УДК 004.056.53

Н.С. Жұматаев, А.С. Кыдырбекова*, З.З. Абдрашов, С.Т. Ахметова
PhD, доцент, М.Әуезов атындағы ОҚУ, Шымкент, Қазақстан
аға оқытушы, М.Әуезов атындағы ОҚУ, Шымкент, Қазақстан
магистрант, М.Әуезов атындағы ОҚУ, Шымкент, Қазақстан
к.ф.-м.н., доцент, М.Әуезов атындағы ОҚУ, Шымкент, Қазақстан
***Корреспондент авторы:** kas.aizat@mail.ru

СҰРАНЫСТАРДЫ ОҢТАЙЛАНДЫРУДЫҢ ЖӘНЕ УНИВЕРСИТЕТТІҢ БІЛІМ БЕРУ ПОРТАЛЫНЫҢ ДЕРЕКТЕР ҚҰРЫЛЫМЫН ӨЗГЕРТУДІҢ ТИІМДІ ӘДІСТЕРІН ЗЕРТТЕУ

Түйін

SQL дерек қорларындағы сұрауларды оңтайландыру деректердің үлкен көлемімен жұмыс істеу кезінде әзірлеушілер алдында тұрған маңызды міндет болып табылады. Мақалада индекстерді пайдалану, Sharding, агрегациялау, сондай-ақ деректерді кэштеу және репликациялау сияқты сұрау өнімділігін жақсартудың негізгі әдістері университеттің білім беру порталының деректер құрылымына өзгерістер енгізуге байланысты талқыланады. Бөлшектеу деректерді бірнеше түйіндер бойынша тиімді таратуға мүмкіндік береді, ал біріктірулер серверде күрделі есептеулерді орындауға мүмкіндік береді. Деректерді кэштеу және репликация жүйенің қол жетімділігін және ақауларға төзімділікті жақсартады. Әрбір әдіс үшін код мысалдары берілген, бұл олардың орындалуын және NoSQL жүйелерінде қолданылуын тереңірек түсінуге мүмкіндік береді. Бұл әдістерді қолдану жүйені масштабталатын және тиімдірек ететін сұрау өнімділігін айтарлықтай арттырады.

Кілттік сөздер: Сұраныстарды оңтайландыру, SQL, Sharding, агрегациялау, кэштеу, репликация.

Кіріспе

Қазіргі уақытта әлемде интернет-технологиялар мен гаджеттердің экспоненциалды дамуымен білім беру жүйесі басқаша көзқарасқа ие бола бастады. Көптеген білім беру ресурстары барлығына қолжетімді болып жатыр. Ұзақ уақыт бойы бұл саладағы барлық жобалар профессорлардың дәрістерінің бейнежазбаларына немесе оқулықтың электронды нұсқасына қол жеткізуді қамтамасыз етуден тұрды. Келесі қадам тест түрінде алынған білімді автоматты түрде тексеру мүмкіндігін қосу болды. Бұл жобалардың барлығы бастапқыда университетте пайдалануға бағытталған және бүкіл әлемнен пайдаланушылар үшін ашық ауқымды онлайн курстар деңгейіне дейін дамып, танымалдылығының қарқынды өсуін бастан кешірді.

Соңғы онжылдықтарда SQL дерекқорлары жоғары жүктемелі және масштабталатын қосымшаларды әзірлеуде кеңінен қолданыла бастады. Дәстүрлі реляциялық ДҚБЖ-дан айырмашылығы, SQL дерекқорлары үлкен көлемдегі ақпаратпен тиімді жұмыс істеуге мүмкіндік беретін икемді деректер үлгісін ұсынады. Дегенмен, бұл сұраныстарды оңтайландыру мәселесін тудырады, бұл мұндай жүйелердің өнімділігі мен ауқымдылығына тікелей әсер етеді [1].

Мәліметтер көлемінің өсу контекстінде дерекқор сұраулары тек дұрыс емес, сонымен қатар ресурстардың қарқындылығы тұрғысынан оңтайландырылған болуы керек. Негізгі міндеттердің бірі - индекстерді тиімді пайдалану, сұраныс архитектурасын дұрыс конфигурациялау және оларды минималды ресурс шығындарымен орындау. NoSQL дерекқорларындағы сұрауларды оңтайландырудың көптеген тәсілдері деректер құрылымы мен жүйенің ерекшеліктеріне байланысты өзгереді [2]. Жағдайдың күрделілігі, реляциялық

ДҚБЖ-дан айырмашылығы, SQL дерекқорлары SQL сияқты стандартты сұрау тілдерін әрдайым пайдаланбайды, бұл арнайы оңтайландыру әдістерін әзірлеуді талап етеді.

Бұл жұмыстың мақсаты NoSQL дерекқорларындағы сұрауларды оңтайландыру әдістерін, соның ішінде индекстерді пайдалануды, деректерді нормадан шығаруды және кэштеу және репликация стратегияларын талдау болып табылады. Мақалада сұраныс өнімділігін арттыруға бағытталған негізгі принциптер мен тәсілдер, сондай-ақ MongoDB, Cassandra және Redis сияқты танымал SQL дерекқорлары үшін тиімді шешімдердің мысалдары қарастырылады.

Деректер қоры құрылымы

Қазіргі ДҚБЖ деректер қорын құруға және жүргізуге арналған құралдармен жабдықталған. Бұл құралдар дерекқор әкімшілеріне арналған және кестелерді жасау және өзгерту, жазбаларды өңдеу, қатынасты шектеу және сақтық көшірмелерді басқару сияқты жалпы тапсырмаларды шешуге көмектеседі. Бұл бағдарламалардың графикалық интерфейсі болғанымен, көптеген тапсырмалар SQL тілінің командалары арқылы ыңғайлырақ шешіледі. Әрине, мұндай интерфейс іскерлік қосымшалар үшін қолайлы емес. Жаппай пайдаланушыға бағытталған жүйе соншалықты түсінікті болуы керек, бұл тақырыпты жақсы білетін пайдаланушы онымен ешқандай дайындықсыз жұмыс істей алады. Мұндай жүйелер деректерге қол жеткізуді графикалық интерфейстің артына жасырады, бұл тапсырманы тиімді шешуге мүмкіндік береді және пайдаланушыны мүмкіндігінше қателерден қорғайды.

Клиент-сервер архитектурасы бизнес-қосымшаларды жасау үшін қолданылады. ДҚБЖ ядросы серверде, ал қолданбалы бағдарлама клиентте жұмыс істейді, ал клиенттің өзі күрделі және бірнеше деңгейден тұруы мүмкін.

Бизнес логикасын серверде сақталатын процедуралар түрінде немесе жоғары деңгейлі бағдарламалау тілін қолданатын клиентте іске асыруға болады. Деректер клиент пен серверде басқаша көрсетілгендіктен, кедергінің сәйкес келмеуі мәселесі туындайды.Деректердің логикалық құрылымының сипаттамасының болуы ДҚБЖ ішінде жеткілікті күрделі деректермен манипуляциялау операцияларын орындауға мүмкіндіктер ашады. Мұндай операциялар сұраныс тілінде жазылады. Қазіргі ДҚБЖ-ға енгізілген сұрау тілдері декларативті болып табылады, яғни олар есептеулердің қажетті нәтижесін сипаттауға мүмкіндік береді, бірақ бұл есептеулерді орындау әдісін емес. Осының арқасында ДҚБЖ нәтиже алу үшін ең тиімді (кейбір өнімділік критерийлері бойынша) алгоритмдерді таңдай алады. Бұл әсіресе деректерді жаппай өңдеу кезінде пайдалы, өйткені ол бір жағынан деректер қоры сервері мен қолданба арасындағы аралық нәтижелерді тасымалдауды болдырмауға және екінші жағынан есептеулерді есепке ала отырып орындаудың оңтайлы әдісін таңдауға мүмкіндік береді. қолданбалы кодта баламалы операцияларды бағдарламалау кезінде қол жеткізу мүмкін емес нақты сақталған деректердің сипаттамалары.

Қазіргі заманғы ДҚБЖ, соның ішінде PostgreSQL қамтамасыз ететін декларациялық сұрауларды өңдеудің қуатты мүмкіндіктері жиі пайдаланылмайды. Бұл көптеген себептерге байланысты болады, олардың арасында сұраныс тілдерінің декларативті құралдарының кеңінен қолданылатын бағдарламалау тілдерінің императивті құралдарымен нашар үйлесімділігін атап өтуге болады.

Сұраныс тілдерінің мүмкіндіктерін пайдаланбай қосымшаларды әзірлеу тәжірибесі мұндай құралдарды қамтамасыз етпейтін бірқатар жүйелердің пайда болуына әкелді (NoSQL жүйелері деп аталады). Мұндай жүйелерді деректерді сақтау үшін пайдаланған кезде, ДҚБЖ функцияларының бір бөлігі қосымшаға беріледі. Бұл қолданбаның күрделілігі мен оны әзірлеу құнының артуына немесе сапаның төмендеуіне әкелуі мүмкін - бұл көп жағдайда қолайлы болып шығады.

SQL дерекқорларындағы деректердің үлкен көлемімен тиімді жұмыс істеу

SQL дерекқорларындағы деректердің үлкен көлемімен тиімді жұмыс істеу үшін маңызды аспект дұрыс конфигурациялау және бөлуді пайдалану, сондай-ақ кеңейтілген біріктіру әдістері болып табылады. Деректер бір түйін үшін тым үлкен болғанда, бөлшектеу жүктемені бірнеше серверлер бойынша таратуға мүмкіндік береді, жүйе өнімділігін арттырады және бір түйінді шамадан тыс жүктеуді болдырмауға көмектеседі [3].

Sharding деректерді әртүрлі серверлерде сақтауға болатын бөліктер деп аталатын бірнеше бөліктерге бөлуге мүмкіндік береді. Бұл деректердің үлкен көлемін қамтитын жинақтармен жұмыс істегенде өте маңызды [4]. Мысалы, MongoDB жүйесінде аймақ сияқты өріс арқылы бөлу сол өрісті сүзгі ретінде пайдаланатын сұрауларды айтарлықтай жылдамдатады.

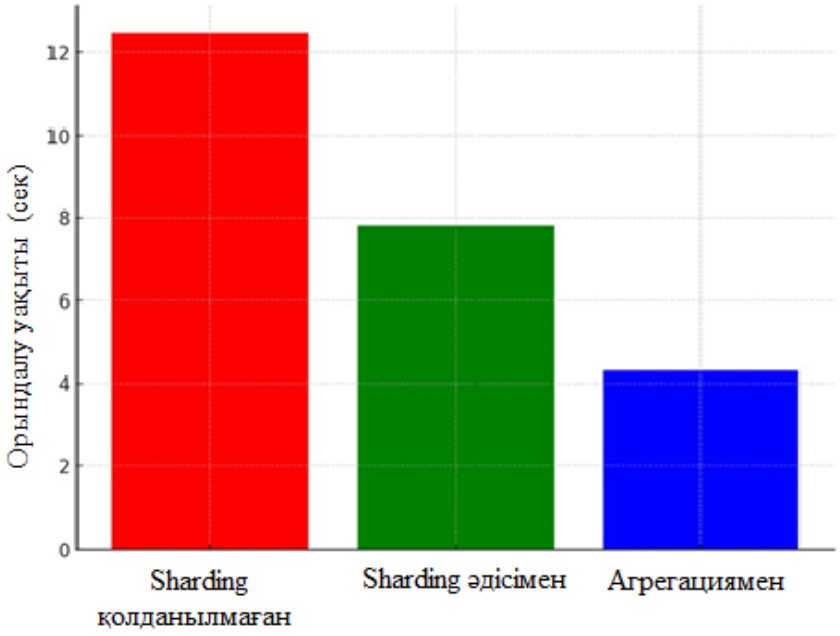
Деректерді өнімдер мен аймақтар бойынша сүзетін, топтайтын және сұрыптайтын күрделі сұрауды іске қосу үшін бөлу және біріктіруді пайдалану мысалын қарастырайық:

```
//Жинақты өріс бойынша Sharding 'region' Сұраныстарды оңтайландыруда
sh.shardCollection("sales_data", { "region": 1 });
// Бөлшектеу арқылы күрделі жиынтық сұрауды іске қосу
db.sales_data.aggregate([
{
$match: {
region: "North America",
date: { $gte: new ISODate("2022-01-01"), $lte: new ISODate("2022-12-31") }
}
},
{
$group: {
_id: "$product_id", // группасағы бойынша топтастыру
 total_sales: { $sum: "$amount" }, // алынған бағасы
avg_sales: { $avg: "$amount" }                    // Орташа балы
},
{
$sort: { total_sales: -1 }  // орта балл бойынша топтастыру
},
{
$limit: 10  // бағалары ьойынша шектеу
}
]);
}
```

Бұл код алдымен MongoDB-ге деректерді бірнеше түйіндер бойынша тиімді таратуға мүмкіндік беретін деректерді бөлу үшін аймақ өрісінде индекс жасайды. Содан кейін деректерді аймақ және күн бойынша сүзгіден өткізетін, өнімдер бойынша топтастыратын және студенттің жалпы және орташа балын есептейтін жинақтау орындалады. Осыдан кейін ол жалпы балдар бойынша сұрыптайды және нәтижені алғашқы 10 өніммен шектейді.

Sharding мұндай сұрауларды орындау кезінде өнімділікті айтарлықтай жақсартады, өйткені ол жүйеге деректердің кішірек бөліктерімен жұмыс істеуге мүмкіндік береді, бұл бір сервердегі жүктемені азайтады. Агрегациямен біріктірілген бұл күрделі операцияларды тікелей дерекқорда орындауға мүмкіндік береді, жауап беру уақытын және тасымалданатын деректер көлемін азайтады [5].

Түсінікті болу үшін 1-сурет MongoDB жүйесіндегі сұраныс өнімділігіне Sharding мен біріктіруді пайдаланудың әсерін көрсетеді.



Сурет 1. Sharding мен біріктірудің сұрау өнімділігіне әсері

Бұл график бөлу және біріктіруді пайдалану сұрауды орындау уақытын қысқартуға қалай әсер ететінін анық көрсетеді. Sharding пайдалану деректерді жылдам өңдеуге көмектеседі, ал біріктіру серверде есептеулерді орындауға мүмкіндік береді, клиент қолданбасындағы жүктемені азайтады және жауап беру уақытын жылдамдатады.

Sharding мен біріктіру сияқты әдістерді пайдалану деректердің үлкен көлемімен жұмыс істеу кезінде NoSQL дерекқорларындағы жоғары өнімділікті қамтамасыз ету үшін өте маңызды, бұл тұтастай жүйенің масштабталуы мен тиімділігін арттырады [6].

Сұраныс нәтижелерін жақсарту үшін деректерді кештеу мен репликациялау

MongoDB және Redis сияқты NoSQL жүйелеріндегі сұрау өнімділігін одан әрі жақсарту үшін деректерді кэштеу және репликациялау сияқты әдістерге ерекше назар аудару керек. Кэштеу жиі орындалатын сұраныстардың нәтижелерін жадта сақтауға мүмкіндік береді, бұл қайталанатын сұрауларға жауап беру уақытын айтарлықтай қысқартады [7]. Redis негізгі дерекқорды қайта баспай-ақ ақпаратты жылдам алуға мүмкіндік беретін кілт-мән пішімінде деректерді сақтау үшін кэштеуді пайдаланады. Бұл әсіресе деректер сирек өзгергенде және қайталанатын сұрауларға тікелей кэштен қызмет көрсетуге болатын кезде пайдалы. Redis көмегімен сұраныстарды кэштеу үшін мысал коды:

```
import redis
# Redis  байланыс орнату
r = redis.Redis(host='localhost', port=6379, db=0)
# Кэште деректер бар-жоғын тексеріміз cached_data = r.get('user:1000')
if cached_data:
# егер кэште мәліметтер болса, оларды пайдаланамыз
print("кэштегі мәліметтер:", cached_data)
else:
```

```
# Деректер кэште болмаса, біз оны дерекқордан аламыз және оны кэштейміз
user_data = get_user_data_from_db(1000)  # Дерекқордан деректерді алу функциясы

r.set('user:1000', user_data) # Кэшируем данные print("Деректер базасынан алынған
деректер:", user_data)
```

Бұл мысалда қажет деректердің Redis кэшінде бар-жоғын тексереміз. Деректер бар болса, ол негізгі дерекқорға қатынаусыз жылдам шығарылады. Деректер болмаса, ол дерекқордан жүктеледі, содан кейін кейінгі сұраулар үшін кэште сақталады. Өнімділікті жақсартудың тағы бір маңызды элементі деректерді репликациялау болып табылады.

NoSQL жүйелеріндегі репликация деректердің бірнеше түйіндерде көшірмелерін жасау үшін қолданылады, бұл деректердің қолжетімділігін арттырады және жүйе ақауларына төзімділікті арттырадым[8]. Репликация сұрауларды бірнеше түйіндер бойынша таратуға мүмкіндік береді, бұл бір сервердегі жүктемені азайтады және жауап беру уақытын жылдамдатады. MongoDB жүйесінде репликация деректердің бірнеше көшірмелерін сақтауға мүмкіндік беретін Replica Set жасау арқылы жүзеге асады. Бір реплика сәтсіз болғанда, екіншісі жүктемені автоматты түрде қабылдайды, үздіксіз жұмысты қамтамасыз етеді.

ҚОРЫТЫНДЫ

NoSQL дерекқорларындағы сұрауларды оңтайландыру деректердің үлкен көлемімен жұмыс істеу кезінде жоғары өнімділікті қамтамасыз етудің негізгі аспектісі болып табылады. Мақалада индекстерді пайдалану, бөлшектеу, біріктіру, сондай-ақ деректерді кэштеу және репликациялау сияқты негізгі оңтайландыру әдістері қарастырылады. Осы әдістердің әрқайсысы жүйе өнімділігін айтарлықтай жақсарта алады, жауап беру уақытын қысқартады және дерекқордың ауқымдылығын арттырады.

Деректерді бірнеше серверлер бойынша тиімді таратуға мүмкіндік беретін sharding және күрделі есептеулерді тікелей дерекқорда орындауға мүмкіндік беретін біріктірулерге ерекше назар аударылады. Бұл әдістерді қолдану жүйеге жүктемені айтарлықтай азайтуға және сұраныстарды өңдеуді жеделдетуге мүмкіндік береді, әсіресе деректердің үлкен көлемі жағдайында.

Бұған қоса, кэштеу және репликация сияқты технологияларды пайдалану деректердің қолжетімділігін жақсартуға және жүйе ақауларына төзімділікті арттыруға көмектеседі. Бұл әдістер бірге өнімділігі жоғары және масштабталатын NoSQL дерекқорларын жасауға көмектесетін сұраныстарды оңтайландырудың кешенді тәсілін құрайды.

Әдебиеттер тізімі

1. Шичкина Ю.А., Ха В.М. Орындалған сұраныстарды ескере отырып, негізгі құжат типті дерекқорлар үшін енгізілген құжаттары бар жинақтарды құру әдісі // Информатика және автоматика. 2020. 19-том. № 4. Р. 829-854.
2. Гуляев А.Г. Үлкен деректер бойынша SQL сұраныстарын оңтайландыру әдісін әзірлеу // Инновациялар. Ғылым. Білім. 2021. № 38. Б. 739-745.
3. Петров Я.А., Степанов С.Ю., Вагизов М.Р., Кардаполова В.Ю. Қазіргі заманғы географиялық ақпараттық жүйелерде NoSQL және үлкен мәліметтерді қолдану тенденциялары туралы // Ақпараттық технологиялар және жүйелер: менеджмент, экономика, көлік, құқық. 2020. № 1. 141-146 б.
4. Лузянин А.В. Үлкен көлемдегі мәліметтерді оңтайландыру және сақтау әдістері // Математикалық модельдеу және ақпараттық технологиялар бойынша жас ғалымдардың XXII Бүкілресейлік конференциясының тезистері. 2021. 55-56-беттер.

5. Әлібиева Ж., Мұқажанов Н., Черикбаева Л., Ерімбетова А., Байымбетов Д. NoSQL бағаналы деректер қорының мүмкіндіктерін салыстыру // ҚазАТК Хабаршысы. 2024. Т. 131. № 2. Б. 350-358.
6. Гармашев М.А., Резников Н.Г. NoSQL технологиясы: техникалық ерекшеліктері, даму перспективалары және SQL-ге қарағанда салыстырмалы артықшылықтары // Ғылыми нәтиже. Ақпараттық технология. 2023. Т. 8. № 3. 56-62 б.
7. Лисин В.А., Серы А.С., Сидорова Е.А. Графиктік деректер базасына негізделген пәндік облыстардың онтологиясын көрсету моделі // Новосибирск мемлекеттік университетінің хабаршысы. Серия: Ақпараттық технологиялар. 2022. 20-том. № 4. 24-38 б.
8. Мухин А.М., Генаев М.А., Расқазов Д.А., Лашын С.А., Афонников Д.А. RDBMS және NoSQL тәсілдерін гибридті қолдану негізінде транскриптомдық деректерді құрылымдау және өңдеу технологиясы // Математикалық биология және биоинформатика. 2020. 15-том. № 2. Б. 455-470.

References

1. Shichkina Ju.A., Ha V.M. Oryndalған sұranystardy eskere otyryp, negizgi құzhat tipti derekқorlar үshin engizilgen құzhattary bar zhinaқtardy құru ədisi // Informatika zhəne avtomatika. 2020. 19-tom. № 4. P. 829-854.
2. Guljaev A.G. Ylken derekter bojynsha SQL sұranystaryn oңtajlandyru ədisin əzirleu // Innovacijalar. Ғylym. Bilim. 2021. № 38. B. 739-745.
3. Petrov Ja.A., Stepanov S.Ju., Vagizov M.R., Kardapolova V.Ju. Қazirgi zamanғy geografijalyқ aқparattyқ zhүjelerde NoSQL zhəne үlken məlimetterdi қoldanu tendencijalary turaly // Aқparattyқ tehnologijalar zhəne zhүjeler: menedzhment, jekonomika, kөlik, құқyқ. 2020. № 1. 141-146 b.
4. Luzjanin A.V. Ylken kөlemdegi məlimetterdi oңtajlandyru zhəne saқtau ədisteri // Matematikalyқ model'deu zhəne aқparattyқ tehnologijalar bojynsha zhas ғalymdardyң HHII Bүkilresejlik konferencijasynyң tezisteri. 2021. 55-56-better.
5. Əlibieva Zh., Mұқazhanov N., Cherikbaeva L., Erimbetova A., Bajymbetov D. NoSQL bағanaly derekter қorynyң mүmkindikterin salystyru // ҚazATK Habarshysy. 2024. T. 131. № 2. B. 350-358.
6. Garmashev M.A., Reznikov N.G. NoSQL tehnologijasy: tehnikalyқ erekshelikteri, damu perspektivalary zhəne SQL-ge қarағanda salystyrmaly artyқshylyқtary // Ғylymi nətizhe. Aқparattyқ tehnologija. 2023. T. 8. № 3. 56-62 b.
7. Lisin V.A., Sery A.S., Sidorova E.A. Grafiktik derekter bazasyna negizdelgen pəndik oblystardyң ontologijasyn kөrsetu modeli // Novosibirsk memlekettik universitetinің habarshysy. Serija: Aқparattyқ tehnologijalar. 2022. 20-tom. № 4. 24-38 b.
8. Muhin A.M., Genaev M.A., Rasқazov D.A., Lashyn S.A., Afonnikov D.A. RDBMS zhəne NoSQL təsilderin gibridti қoldanu negizinde transkriptomdyқ derekterdi құrylymdau zhəne өңdeu tehnologijasy // Matematikalyқ biologija zhəne bioinformatika. 2020. 15-tom. № 2. B. 455-470.

Н. С. Жуматаев, А. С. Кыдырбекова*, З. З. Абдрашов, С. Т. Ахметова
PhD, доцент, ЮКУ им. М. Ауэзова, Шымкент, Казахстан
старший преподаватель, ЮКУ им. М. Ауэзова, Шымкент, Казахстан
магистрант, ЮКУ им. М. Ауэзова, Шымкент, Казахстан
к.ф.-м.н., доцент, ЮКУ им. М. Ауэзова, Шымкент, Казахстан
*__Автор для корреспонденции:__ kas.aizat@mail.ru

ИССЛЕДОВАНИЕ ЭФФЕКТИВНЫХ МЕТОДОВ ОПТИМИЗАЦИИ ЗАПРОСОВ И ИЗМЕНЕНИЯ СТРУКТУРЫ ДАННЫХ ОБРАЗОВАТЕЛЬНОГО ПОРТАЛА УНИВЕРСИТЕТА

Аннотация

Оптимизация запросов в базах данных SQL — важная задача, стоящая перед разработчиками при работе с большими объемами данных. В статье рассматриваются ключевые методы повышения производительности запросов, такие как использование индексов, шардинга, агрегации, а также кэширования и репликации данных, в связи с изменением структуры данных образовательного портала университета. Фрагментация позволяет эффективно распределять данные по нескольким узлам, а агрегация позволяет выполнять сложные вычисления на сервере. Кэширование и репликация данных повышают доступность и отказоустойчивость системы. Для каждого метода приведены примеры кода, позволяющие глубже понять их реализацию и использование в системах NoSQL. Использование этих методов значительно повышает производительность запросов, делая систему более масштабируемой и эффективной.

Ключевые слова: оптимизация запросов, SQL, Sharding, агрегация, кэширование, репликация.

N.S. Zhumataev, A.S. Kydyrbekova*, Z.Z. Abdrashov, S.T. Akhmetova
PhD, associate professor, M. Auezov SKU, Shymkent, Kazakhstan
senior lecturer, M. Auezov SKU, Shymkent, Kazakhstan
master's student, M. Auezov SKU, Shymkent, Kazakhstan
Cand. Phys.-Math. Sci., associate professor, M. Auezov SKU, Shymkent, Kazakhstan
*__Corresponding author's email:__ kas.aizat@mail.ru

STUDY OF EFFECTIVE METHODS FOR OPTIMIZING REQUESTS AND CHANGING THE DATA STRUCTURE OF THE UNIVERSITY EDUCATIONAL PORTAL

Abstract

Query optimization in SQL databases is an important task facing developers when working with large volumes of data. The article discusses key methods for improving query performance, such as the use of indexes, sharding, aggregation, as well as data caching and replication, in connection with the change in the data structure of the university educational portal. Fragmentation allows you to effectively distribute data across multiple nodes, and aggregation allows you to perform complex calculations on the server. Data caching and replication increase the availability and fault tolerance of the system. Code examples are provided for each method, allowing you to better understand their implementation and use in NoSQL systems. Using these methods significantly improves query performance, making the system more scalable and efficient.

Keywords: query optimization, SQL, Sharding, aggregation, caching, replication.