrovos testavimas



6 pav. MongoDB maišos funkcija išskaidyto klasterio apkrovos testavimas



7 pav. MongoDB diapazoniniu skaidymu išskaidyto klasterio apkrovos testavimas

2–4 paveikslėliai atspindi PostgreSQL konfigūracijų elgseną esant didelei apkrovai. Vieno serverio (2 pav.) atsako laikas dramatiškai svyruoja nuo 1000 ms iki 4500 ms, maišos skaidymo atveju (3 pav.) išlieka panašūs svyravimai 1000-4700 ms ribose, tačiau diapazoninio skaidymo konfigūracija (4 pav.) demonstruoja ženkliai stabilesnius rezultatus – atsako laikas dažniausiai neviršija 1500 ms. Tai patvirtina diapazoninio skaidymo pranašumą PostgreSQL aplinkoje valdant kintančias apkrovas.

MongoDB konfigūracijų testavimo rezultatai (5–7 pav.) atskleidžia itin efektyvų elgesį – stebima žymiai mažesnė atsako laiko skalė. Vieno serve-

rio atveju (5 pav.) atsako laikas siekia iki 190 ms, maišos funkcija išskaidyto klasterio (6 pav.) – iki 110 ms, o diapazoninio skaidymo (7 pav.) – iki 120 ms. Lyginant su PostgreSQL, MongoDB atsako laikai yra apie 10-20 kartų mažesni, tačiau abiejose sistemose išryškėja ta pati tendencija – diapazoninis skaidymas užtikrina stabilesnį elgesį esant didelei apkrovai.

6 Išvados

1. Diapazoninis skaidymas PostgreSQL/Citus klasteryje leido pasiekti 45,16 mln. užklausų per 10 min. (94 % ≤ 500 ms), t. y. 2,5 kartus didesnį pralaidumą nei to paties klasterio maišos skaidymas (17,70 mln.) ir 49 % daugiau nei vieno serverio PostgreSQL (30,23 mln.) – tai paneigia plačiai cituotą prielaidą, kad maišos strategija Citus terpėje yra universaliai efektyviausia.
2. Maišos skaidymas Citus aplinkoje ne tik nespartino, bet 41 % sulėtino bendrą pralaidumą lyginant su neskaidyta PostgreSQL instancija, atskleisdamas iki šiol nefiksuotą koordinatoriaus – darbininkų sinchronizacijos kliūtį, pasireiškiančią esant ~79 mln. eilučių ir keturiems mazgams.
3. MongoDB diapazoninis klasteris, apdorojęs 103,79 mln. užklausų (96 % ≤ 500 ms), 48 % aplenkė to paties klasterio maišos strategiją (70,04 mln.) ir 63 % vieno serverio MongoDB (63,67 mln.), parodydamas, kad dokumentinei DB diapazoninis skaidymas gali būti pranašesnis nei maišos skaidymas.
4. Staiga padidinus apkrovą keturis kartus (1000 → 4000 vartotojų), diapazoninis skaidymas išlaikė stabilų vėlinimą (PostgreSQL ≤ 1 500 ms, MongoDB ≤ 120 ms), o maišos skaidymas PostgreSQL svyravo iki 4700 ms.
5. MongoDB klasteryje maksimalus atsako laikas (≤ 190 ms) buvo 10–20 kartų trumpesnis nei analogiškai apkrautame PostgreSQL diapazoniniame klasteryje (≤ 1 500 ms), kiekybiškai pademonstruojant dokumentinio modelio pranašumą PoS infrastruktūroje.
6. PostgreSQL hash skaidymas sumažino order_items lentelę nuo 8,9 GB iki 5,85 GB, tačiau MongoDB hash strategija tą pačią kolekciją (4,2 mln. dokumentų) suspaudė iki 610 MB – 9,6 kartų mažiau baitų vienam įrašui, atskleisdama, kad dokumentų saugykloje struktūrinis apkarpymas ir indeksų organizavimas gali lemti ženkliai geresnį talpos efektyvumą nei reliaciniame formate.

7. Tyrimas identifikavo, kad Citus diapazoniniam skaidymui teko naudoti schemomis grįstą imitaciją, kuri nepaisant papildomos metaduomenų naštos (lentelė išaugo iki 6,41 GB) vis tiek pranoko maišos skaidymą pagal pralaidumą ir stabilumą.

Tyrimo ribotumai apima konkrečią duomenų schemą, ribotą įrašų kiekį ir pasirinktą apkrovą. Tolimesniuose tyrimuose rekomenduojama išplėsti skaidymo raktų įvairovę ir taikyti išsamesnes apkrovas, siekiant patikrinti, ar nustatytos tendencijos išlieka skirtingomis sąlygomis.

Literatūra

[1] M. Tamer Özsu ir P. Valduriez, Principles of Distributed Database Systems., Springer, 2020.

[2] A. Silberschatz, H. F. Korth ir S. Sudarshan, Database System Concepts, McGraw-Hill, 2019.

[3] H. Jing, E. Haihong, L. Guan ir J. Du, „Survey on NoSQL database," *2011 6th International Conference on Pervasive Computing and Applications,* pp. 363-366, 2011.

[4] Microsoft Company, „Citus Documentation v13.0.3," 2024. [Tinkle]. Available: https://docs.citusdata.com/en/v13.0/. [Kreiptasi 2025-03-29].

[5] Docker Inc., „Docker Docs," 2025. [Tinkle]. Available: https://docs.docker.com/. [Kreiptasi 2025-03-29].

[6] MongoDB, Inc., „MongoDB Documentation v4.4," 2024. [Tinkle]. Available: https://www.mongodb.com/docs/v4.4/. [Kreiptasi 2025-03-29].

[7] C. Curino, E. Jones, Y. Zhang ir S. Madden, „Schism: a workload-driven approach to database replication and partitioning," *Proc. VLDB Endow.,* pp. 48-57, 2010.

[8] Apache Software Foundation, „Apache JMeter User Manual," 2024. [Tinkle]. Available: https://jmeter.apache.org/usermanual/. [Kreiptasi 2025-03-29].