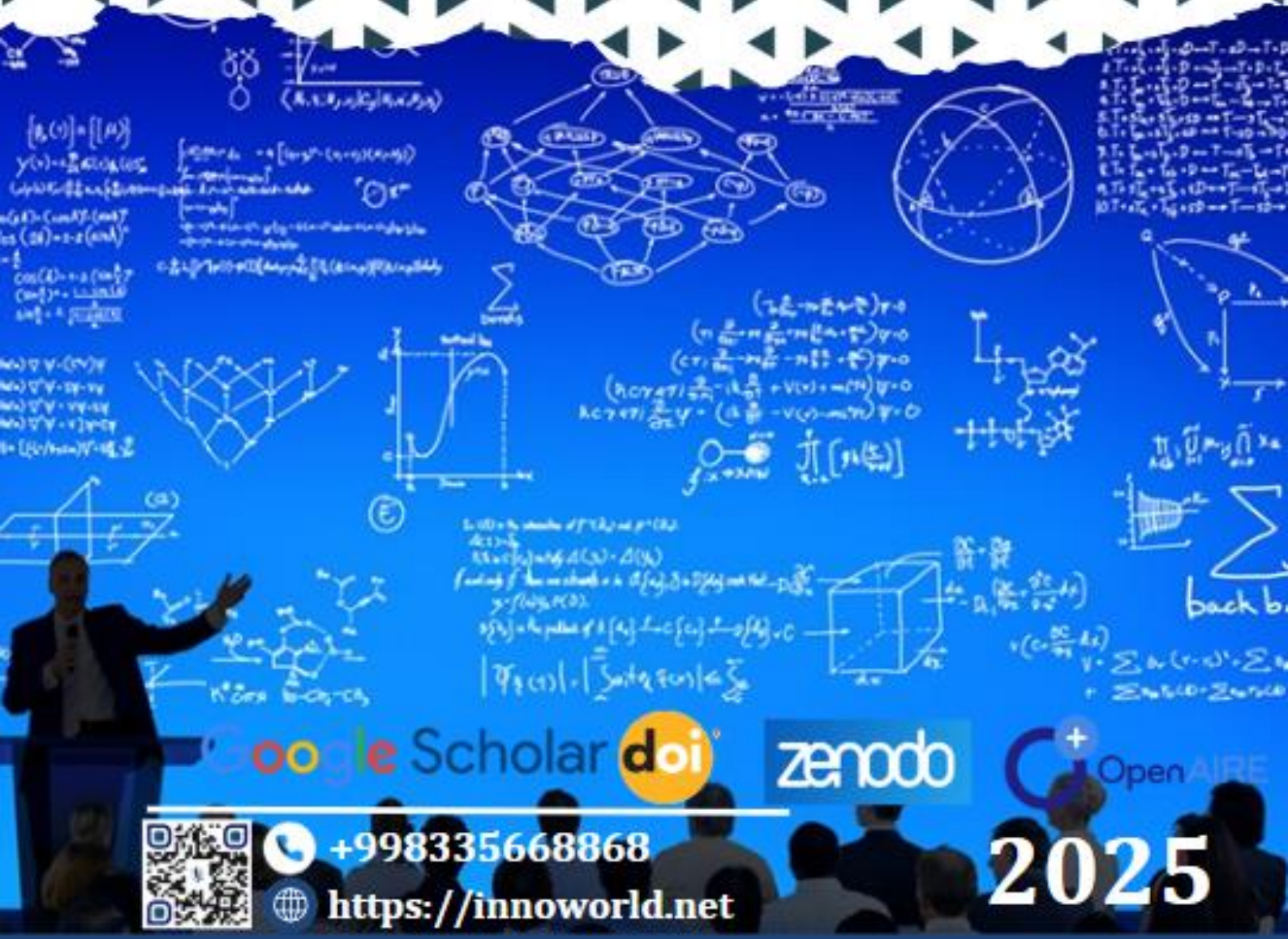
INNOVATIVE WORLD
Ilmiy tadqiqotlar markazi

ZAMONAVIY ILM-FAN VA TA'LIM: MUAMMO VA YECHIMLAR

ILMIY-AMALIY KONFERENSIYA



+998335668868
https://innoworld.net
2025

«INNOVATIVE WORLD» ILMIY TADQIQOTLARNI QO'LLAB-QUVVATLASH MARKAZI

«ZAMONAVIY ILM-FAN VA TADQIQOTLAR: MUAMMO VA YECHIMLAR» NOMLI 2025-YIL № 5-SONLI ILMIY, MASOFAVIY, ONLAYN KONFERENSIYASI

ILMIY-ONLAYN KONFERENSIYA TO'PLAMI
СБОРНИК НАУЧНЫХ-ОНЛАЙН КОНФЕРЕНЦИЙ
SCIENTIFIC-ONLINE CONFERENCE COLLECTION

www.innoworld.net
O'ZBEKISTON-2025

MYSQLDA ALGORITMLARNING AHAMIYATI

Abduvahhobov Muhammadqodir Abrorjon o'g'li
Farg'ona Davlat texnika universiteti ATT fakulteti 640-24 Axborot xavfsizligi
muhammadqodirabduvahhobov@gmail.com, +998 94 275 07 06
Abdusattorov Faxriyor Po'latjon o'g'li
abdusattorovfaxriyor4@gmail.com, +998 94 445 07 30

ANNOTATSIYA. Ushbu maqola MySQL ma'lumotlar bazasi samaradorligini oshirishda indekslash, qidirish va ketma-ket o'qish operatsiyalari va tranzaksiyalar boshqaruvi va ma'lumotlar yaxlitligini ta'minlovchi qulflash (Locking) ahamiyati tahlil qilingan. MySQL administratorlari va ishlab chiquvchilari uchun indekslash strategiyalarini optimallashtirish va yuqori unumdorlikka erishish bo'yicha amaliy tavsiyalar berilgan.

АННОТАЦИЯ. В этой статье анализируется важность индексирования, поиска и последовательного чтения, а также управления транзакциями и блокировок для обеспечения целостности данных для повышения производительности базы данных MySQL. В статье представлены практические рекомендации для администраторов и разработчиков MySQL по оптимизации стратегий индексирования и достижению высокой производительности

ABSTRACT. This article analyzes the importance of indexing, searching, and sequential read operations, as well as transaction management and locking to ensure data integrity, in improving MySQL database performance. It provides practical recommendations for MySQL administrators and developers to optimize indexing strategies and achieve high performance.

Kalit so'zlar: MySQL, B+ Tree, optimallashtirish, Indekslash, CBO, samaradorlik, InnoDB, Tranzaksiya.

Ключевые слова: MySQL, B+ Tree, оптимизация, индексирование, CBO, эффективность, InnoDB, транзакции

Keywords: MySQL, B+ Tree, optimization, Indexing, CBO, efficiency, InnoDB, Transaction.

Kirish. Bugungi raqamli dunyoda, smartfon ilovasi bo'ladimi, yirik elektron tijorat sayti bo'ladimi yoki yirik korxonaning ichki hisob-kitob tizimi bo'ladimi — barcha ish jarayonlarining ostida ma'lumotlar bazasi yotadi. Ayniqsa, global miqyosda eng ko'p ishlatiladigan ma'lumotlar bazasi tizimlaridan biri bo'lgan MySQL ning ishlash tezligi va ishonchliligi — bu butun tizimning muvaffaqiyatini belgilaydi.

So'nggi yillarda ma'lumotlar hajmi shunchalik tez o'sib boryaptiki (buni "Katta ma'lumotlar" (Big Data) davri deb atashadi), oddiy so'rov yozish endi yetarli emas. Agar sizning ma'lumotlar bazangizda milliardlab ma'lumotlar bo'lsa, u yerda kerakli ma'lumotni topish uchun atigi bir necha millisekund vaqt ketishi kerak. Agar bu jarayon sekinlashsa, mijoz sizning saytingizni tark etadi, yoki bank tranzaksiyasi kechikib ketadi.

Bu yerda biz maqolamizning asosiy mavzusiga, ya'ni algoritmlarga yetib kelamiz. Ko'pchilik dasturchilar uchun MySQL — bu shunchaki SELECT * FROM table WHERE... kabi SQL so'rovlarini tushunadigan "qora quti". Ammo bu "quti" ichida ma'lumotlarni aql bilan boshqaruvchi murakkab "miya" yashiringan.

Bu "miya" qanday ishlaydi? Misol uchun:

Qanday saqlash kerak? Ma'lumotlarni disklarda qanday joylashtirish kerakki, ularni topish iloji boricha tez bo'lsin. Bu yerda B+ Tree kabi aqlli indekslash algoritmlari ishga tushadi.

Qanday topish kerak? Bir so'rovni bajarishning ming xil yo'li bo'lishi mumkin. So'rov Optimallashtiruvchisi qaysi yo'l eng tez (eng arzon) ekanligini aniqlaydi. Bu jarayon Kosmisiyaga asoslangan optimallashtirish (CBO) deb nomlanuvchi algoritm bilan amalga oshiriladi.

To'qnashuvni qanday hal qilish kerak? Agar bir vaqtning o'zida minglab foydalanuvchi bitta ma'lumotni o'zgartirmoqchi bo'lsa, ma'lumotlar bir-biriga aralashib ketmasligi kerak. Bu vazifani esa qulflash (Locking) mexanizmlari tartibga soladi.

Shunday qilib, ushbu maqolaning asosiy maqsadi — MySQL ning ishlash samaradorligini ta'minlovchi mana shu "yashirin qahramonlar" (algoritmlar) bilan tanishishdir. Biz ularning ishlash prinsipini tushunib, bu bilimlarni amalda qanday qo'llashni o'rganamiz: Ma'lumotlar bazasi administratorlari va ishlab chiquvchilari qanday qilib o'z indekslash strategiyalarini va SQL so'rovlarini to'g'ri algoritmlar bilan uyg'unlashtirib, tizimdan maksimal unumdorlikka erishishlari mumkinligini ko'rsatib beramiz.

Biz hozirda "Ma'lumotlar Inqilobi" deb ataladigan davrda yashayapmiz. Har bir smartfon tugmasi bosilganda, har bir onlayn xarid amalga oshirilganda, yoki har bir ijtimoiy tarmoq xabari yuborilganda — milliardlab ma'lumotlar to'planadi. Shu sababli, zamonaviy ilovalar va korporativ tizimlarning ishlash samaradorligi nafaqat qulay interfeysga, balki ushbu katta hajmli ma'lumotlarni (Big Data) qanchalik tez va ishonchli boshqara olishimizga bog'liq bo'lib qoldi.

Bu jarayonning markazida Ma'lumotlar Bazasini Boshqarish Tizimlari (DBMS) turadi. Ular ma'lumotlarni tartiblash, saqlash va kerak bo'lganda bir zumda yetkazib berish uchun javobgardir. Aytish mumkinki, ma'lumotlar bazasining tezkorligi — bu kompaniyaning raqobatbardoshligini, mijozning esa qoniqish darajasini belgilovchi asosiy ko'rsatkichdir.

Shu nuqtai nazardan, MySQL uzoq yillardan beri o'zining ishonchliligi, ochiq manbaliligi va katta jamoasi tufayli dunyodagi eng mashhur Relyatsion Ma'lumotlar Bazasini Boshqaruv Tizimlaridan (RDBMS) biri bo'lib kelmoqda. U kichik loyihalardan tortib, yirik global platformalarning yukini ko'tarib keladi.

Biroq, bu tizimning potentsialini to'liq ochish uchun, faqatgina tashqi ko'rinishga — ya'ni, SQL so'rovlariga yoki jadval dizayniga e'tibor qaratish yetarli emas. MySQL ning haqiqiy kuchi va unumdorligi uning ichki tuzilishida yashiringan — ma'lumotlar bazasining ichki algoritmik tuzilishi uning samaradorligining asosiy omili hisoblanadi. Dasturchilar bu yashirin mexanizmlarni (masalan, indeks qanday ishlashi yoki so'rov qanday optimallashtirilishi) tushunmas ekan, hatto ideal yozilgan SQL so'rovi ham sekin

ishlashi mumkin. Shu bois, maqolamizda biz aynan shu ichki "mexanizmlar"ni o'rganish orqali MySQL'dan maksimal unumdorlik olish yo'llarini tahlil qilamiz.

Ushbu tadqiqot ishining asosiy maqsadi — ma'lumotlar bazasi mutaxassislari va ishlab chiquvchilari uchun MySQL tizimining "kapoti ostida" nimalar sodir bo'lishini ochib berishdan iborat. Bizning asosiy e'tiborimiz MySQL ning ichki mexanizmlarida qo'llaniladigan asosiy algoritmlarni chuqur ilmiy-amaliy tahlil qilishga qaratilgan.

Maqolada, biz uchta asosiy yo'nalishni yoritamiz:

Ma'lumotlarni Saqlash Algoritmlari: InnoDB kabi saqlash mexanizmlarining tezkor qidiruvni qanday ta'minlashini (masalan, B+ Tree algoritmi) va ma'lumotlar yaxlitligi uchun tranzaksiyalarni boshqarish usullarini tahlil qilamiz.

So'rov Optimallashtirish Algoritmlari: So'rov Optimallashtiruvchisi (Query Optimizer) qanday qilib yuzlab mumkin bo'lgan yo'llar ichidan eng tezkorini tanlashini (Kosmisiyaga asoslangan optimallashtirish - CBO) tushuntiramiz.

Amaliy Qo'llash: Eng muhimi, biz ushbu nazariy bilimlarni to'g'ridan-to'g'ri amaliyotga tatbiq etish yo'llarini ko'rsatamiz. Maqola yakunida, ishlab chiquvchilar va administratorlar indekslash strategiyalarini va SQL so'rovlarini ushbu algoritmlar bilan uyg'unlashtirib, o'z tizimlaridan maksimal samaradorlikka erishishlari uchun aniq va isbotlangan amaliy tavsiyalar ishlab chiqiladi.

MySQL'dan biz so'rov yuborganimizda, u orqada ma'lumotlarni qanday saqlashi va qanday himoyalashi to'g'risidagi qoidalar Saqlash Mexanizmlari (Storage Engines) deb ataladi. Ular ichida eng ishonchli va tezkor tashkilotchi — bu InnoDB. InnoDB'ning barcha mo'jizaviy tezlik va ishonchlilik sirlari uning ishlatadigan ikkita asosiy algoritmlar guruhida yashiringan.

1. InnoDB'ning Bosh Qomusi: B+ Tree Algoritmi

Tasavvur qiling, sizga milliardlab odamning ism-sharifi yozilgan katta qog'oz kitobdan bitta ma'lumotni topish kerak. Agar kitob tartibsiz bo'lsa, siz soatlab qidirasiz. InnoDB bu muammoni B+ Tree (B-plus daraxti) deb nomlangan aqlli algoritm yordamida hal qiladi.

B+ Tree nima qiladi?

B+ Tree ma'lumotlarni shunday tartiblaydiki, siz istalgan ma'lumotni darhol topa olasiz:

Tezkor Topish Sirti: Bu daraxt kitobdagi Alfavit Ko'rsatkichiga (Index) o'xshaydi, lekin undan ham yaxshiroq. Bu daraxt juda "yassi" va "keng" bo'ladi. Masalan, millionlab yozuvga ega bo'lsangiz ham, siz izlayotgan ma'lumotni topish uchun atigi 3-4 qadamda (daraxtning yuqorisidan pastigacha) yetib borasiz. Bu tezlik shundan kelib chiqadiki, MySQL diskdan bir safar ma'lumot olganida, faqat bitta ma'lumotni emas, balki bir nechta qo'shni ma'lumotlarni ham olib, xotirada saqlab qo'yadi. Bu "diskka kirish" (Disk I/O) sonini juda kamaytiradi, natijada ish tezlashadi.

Ketma-ketlikning Qulayligi: B+ Tree'ning eng muhim qismi shundaki, barcha ma'lumotlar (daraxtning faqat eng pastki qavatida joylashganlari) bir-biri bilan zanjirdek bog'langan. Shuning uchun, agar sizga "100 dan 200 gacha

narxdagi barcha mahsulotlarni top" degan so'rov kelsa, MySQL daraxtni boshidan oxirigacha aylanish o'rniga, shunchaki 100 ni topadi-da, zanjir bo'ylab 200 gacha tezda o'qib ketadi.

Bu algoritm o'qishda ajoyib bo'lsa-da, yangi ma'lumot kiritish, o'chirish yoki o'zgartirish (INSERT, DELETE, UPDATE) vaqtida biroz "vaqt yo'qotadi". Sababi, ma'lumot kiritilganda, daraxt o'zining zo'r tartibini saqlab qolish uchun qayta tuzilishi, ba'zan "bo'linishi" kerak bo'ladi. Bu esa qo'shimcha ish talab qiladi.

Ma'lumotlar Soqchisi: Qulflash (Locking) Algoritmlari

Tasavvur qiling, bank hisobingizda 1000 so'm bor. Bir vaqtning o'zida siz smartfoningiz orqali 500 so'm to'lov qilyapsiz, boshqa do'stingiz esa sizga 500 so'm o'tkazmoqchi. Agar bu ikki jarayon bir-biriga xalaqit bersa, hisobingiz to'g'ri ishlamay qoladi.

Bu kabi to'qnashuvlarning oldini olish uchun InnoDB Qulflash (Locking) Algoritmlarini ishlatadi.

Ikki Bosqichli Qulflash (Two-Phase Locking - 2PL): Bu shuni anglatadiki, biror to'lov jarayoni boshlanganida, u o'ziga kerakli ma'lumotni "qulflab oladi". Bu o'sish bosqichi. Boshqa hech kim uni o'zgartira olmaydi. Jarayon tugagandan keyingina u ma'lumotni "ochib beradi" (pasayish bosqichi). Bu tizim ma'lumotlar yaxlitligini (pul summasi to'g'ri bo'lishini) ta'minlaydi.

Deadlock (O'lik Holat) Yechimi: Ba'zida ikki jarayon shunday qulflashadiki, ular bir-birini abadiy kutib qoladi (masalan, Jarayon A B ni, Jarayon B esa A ni kutadi). Bu Deadlock deyiladi. InnoDB aqlli soqchi vazifasini bajarib, bunday "o'lik holat"ni darhol aniqlaydi va ikkita tranzaksiyadan birini (odatda, eng kam ish qilganini) avtomatik bekor qiladi. Shu tariqa, butun tizim bir kishi tufayli "qotib qolishi"ning oldi olinadi.

Algoritmlar va So'rovlarni Optimallashtirish.

Bu MySQL ning "miya" qismi bo'lib, eng muhim algoritmik jarayon.

So'rov Optimallashtiruvchisi (Query Optimizer)

Kosmisiyaga asoslangan optimallashtirish (Cost-Based Optimization - CBO): Ma'lumotlar bazasining statistik ma'lumotlariga (masalan, indeksdagi ma'lumotlar zichligi) asoslanib, so'rovni bajarishning eng arzon (tezkor) yo'lini hisoblab topuvchi algoritm.

CBO ning qanday qilib turli so'rov rejalarini (Execution Plans) yaratishi va baholashi.

Bog'lanish Algoritmlari (Join Algorithms)

Nested Loop Join (NLJ), Block Nested Loop Join (BNLJ) va Hash Join (Hash Join/BNLJ o'rniga): So'rovlar turlari va jadval hajmlariga qarab, optimallashtiruvchi qaysi bog'lanish algoritmini tanlashi. Ularning ishlash prinsipi.

Yuqorida biz MySQL ning ichki qismlarida ishlaydigan murakkab algoritmlar bilan tanishdik. Endi asosiy savolga javob beramiz: bu nazariy bilimlar bizga amalda qanday foyda beradi? Algoritmlarni tushunish ishlab chiquvchilar va administratorlarga shunchaki kod yozishdan ko'ra, tizimni muhandislik nuqtai nazaridan optimallashtirish imkonini beradi.

Indeks Strategiyasi:

Indeks yaratish ma'lumotlarni qidirishni tezlashtiruvchi "sehrli tayoqcha" bo'lishi mumkin. Lekin noto'g'ri qo'llanilgan indeks tizimni sekinlashtiruvchi "tormoz"ga aylanadi. Bu yerda B+ Tree algoritmi haqidagi bilim muhim rol o'ynaydi.

Noto'g'ri Indeksning Xavfi: Biz bilamizki, B+ Tree ma'lumotlarni tartiblash uchun ajoyib, ammo har bir yangi indeks yaratilganida, MySQL unga bog'liq bo'lgan yangi B+ Tree ni diskda yaratishi kerak. Agar siz indekslarni juda ko'p yaratib yuborsangiz (masalan, jadvaldagi har bir ustunga), bu nafaqat disk maydonini isrof qiladi, balki kiritish (INSERT), o'chirish (DELETE) va yangilash (UPDATE) operatsiyalarini keskin sekinlashtiradi. Chunki har qanday ma'lumot o'zgarganda, MySQL endi bitta emas, balki bir nechta B+ Tree tuzilmalarini yangilashga majbur bo'ladi.

Samarali Indeks Tanlash Tavsiyalari: Algoritmik xarajatlarni kamaytirish uchun, indekslar faqat eng zarur joylarda yaratilishi kerak:

Faqatgina qidiruv (WHERE) va bog'lash (JOIN) operatsiyalarida tez-tez ishlatiladigan ustunlarga indeks qo'ying.

Kamdan-kam o'zgaruvchi ustunlarga indeks yaratishni afzal ko'ring. Agar ustun doimiy yangilansa, B+ Tree ni saqlash xarajati, qidiruvdan olinadigan foydadan oshib ketishi mumkin.

Ma'lumotlar Yaxlitligi va Tranzaksiyalar: Qulflashni Boshqarish

Ma'lumotlar bazasida tranzaksiyalarning to'g'ri ishlashi (masalan, bank hisobidagi pulning adashmasligi) qulflash (Locking) algoritmlarining qanday ishlashiga bevosita bog'liq. Ishlab chiquvchi Tranzaksiya Izolyatsiya Darajalarini to'g'ri tanlab, bu algoritmlarni boshqarishi mumkin.

Izolyatsiya Darajalarining Ta'siri: MySQL da READ COMMITTED, REPEATABLE READ, SERIALIZABLE kabi izolyatsiya darajalari mavjud.

Agar siz yuqori darajani (masalan, SERIALIZABLE) tanlasangiz, qulflash algoritmlari juda ehtiyotkor ishlaydi va bir tranzaksiya tugaguniga qadar ma'lumotlarni uzoqroq muddatga qulflaydi. Bu ma'lumotlar yaxlitligini maksimal darajada himoyalaydi, ammo tizimning bir vaqtda ishlash qobiliyatini (Concurrency) pasaytirib, Deadlock ehtimolini oshiradi.

Agar past darajani (masalan, READ COMMITTED) tanlasangiz, qulflash kamroq amalga oshiriladi, bu tizimning tezligini oshiradi, ammo "Dirty Reads" kabi kichik muammolar paydo bo'lishi xavfi tug'ilishi mumkin.

Optimal Tanlov: Algoritmlar haqidagi bilim, ishlab chiquvchiga o'z loyihasining ehtiyojlariga mos keladigan xavfsizlik va tezlik o'rtasidagi oltin muvozanatni topishga yordam beradi. Aksariyat hollarda MySQL ning standart darajasi (REPEATABLE READ) yaxshi ishlaydi, ammo yuqori tezlik talab qilinadigan tizimlarda bu darajani tushirish yoki ko'tarish kerak bo'ladi.

Xulosa. Bizning tahlilimiz shuni ko‘rsatdiki, MySQL kabi qudratli ma’lumotlar bazasi shunchaki ma’lumot saqlanadigan joy emas. U — bu murakkab mexanizm bo‘lib, uning samarali ishlashi uning ichidagi algoritmlar deb ataluvchi yashirin "qoidalar" to‘plamiga bog‘liq.

Oddiy qilib aytganda, tizimingizning tezligi uchta asosiy "aqlli harakat" bilan ta'minlanadi:

1. Aqlli Tartiblash: B+ Tree algoritmi tufayli ma'lumotlarimiz doimo shunday tartiblanganki, biz ularni butun diskni qidirib o'tirmay, chindan ham bir zumda topa olamiz.
2. Aqlli So'rov Rejasi: So'rov Optimallashtiruvchisi biz yozgan SQL so'rovni qabul qilib, uni bajarishning barcha mumkin bo'lgan variantlari ichidan eng tez va eng tejamkor yo'lini aql bilan tanlab oladi.
3. Aqlli Himoya: Qulflash mexanizmlari tufayli esa, o'nlab odam bir vaqtning o'zida ma'lumotni o'zgartirayotgan bo'lsa ham, hisob raqamingizdagi summa yoki inventardagi mahsulot soni adashmasdan to'g'ri qoladi.

Bu algoritmlarni tushunish shunchaki nazariy qiziqish emas. Bu — amaliy ustunlikdir. Endi biz indekslarni yaratayotganimizda, shunchaki "yaxshiroq ishlaydi" deb emas, balki "Bu B+ Tree'ning tezkor qidiruv tamoyiliga mos keladi va UPDATE xarajatini kamaytiradi" deb aytishimiz mumkin. Maqola sizga aynan shu bilimlarni qo'lga kiritishga yordam berdi: MySQL'ni shunchaki ishlatish emas, balki uni ichidan boshqarish. Kelajak esa yanada hayajonli. Yangi Hash Join kabi algoritmlar gigabaytlab ma'lumotlarni bog'lashni yanada tezlashtirmoqda. Eng muhimi, optimallashtiruvchilar o'z xatolaridan o'rganish va yuklamaga moslashish uchun sun'iy intellekt (AI) elementlarini qabul qilishga tayyorlanmoqda. Bu esa MySQL'ning tezligini yana bir yangi bosqichga olib chiqadi. Xulosa qilib aytganda, ma'lumotlar bazasidagi algoritmlar — bu tizimning yashirin muhandislik mo'jizalaridir. Ushbu mo'jizalarni tushunish, raqamli davrda muvaffaqiyatga erishmoqchi bo'lgan har bir texnolog mutaxassis uchun shaxsiy super-kuch hisoblanadi.

Foydalanilgan Adabiyotlar Ro'yxati.

1. **Date, C. J.** (2004). An Introduction to Database Systems (8th ed.). Addison-Wesley.
2. **Graves, R., Krol, Z., & Zamir, P.** (2018). High Performance MySQL: Practical Guide to High-Performance Web Applications (4th ed.). O'Reilly Media.
3. **Silberschatz, A., Korth, H. F., & Sudarshan, S.** (2020). Database System Concepts (7th ed.). McGraw Hill.
4. **Bayer, R., & McCreight, E.** (1972). Organization and Maintenance of Large Ordered Indexes. Acta Informatica, 1(3), 173–189.
5. **Chaudhuri, S., & Narasayya, V. R.** (2007). An Efficient, Cost-Based Query Optimizer. VLDB Journal, 16(3), 299-322.
6. **Oracle Corporation.** (So'nggi versiya). MySQL Documentation: The InnoDB Storage Engine. [Rasmiy Hujjat]. Olingan manba: