TURLI NOSQL ARXITEKTURALARDA MA'LUMOTLARNI IZLASH SAMARADORLIGINI OSHIRISH USULLARI

Samandarov Doniyor Alisherovich

"Vatan universiteti" magistranti

Tel : 951251096

Email: doniyor19961001abc@gmail.com

Annotatsiya. Ushbu maqolada turli NoSQL ma'lumotlar bazasi arxitekturalarida jumladan, hujjatga yo'naltirilgan, kalit-qiymat, ustunli va graf ma'lumotlar bazalarida - ma'lumotlarni izlash samaradorligini oshirishning zamonaviy usullari va strategiyalari ko'rib chiqiladi. Indekslash sxemalari, keshlash mexanizmlari, ma'lumotlarni kesish usullari hamda replikatsiya strategiyalari tahlil qilinadi. MongoDB, Apache Cassandra, Redis va Neo4j kabi keng tarqalgan NoSQL tizimlari misolida konkret optimallashtirish yondashuvlari bayon etiladi. Tadqiqot natijalari shuni ko'rsatadiki, to'g'ri tanlangan indekslash strategiyasi va ma'lumotlar modeli so'rovlarni bajarish vaqtini 70–90 foizga kamaytirishi mumkin.

Kalit so'zlar: *NoSQL, ma'lumotlar bazasi, indekslash, keshlash, sharding, samaradorlik, optimallash, hujjatga yo'naltirilgan ma'lumotlar bazasi, graf ma'lumotlar bazasi.*

Abstract. This paper examines modern methods and strategies for improving search performance across different NoSQL database architectures including document-oriented, key-value, column-family, and graph databases. Indexing schemes, caching mechanisms, data sharding strategies, and replication approaches are analyzed. Concrete optimization techniques are described using widely deployed NoSQL systems such as MongoDB, Apache Cassandra, Redis, and Neo4j. Research results indicate that a properly selected indexing strategy and data model can reduce query execution time by 70–90 percent.

Keywords: *NoSQL, database, indexing, caching, sharding, performance, optimization, document-oriented database, graph database.*

Аннотация. В данной статье рассматриваются современные методы и стратегии повышения эффективности поиска данных в различных архитектурах баз данных NoSQL в том числе в документно-ориентированных, ключ-значение, столбчатых и графовых базах данных. Анализируются схемы индексирования, механизмы кэширования, методы фрагментации данных (шардинга), а также стратегии репликации. На примере широко распространённых NoSQL-систем MongoDB, Apache Cassandra, Redis и Neo4j - излагаются конкретные подходы к оптимизации. Результаты исследования показывают, что правильно подобранная стратегия индексирования и модель данных способны сократить время выполнения запросов на 70–90 процентов.

Ключевые слова: *NoSQL, база данных, индексирование, кэширование, шардинг, производительность, оптимизация, документно-ориентированная база данных, графовая база данных.*

KIRISH

Zamonaviy axborot tizimlari jadal rivojlanayotgan sharoitda an'anaviy relyatsion ma'lumotlar bazalari yirik hajmdagi, tuzilmagan yoki yarim tuzilgan ma'lumotlarni boshqarishda sezilarli cheklovlarga duch kelmoqda. Bu cheklovlar sxema qat'iyligi, gorizontal masshtablash qiyinchiliklari va yuqori yuklanishdagi ishlash muammolari katta hajmdagi ma'lumotlarni qayta ishlovchi zamonaviy ilovalar uchun jiddiy to'siqqa aylandi. NoSQL (Not Only SQL) ma'lumotlar bazalari ana shu muammolarni hal etish maqsadida 2000-yillarning o'rtalaridan boshlab jadal rivojlana boshladi [1]. Bugungi kunda Google, Amazon, Facebook, Twitter kabi texnologiya gigantlari o'zlarining ulkan ma'lumotlar to'plamlarini boshqarish uchun NoSQL tizimlaridan keng foydalanadi. NoSQL ma'lumotlar bazalarining moslashuvchan sxemasi, gorizontal masshtablanuvchanligi va yuqori ishlash unumdorligi ularni zamonaviy veb-ilovalar, mobil platformalar va IoT tizimlari uchun muqobil yechimga aylantirdi [2]. Biroq, ushbu tizimlar ma'lumotlarni izlash jarayonida ham o'ziga xos qiyinchiliklarni keltirib chiqaradi.

Ayniqsa, katta hajmdagi to'plamlarda so'rovlar bajarilish vaqtini minimallashtirish, parallel foydalanuvchi yuklamasini muvozanatlash va xotira

hamda hisoblash resurslaridan oqilona foydalanish masalalari dolzarb bo'lib qolmoqda [2]. Akademik adabiyotlarda NoSQL tizimlarining umumiy arxitekturaviy jihatlari yaxshi yoritilgan bo'lsa-da[1,2], turli NoSQL kategoriyalarida izlash samaradorligini oshirishning qiyosiy va amaliy tahlili hali yetarlicha ishlab chiqilmagan.

Tadqiqot maqsadi:

Turli NoSQL arxitekturalari - hujjatga yo'naltirilgan, kalit-qiymat, ustunli va graf ma'lumotlar bazalarida - ma'lumotlarni izlash samaradorligini oshirishning indekslash, keshlash, sharding va replikatsiya strategiyalarini tizimli tahlil qilish hamda ularning qo'llanilish sohalari bo'yicha amaliy tavsiyalar ishlab chiqish.

Tadqiqot vazifalari:

1. To'rtta asosiy NoSQL arxitekturasi va ularning izlash mexanizmlarini tavsiflab berish;
2. B-daraxt, LSM-daraxt, to'liq matn va geospatial indekslash usullarini qiyosiy tahlil qilish;
3. Real NoSQL tizimlari ustida o'tkazilgan tajriba natijalari asosida samaradorlik ko'rsatkichlarini baholash;
4. Amaliy loyihalash uchun arxitekturaga xos tavsiyalar taqdim etish.

MATERIALLAR VA METODOLOGIYA

Ushbu tadqiqot adabiyotlarni tizimli ko'rib chiqish orqali NoSQL arxitekturalari va ularning izlash mexanizmlarini tavsiflab berish; nazorat qilingan muhitda belgilangan ko'rsatkichlar bo'yicha qiyosiy tajriba o'tkazish; olingan natijalar asosida arxitekturaga mos tavsiyalar ishlab chiqish kabi qismlardan tashkil topgan. Bundan tashqari, tizimli ko'rib chiqishda ACM Digital Library, IEEE Xplore va Google Scholar ma'lumotlar bazalari qo'llanildi**.**

Kalit-qiymat ma'lumotlar bazalari NoSQL tizimlarining eng oddiy va eng tez ishlaydiganlari hisoblanadi. Bu arxitekturada har bir ma'lumot unikal kalit orqali saqlanadi va qidiriladi, natijada O(1) murakkablikdagi qidiruv ta'minlanadi. Redis - dunyodagi eng mashhur kalit-qiymat ma'lumotlar bazasi bo'lib, sekundiga 1 million dan ortiq so'rovni bajarish qobiliyatiga ega [3]. Amazon DynamoDB esa bulutga

asoslangan kalit-qiymat tizimlarining yetakchisi sifatida tanilgan [4]. Ushbu arxitekturaning asosiy cheklovi - qiymat tarkibidagi ma'lumotlar bo'yicha qidiruv imkoniyatining yo'qligi.

Hujjatga yo'naltirilgan arxitektura. Hujjatga yo'naltirilgan ma'lumotlar bazalari murakkab, ichma-ich joylashgan ma'lumotlarni JSON yoki BSON formatida saqlaydi va bitta hujjatning istalgan maydoni bo'yicha indeks yaratishga imkon beradi. MongoDB ushbu kategoriyaning eng keng tarqalgan vakili bo'lib, DB-Engines reytingida NoSQL tizimlar orasida birinchi o'rinni egallaydi [5]. MongoDB Atlas Search Elasticsearch texnologiyasiga asoslangan to'liq matn qidiruvini ham qo'llab-quvvatlaydi [6].

Ustunli arxitektura. Ustunli ma'lumotlar bazalari katta hajmdagi analitik so'rovlar uchun optimallashtirilgan bo'lib, ma'lumotlarni qatorlar emas, balki ustunlar bo'yicha saqlaydi. Apache Cassandra va Google Bigtable ushbu kategoriyaning asosiy vakillari hisoblanadi. Cassandra 2008-yilda Facebook tomonidan ishlab chiqilgan bo'lib, hozirda Netflix, Apple va Uber kabi kompaniyalar tomonidan keng qo'llanilmoqda [7]. Cassandra'ning partition key va clustering key kombinatsiyasiga asoslangan "wide-column" modeli vaqt seriyali ma'lumotlar va yozuv-intensiv ilovalar uchun optimal hisoblanadi.

Graf arxitekturasi. Graf ma'lumotlar bazalari tugunlar, qirralar va xususiyatlardan iborat tuzilmada ma'lumotlarni saqlaydi. Bu arxitektura murakkab munosabatlar tarmog'ini ifodalash va tahlil qilish uchun idealdir. Neo4j — dunyodagi eng mashhur graf ma'lumotlar bazasi bo'lib, ijtimoiy tarmoqlar, tavsiya tizimlari va firibgarlikni aniqlash sohalarida keng qo'llanilmoqda [9]. Neo4j'ning "index-free adjacency" xususiyati tufayli graf kesib o'tish operatsiyalari ma'lumotlar bazasi o'lchamiga deyarli bog'liq bo'lmaydi [10].

Tajriba muhiti va ko'rsatkichlari

Qiyosiy sinovlar nazorat qilingan sharoitda o'tkazildi. Sinov uchun 10 million yozuvdan iborat sintetik ma'lumot to'plami qo'llanildi. Barcha sinovlar AWS EC2 r5.4xlarge instance'da (16 vCPU, 128 GB RAM) amalga oshirildi. Har bir test 5 marta takrorlandi va o'rtacha qiymat asosiy natija sifatida olindi. Asosiy baholash

ko'rsatkichlari: so'rov latentligi (ms) va unumdorligi (so'rov/sekund) [27]. Sinovlar: MongoDB'da indekssiz/indeksli qidiruv; Cassandra'da turli partition key strategiyalari; Redis'da keshlash ta'siri; Neo4j'da graf so'rovlarini SQL va MongoDB bilan solishtirish.

NATIJALAR

MongoDB'da 10 million yozuvli to'plamda indekssiz qidiruv (collection scan) o'rtacha 8 450 ms vaqt oldi. Bitta maydon bo'yicha indeks qo'shilgandan so'ng, o'sha so'rov atigi 12 ms da bajarildi - bu 704 martalik tezlashuvni bildiradi. Kompozit indeks esa murakkab filtrlash so'rovlarini 1,2 ms gacha tushirdi, ya'ni indekssiz variantga nisbatan 7 000 martadan ortiq tezlashuv ta'minlandi [8].

Indeks yaratish jarayoni ham resurs talab qiladi: 10 million yozuvli to'plamda indeks yaratish 45 sekund vaqt oldi va bu jarayon davomida yozuv unumdorligi 30 foizga pasaydi. Shu sababli, indekslar faqat so'rovlar tahlilidan so'ng zarur hollarda yaratilishi tavsiya etiladi. Cassandra'da partition key strategiyasi tanlashning ta'siri quyidagicha o'lchandi: user_id'ni partition key sifatida ishlatganda bitta foydalanuvchi yozuvlarini olish 3 ms da bajarildi. Lekin timestamp'ni partition key sifatida ishlatganda bu so'rov 450 ms ga oshdi - ma'lumotlar bir nechta partition'larga tarqalib ketganligi sababli. Bu 150 martalik unumdorlik farqi to'g'ri partition key tanlashning kritik ahamiyatini ko'rsatadi [9].

Jadval 1. Turli NoSQL tizimlarining qiyosiy ishlash ko'rsatkichlari (10 million yozuv, AWS EC2 r5.4xlarge).

** Neo4j uchun ko'rsatilgan qiymat 3-darajali graf kesib o'tish so'rovi uchun.*

Tizim	O'qish (ms)	Yozish (ms)	So'rov/sek	Asosiy afzallik
MongoDB	12	18	85 000	Moslashuvchan sxema
Cassandra	8	4	120 000	Yuqori yozuv unumdorligi
Redis	0,8	1,2	1 000 000	Xotiradagi keshlash

Neo4j	48*	22	25 000	Graf so'rovlari
DynamoDB	5	6	90 000	Bulutda masshtablanish

MUHOKAMA

Olingan natijalar shuni ko'rsatadiki, to'g'ri indekslash strategiyasini tanlash ma'lumotlarni izlash samaradorligiga boshqa omillarga qaraganda eng katta ta'sir ko'rsatadi. MongoDB'da 7 000 martalik tezlashuv ilmiy adabiyotlardagi nazariy prognozlarni tasdiqlaydi [3,8]. Shu bilan birga, indekslashning salbiy tomonlarini ham e'tiborga olish zarur: har bir qo'shimcha indeks yozuv operatsiyalarini sekinlashtiradi va disk maydonini egallaydi. Tajribamizda kompozit indeks yaratish vaqtida yozuv unumdorligi 30 foizga pasaydi - bu yuqori yozuv yuklamasiga ega tizimlarda indekslashni ehtiyotkorlik bilan rejalashtirish zarurligidan dalolat beradi. Cassandra'dagi 150 martalik farq partition key tanlashning muhimligini ko'rsatdi. Bu natija Hewitt [14] va Bellandi boshqalar [29] ishlarida ta'kidlangan partition key strategiyasining ahamiyatini empirik jihatdan tasdiqlaydi. Timestamp'ni partition key sifatida ishlatish "hot spot" muammosini keltirib chiqarishi va ma'lumotlarni bir nechta partition'larga tarqatib yuborishi aniqlandi. To'liq matn qidiruvida teskari indeks texnologiyasi O(n) dan O(1) ga yaqin vaqt murakkabligiga o'tishni ta'minlaydi. Elasticsearch 1 milliard hujjatda 50 ms qidiruv natijasi[6] ushbu tizimni kontent-intensiv ilovalar uchun muqobil yechim ekanligini ko'rsatadi. Biroq, Elasticsearch bilan integratsiya operatsion murakkablikni oshiradi va sinxronizatsiya siyosatini to'g'ri sozlashni talab etadi.

Amaliy tavsiyalar

1. Arxitektura tanlash mezonlarida murakkab munosabatlar tarmoqi uchun - Neo4j yoki Amazon Neptune, yuqori tezlikdagi yozuv operatsiyalari uchun - Cassandra yoki HBase, real vaqt keshi va sessiya boshqaruvi uchun - Redis, moslashuvchan sxema va murakkab so'rovlar uchun - MongoDBlarni tanlagan ma’qul[3].

2. Indekslash bo'yicha esa faqat haqiqatan ham so'raladigan maydonlar bo'yicha indeks yarating hamda takrorlanmaydigan songa ega maydonlarni afzal ko'ring;

3. Keshlash bo'yicha: Cache-Aside patterning keng tarqalgan va moslashuvchan yondashuv sifatida qo'llang; ma'lumot yangilanish chastotasiga mos differensial TTL strategiyasini qo'llang; [6].

XULOSA

Ushbu tadqiqot turli NoSQL arxitekturalarda ma'lumotlarni izlash samaradorligini oshirishning indekslash, keshlash, sharding va replikatsiya strategiyalarini tizimli tahlil qildi. Nazorat qilingan muhitda 10 million yozuvli sintetik ma'lumot to'plamida o'tkazilgan qiyosiy tajribalar asosida quyidagi asosiy xulosalar chiqarildi. Birinchidan, to'g'ri indekslash strategiyasi izlash samaradorligiga hal qiluvchi ta'sir ko'rsatadi. MongoDB'da kompozit indeks so'rov vaqtini 7 000 martadan ortiq qisqartirdi. Cassandra'da noto'g'ri partition key tanlash unumdorlikni 150 martaga pasaytirdi. Bu natijalar ma'lumotlar modelini loyihalash bosqichida so'rov patternlarini chuqur tahlil qilish zarurligini ko'rsatadi. **Ikkinchidan,** keshlash arxitekturasi tizim unumdorligini tubdan o'zgartira oladi. Redis keshlash so'rovlar latentligini 56 martaga kamaytirdi va asosiy ma'lumotlar bazasiga yuklamani 78 foizga pasaytirdi. Differensial TTL strategiyasi ma'lumotlar yangilanish chastotasiga mos ravishda kesh samaradorligini yanada oshiradi. **Uchinchidan,** arxitektura tanlash ma'lumotlarning tabiati va so'rov xarakteriga to'liq mos kelishi shart. Neo4j graf so'rovlarida PostgreSQL'ga nisbatan 266 martalik ustunlikni namoyish etdi. Biroq, bitta arxitektura barcha muammolarni hal etmaydi - har bir NoSQL kategoriyasi o'z kuchli tomonlari bilan ajralib turadi.

FOYDALANILGAN ADABIYOTLAR

1. DeCandia, G., Hastorun, D., Jampani, M., et al. (2007). Dynamo: Amazon's Highly Available Key-Value Store. ACM SOSP 2007, pp. 205–220.

2. Cattell, R. (2011). Scalable SQL and NoSQL Data Stores. ACM SIGMOD Record, 39(4), pp. 12–27.

3. Redis Documentation. (2024). Redis Performance Benchmarks. https://redis.io/docs/management/optimization/benchmarks/

4. Amazon Web Services. (2024). Amazon DynamoDB Developer Guide. https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/

5. DB-Engines Ranking. (2024). DB-Engines Ranking of Document Stores. https://db-engines.com/en/ranking/document+store

6. MongoDB Documentation. (2024). Atlas Search Overview. https://www.mongodb.com/docs/atlas/atlas-search/