MONGODB ДЕРЕКТЕР ҚОРЫНА КІРІСПЕ

1. Кіріспе

MongoDB – құжатқа бағытталған (document-oriented) NoSQL дерекқоры, құрылымдалмаған және жартылай құрылымдалған деректермен жұмыс істеуге бейімделген. Үлкен деректер (Big Data) концепциясы үлкен көлемді, жылдам өңдеуді талап ететін және әртүрлі құрылымдағы мәліметтерді қамтиды.

Бұл лекцияда MongoDB-нің негізгі қағидалары, оның үлкен деректермен жұмыс істеу мүмкіндіктері және оның реляциялық дерекқорлардан айырмашылығы қарастырылады.

2. MongoDB-нің негізгі ерекшеліктері

MongoDB – деректерді **JSON (JavaScript Object Notation) тәрізді құжаттар** түрінде сақтайтын дерекқор. Оның басты ерекшеліктері:

1. **Гибридті құрылым:** Деректер реляциялық дерекқорлардағыдай қатаң кестелермен шектелмейді.
2. **Жоғары масштабталу:** Көлденең масштабтау (sharding) арқылы үлкен деректерді тиімді сақтау және өңдеу.
3. **Икемділік:** Динамикалық схемасы (schema-less), яғни бағандар алдын ала анықталмайды.
4. **Жылдамдық:** JSON-пішінді құжаттарды индекстеу арқылы сұраныстар жылдам орындалады.

JavaScript Object Notation (JSON) деген не?

JavaScript Object Notation (JSON) – мәліметтерді құрылымдалған түрде сақтауға және алмасуға арналған жеңіл әрі ыңғайлы формат. JavaScript тілінің объектілеріне негізделген, бірақ қазіргі таңда көптеген бағдарламалау тілдерінде кеңінен қолданылады.

MongoDB мен MS SQL салыстыру

Біз MS SQL-мен жұмыс істеуді білсеміз, сіз үшін MongoDB бастапқыда біраз түсініксіз болуы мүмкін. Себебі, ол дәстүрлі кестелермен (TABLE), бірегей кілттермен (PRIMARY KEY) және JOIN операторларымен жұмыс істемейді.

JSON құрылымы

JSON мәліметтерді **"кілт – мән" (key-value pair)** түрінде сақтайды.

1. **Қарапайым JSON объектісі**

```
{
  "name": "Алихан",
  "age": 25,
  "city": "Алматы"
}
```

Бұл жерде:

"name" – **кілт (key)**, "Алихан" – **мән (value)**

"age" – **кілт**, 25 – **мән**

"city" – **кілт**, "Алматы" – **мән**

MongoDB-нің үлкен деректермен (Big Data) жұмыс істеуі

MongoDB үлкен деректерді өңдеу және сақтау үшін кеңінен қолданылады. Оның үлкен деректермен жұмыс істеуге арналған негізгі технологиялары:

- Sharding (кесінділеу): Үлкен деректерді бірнеше серверге бөлшектеп сақтау.
- Replication (көшіру): Деректердің сақталу қауіпсіздігін арттыру үшін бірнеше реплика құру.
- Aggregation Framework: Үлкен деректермен күрделі аналитикалық операцияларды орындау үшін тиімді құрал.
- MapReduce: Үлкен деректерді параллельді түрде өңдеу.

MONGODB МЕН SQL SERVER: НЕГІЗГІ АЙЫРМАШЫЛЫҚТАР

Мәліметтер құрылымы мен икемділігі

SQL Server: Қатаң Схема

SQL Server-де деректер схемасы (кестелер, бағандар, деректер типтері) дерекқор жасалғанға дейін толық анықталуы қажет.

Бұл қатаңдық деректердің сапасы мен біркелкілігін қамтамасыз етеді, бірақ болашақтағы өзгерістерді енгізуді қиындатады.

MongoDB: Динамикалық Схема

MongoDB динамикалық схеманы қолданады, яғни құжаттар әртүрлі өрістер мен құрылымдарға ие болуы мүмкін.

Бұл икемділік әзірлеушілерге деректер моделін тез өзгертуге және жаңа функцияларды оңай қосуға мүмкіндік береді, бұл әсіресе стартаптар мен инновациялық жобалар үшін пайдалы.

Транзакциялар және деректер тұтастығы

SQL Server: Толық ACID сәйкестігі

- SQL Server-де ACID (Atomicity, Consistency, Isolation, Durability) қасиеттеріне толық сәйкес келетін транзакциялар бар.
- Бұл қаржылық операциялар, есеп жүйелері сияқты деректердің абсолютті тұтастығы мен сенімділігі талап етілетін салаларда өте маңызды.
- Кез келген операция толық орындалады немесе мүлдем орындалмайды, бұл деректердің қате болуын болдырмайды.

MongoDB: Көп құжатты транзакциялар

- MongoDB 4.0 нұсқасынан бастап көп құжатты ACID транзакцияларын қолдауды енгізді, бұл оның деректер тұтастығын айтарлықтай арттырды.
- Дегенмен, реляциялық дерекқорларға қарағанда кейбір шектеулері болуы мүмкін, әсіресе таратылған ортада.
- Бұл MongoDB-ді көптеген қолданбалар үшін жеткілікті сенімді етеді, бірақ аса қатаң талаптарға сай болмауы мүмкін.

3. Реляциялық дерекқорлармен (SQL) салыстыру

Сипаттама	MongoDB (NoSQL)	Реляциялық дерекқорлар (SQL)
Деректерді сақтау	JSON-құжаттар (BSON форматында)	Кестелер (таблицалар)
Деректер құрылымы	Коллекциялар (Collections) және құжаттар (Documents)	Кестелер (Rows & Columns)
Схема	Гибридті, динамикалық	Қатаң, алдын ала анықталған
Шартсыз байланыстар	Байланыстарды сақтау үшін құжат ішіндегі массивтер қолданылады	Көптеген кестелермен байланыс (JOIN)
Шкала	Көлденең масштабтау (Sharding)	Тік масштабтау (Vertical scaling)
Сұраныстар	NoSQL сұраныстары (find, aggregate)	SQL сұраныстары (SELECT, JOIN)

MongoDB және SQL командаларының ұқсастығы

Операция	MS SQL	MongoDB
Жаңа дерекқор құру	CREATE DATABASE testDB;	use testDB
Кесте немесе коллекция құру	CREATE TABLE users (id INT, name VARCHAR(50));	db.createCollection("users")
Жаңа жазба енгізу	INSERT INTO users (id, name) VALUES (1, 'Али');	db.users.insertOne ({id: 1, name: "Али"})
Деректерді шығару	SELECT * FROM users;	db.users.find()
Белгілі бір шартпен іздеу	SELECT * FROM users WHERE id = 1;	db.users.find ({id: 1})
Жазбаны жаңарту	UPDATE users SET name = 'Петр WHERE id = 1;	db.users.updateOne ({id: 1}, {$set: {name: "Петр"}})
Жазбаны жою	DELETE FROM users WHERE id = 1;	db.users.deleteOne({id: 1})

3. JOIN орнына ішкі құжаттар (Embedded Documents) қолданылады

SQL-да JOIN арқылы байланыс жасалады:

```
SELECT users.name, orders.product
FROM users
JOIN orders ON users.id = orders.user_id;
```

MongoDB-де байланыс келесідей сақталады:

```
{
"name": "Али",
"orders": [ {"product": "Ноутбук", "price": 1000},
{"product": "Мышь", "price": 50}  ]}
```

Бұл әдіс **JOIN операциясын азайтып, деректерді жылдам алуға мүмкіндік береді**.

Қорытындылай келе, MongoDB мен SQL Server деректерді сақтау мен өңдеуге арналғанымен, тәсілдері әртүрлі. SQL Server реляциялық модельге сүйеніп, қатаң схема мен ACID транзакцияларын қамтамасыз етеді, сондықтан есеп-қисап және корпоративтік жүйелерге қолайлы. Ал MongoDB құжаттық модель арқылы икемді құрылым ұсынып, тез өзгеретін талаптары бар және үлкен көлемді деректермен жұмыс істейтін веб/микросервис жобаларында тиімді. Таңдау жоба қажеттілігіне байланысты: тұрақтылық пен қатаң құрылым керек болса – SQL Server, ал икемділік пен масштабтау маңызды болса – MongoDB.

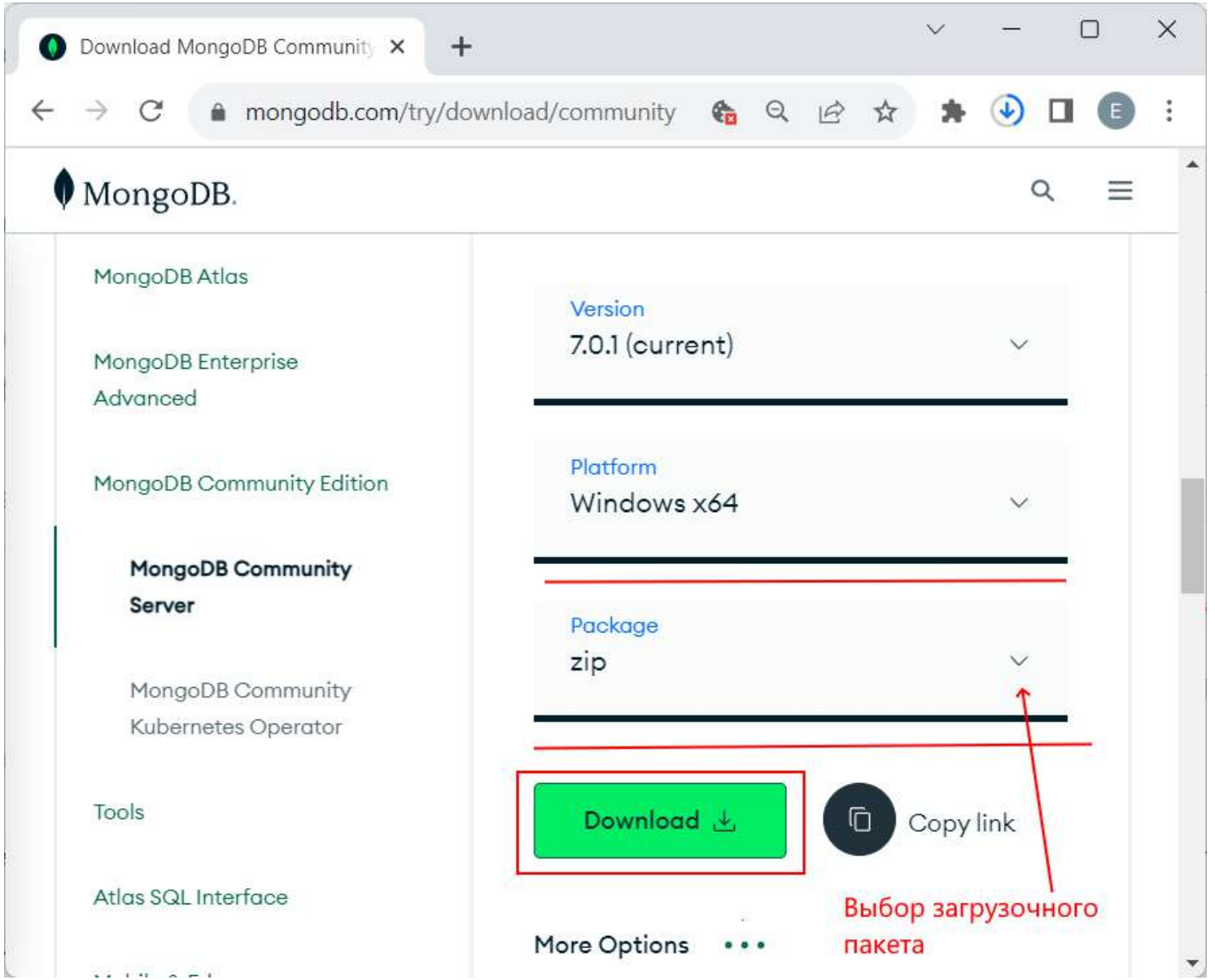
Қорытындылай келе, MongoDB мен SQL Server деректерді сақтау мен өңдеуге арналғанымен, тәсілдері әртүрлі. SQL Server реляциялық модельге сүйеніп, қатаң схема мен ACID транзакцияларын қамтамасыз етеді, сондықтан есеп-қисап және корпоративтік жүйелерге қолайлы. Ал MongoDB құжаттық модель арқылы икемді құрылым ұсынып, тез өзгеретін талаптары бар және үлкен көлемді деректермен жұмыс істейтін веб/микросервис жобаларында тиімді. Таңдау жоба қажеттілігіне байланысты: тұрақтылық пен қатаң құрылым керек болса – SQL Server, ал икемділік пен масштабтау маңызды болса – MongoDB.

MONGODB СЕРВЕРІН ОРНАТУ

MONGOD СЕРВЕРІН ОРНАТУ

- Ресми сайт әртүрлі платформаларға арналған дистрибутив пакеттерін ұсынады: Windows, Linux, macOS, Solaris. Әр платформа үшін бірнеше дистрибутив қолжетімді. Сонымен қатар сервердің екі нұсқасы бар: тегін **Community** және ақылы **Enterprise**. Бұл нұсқаулықта біз тегін **Community** нұсқасын қолданамыз.
- MongoDB орнату үшін ресми сайттан таралатын пакеттердің бірін жүктейміз.

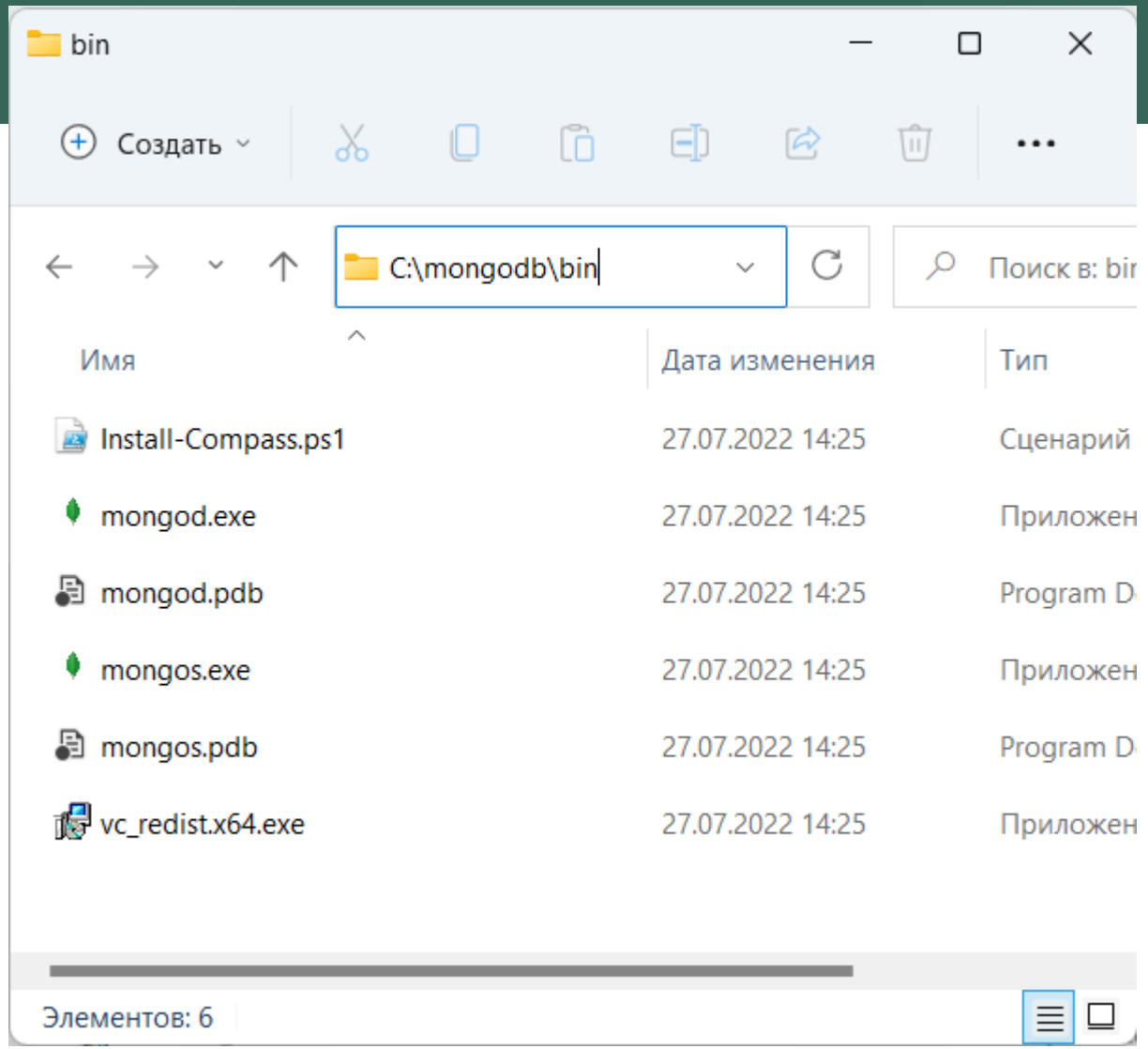
- Қажетті файлдарды жүктеу үшін операциялық жүйені және сәйкес пакет түрін таңдаймыз. Орнатуды Windows операциялық жүйесі мысалында қарастырайық.
- Windows үшін MongoDB-ні бірнеше нұсқада жүктеуге болады: **msi орнатқышы** немесе **zip архиві**. Іс жүзінде zip-архивті жүктеп, оны қажетті қалтаға ашу жеткілікті. Сондықтан осы нұсқаны таңдаймыз (дегенмен msi орнатқышын да қолдануға болады).



https://metanit.com/nosql/mongodb/1.2.php

MongoDB орнату

Егер орнатудан бұрын MongoDB-нің ескі нұсқасы орнатылған болса, оны алдымен жою қажет.
Архивті жүктегеннен кейін оны C:\mongodb қалтасына ашамыз.
Орнатылған архивтегі bin қалтасын (C:\mongodb\bin) ашсақ, әртүрлі қызмет атқаратын көптеген қосымшаларды көреміз.



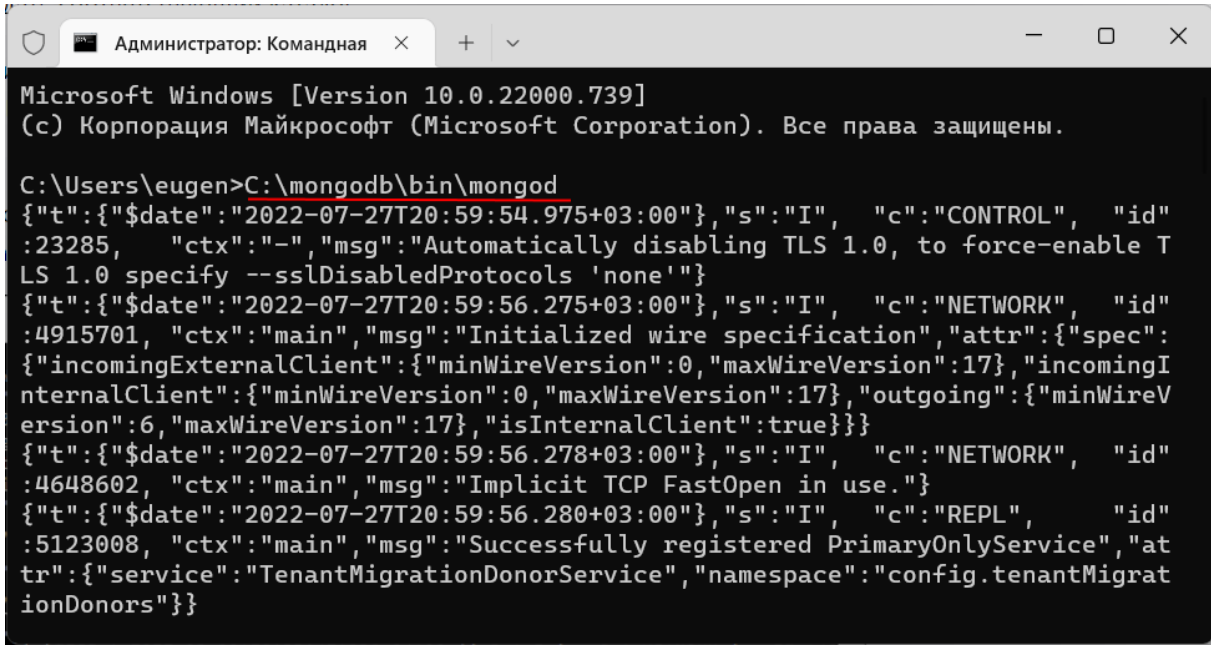
https://metanit.com/nosql/mongodb/1.2.php

MONGODB КАТАЛОГЫ

- **mongod** – MongoDB деректер қорының сервері. Ол сұраныстарды өңдейді, деректер форматын басқарады және деректер қорын басқаруға қатысты фондық операцияларды орындайды.
- **mongos** – MongoDB маршрутизация қызметі. Ол сұраныстарды өңдеуге және кластердегі деректердің орналасуын анықтауға көмектеседі.

ДҚ ҮШІН КАТАЛОГ ҚҰРУ ЖӘНЕ MONGODB ІСКЕ ҚОСУ

Орнатудан кейін қатты дискіде MongoDB деректер қорлары сақталатын каталог құру қажет.
Егер деректерді басқа жолда сақтау қажет болса, MongoDB іске қосылғанда --dbpath параметрі арқылы жолды көрсетуге болады.
ДҚ сақтау каталогы құрылғаннан кейін MongoDB серверін іске қосуға болады. Сервер – bin қалтасында орналасқан **mongod** қосымшасы. Ол үшін терминалды (командалық жолды) ашып, тиісті команданы енгіземіз. Windows-та бұл келесідей орындалады.

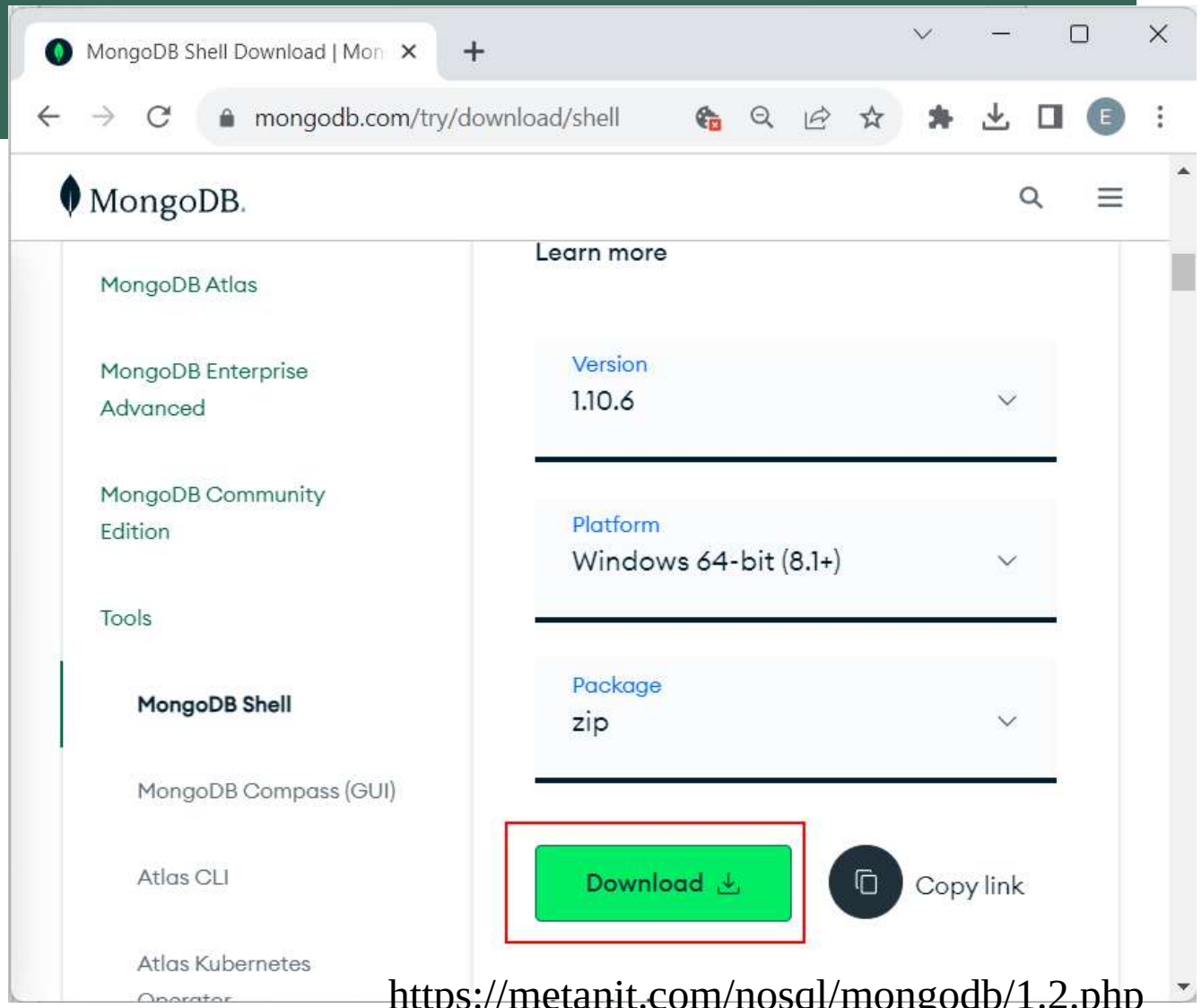


https://metanit.com/nosql/mongodb/1.2.php

Сервер іске қосылғанда командалық жолда қызметтік ақпарат көрсетіледі.
Сервер сәтті іске қосылғаннан кейін деректер қорымен клиент арқылы жұмыс істей аламыз.

MONGOSH КЛИЕНТІН ОРНАТУ

- Жоғарыда біз MongoDB серверін орнаттық. Бірақ сервермен жұмыс істеу үшін клиент қажет. Ең қарапайым клиент – **MongoDB Shell (mongosh)**, яғни серверге сұраныстар жіберуге арналған консольдік қабық. Ол да MongoDB компаниясы тарапынан ұсынылады.
- Клиентті орнату үшін келесі адреске өтеміз: https://www.mongodb.com/try/download/shell
- Мұнда да әртүрлі операциялық жүйелерге арналған нұсқаларды таңдауға болады. Оны жүктеп, C:\mongosh қалтасына ашамыз. Архивтегі bin қалтасына (C:\mongosh\bin) кірсек, MongoDB серверімен жұмыс істеуге арналған mongosh консольдік утилитасын көреміз.



https://metanit.com/nosql/mongodb/1.2.php

КЛИЕНТ АРҚЫЛЫ СЕРВЕРГЕ ҚОСЫЛУ

Енді орнатылған **mongosh** клиентін MongoDB серверімен өзара әрекеттесу үшін қолданамыз (mongosh-пен жұмыс істегенде сервер **mongod** іске қосулы болуы керек екенін ұмытпаңыз).

mongosh файлын іске қосамыз. Бағдарлама іске қосылғанда, ол MongoDB серверіне қосылу үшін қандай қосылу жолын қолданатынын сұрайды. Бұл жерде жай ғана **Enter** батырмасын басамыз – әдепкі қосылу жолы қолданылады.

Әдепкі бойынша MongoDB сервері 27017 портында жұмыс істейді, ал толық қосылу жолы мынадай:

mongodb://localhost:27017 немесе mongodb://127.0.0.1:27017

Қосылғаннан кейін консоль қызметтік ақпаратты көрсетіп, **test** деректер қорына қосылады.

Енді қарапайым әрекеттерді орындайық. Консольге келесі командаларды ретімен енгіземіз:

```
db.users.insertOne({ name: "Tom" })
db.users.find()
```

db.users.insertOne() функциясы арқылы **test** деректер қорының **users** коллекциясына *{ name: "Tom" }* объектісі қосылады.

db – ағымдағы деректер қорын білдіреді. Біз әдепкі бойынша **test** деректер қорына қосылғанбыз. Егер мұндай деректер қоры болмаса, ол автоматты түрде құрылады.

users – деректер қосылатын коллекция. SQL-дегідей кестелерді алдын ала құрудың қажеті жоқ, MongoDB коллекцияларды автоматты түрде жасайды.

Қосылатын объект JSON форматына ұқсас түрде сипатталады. Бұл мысалда "name" кілтіне "Tom" мәні берілген, яғни біз Tom атты пайдаланушыны қостық.

Объект сәтті қосылса, консоль операция нәтижесін, соның ішінде қосылған объектінің идентификаторын көрсетеді.

db.users.insertOne() функциясы арқылы **test** деректер қорының **users** коллекциясына *{ name: "Tom" }* объектісі қосылады.

db – ағымдағы деректер қорын білдіреді. Біз әдепкі бойынша **test** деректер қорына қосылғанбыз. Егер мұндай деректер қоры болмаса, ол автоматты түрде құрылады.

users – деректер қосылатын коллекция. SQL-дегідей кестелерді алдын ала құрудың қажеті жоқ, MongoDB коллекцияларды автоматты түрде жасайды.

Қосылатын объект JSON форматына ұқсас түрде сипатталады. Бұл мысалда "name" кілтіне "Tom" мәні берілген, яғни біз Tom атты пайдаланушыны қостық.

Объект сәтті қосылса, консоль операция нәтижесін, соның ішінде қосылған объектінің идентификаторын көрсетеді.

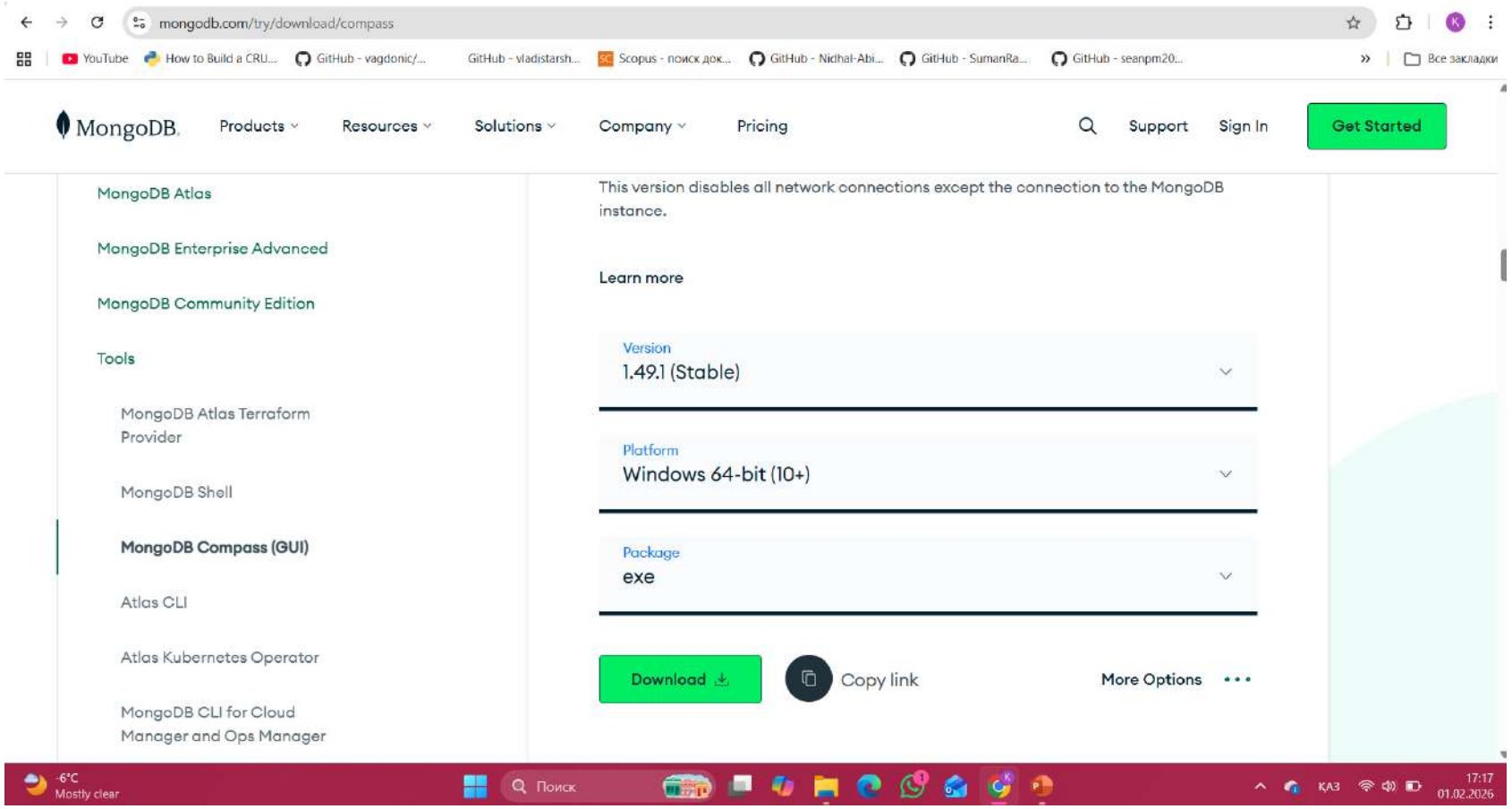
MONGODB COMPASS ГРАФИКАЛЫҚ КЛИЕНТІ

MongoDB Compass - MongoDB дерекқорымен жұмыс істеуге арналған ең қолайлы графикалық интерфейстердің бірі. MongoDB үшін ең тиімді, себебі ол деректермен жұмысты айтарлықтай жеңілдетеді. Құрал құжаттардың (documents) құрылымын бірден көрнекі түрде көрсетіп, деректерді түсінуді жылдамдатады.

Сонымен қатар, Compass-та **өнімділікті нақты уақытта бақылау** мүмкіндігі бар: сұраныстардың орындалуын, жүктемені және негізгі көрсеткіштерді қарап, жүйенің қалай жұмыс істеп тұрғанын тез бағалауға болады. Бұл – диагностика жасауға, сұраныстарды оңтайландыруға және индекстердің тиімділігін түсінуге көмектесетін маңызды функция.

Оны жүктеу үшін келесі адреске өтеміз: https://www.mongodb.com/try/download/compass

- Бұл бетте Compass нұсқасын және мақсатты операциялық жүйені таңдауға болады. Орнатуды Windows операциялық жүйесі мысалында қарастырайық.
- Windows үшін бір ғана **exe** файлы түріндегі дайын (скомпиляцияланған) қосымша қолжетімді. Оны жүктейміз.



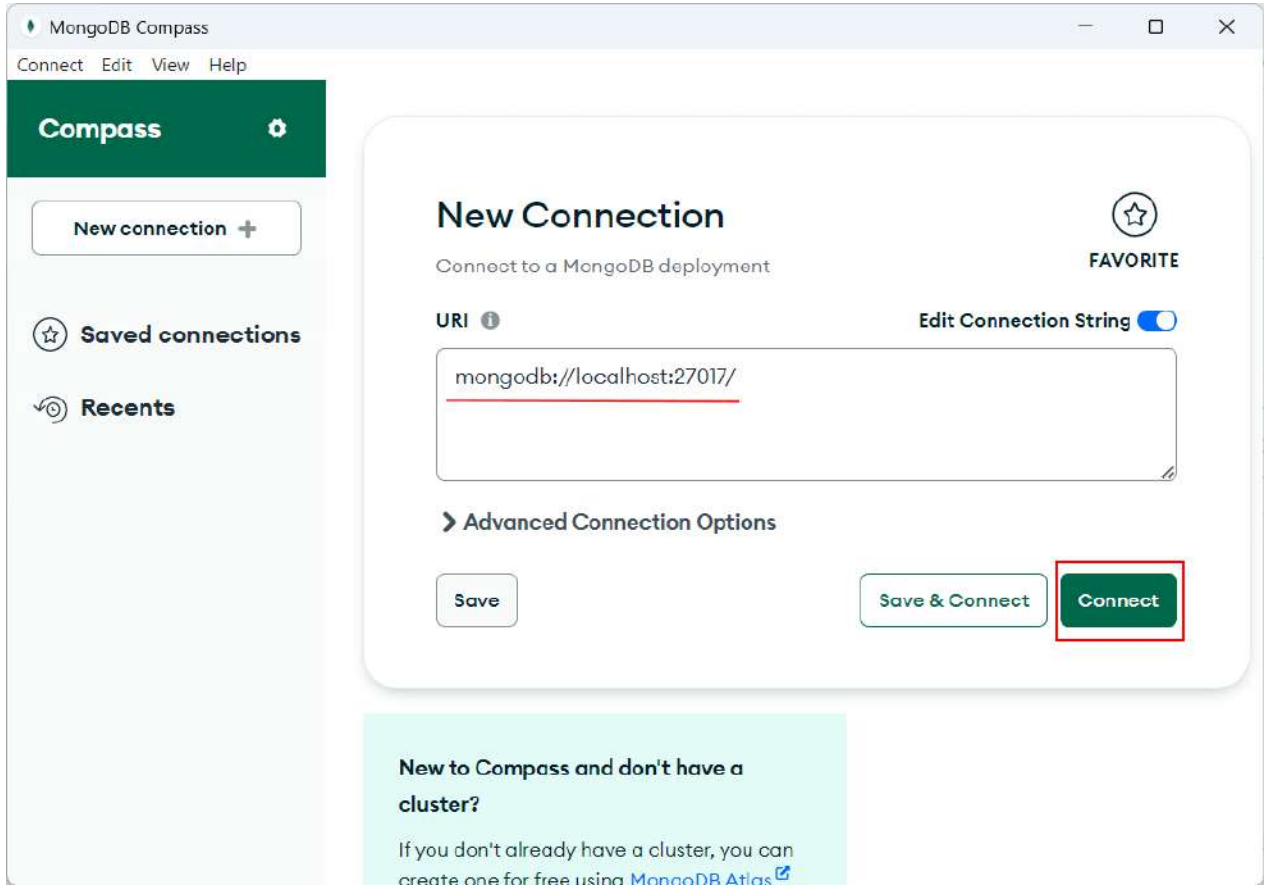
Дереккөз:https://www.mongodb.com/try/download/compass

MONGODB COMPASS ІСКЕ ҚОСУ

Файл жүктелгеннен кейін алдымен **mongod** серверін іске қосамыз, содан соң жүктелген **MongoDB Compass** файлын ашамыз.

COMPASS АРҚЫЛЫ ЖЕРГІЛІКТІ MONGODB СЕРВЕРІНЕ ҚОСЫЛУ

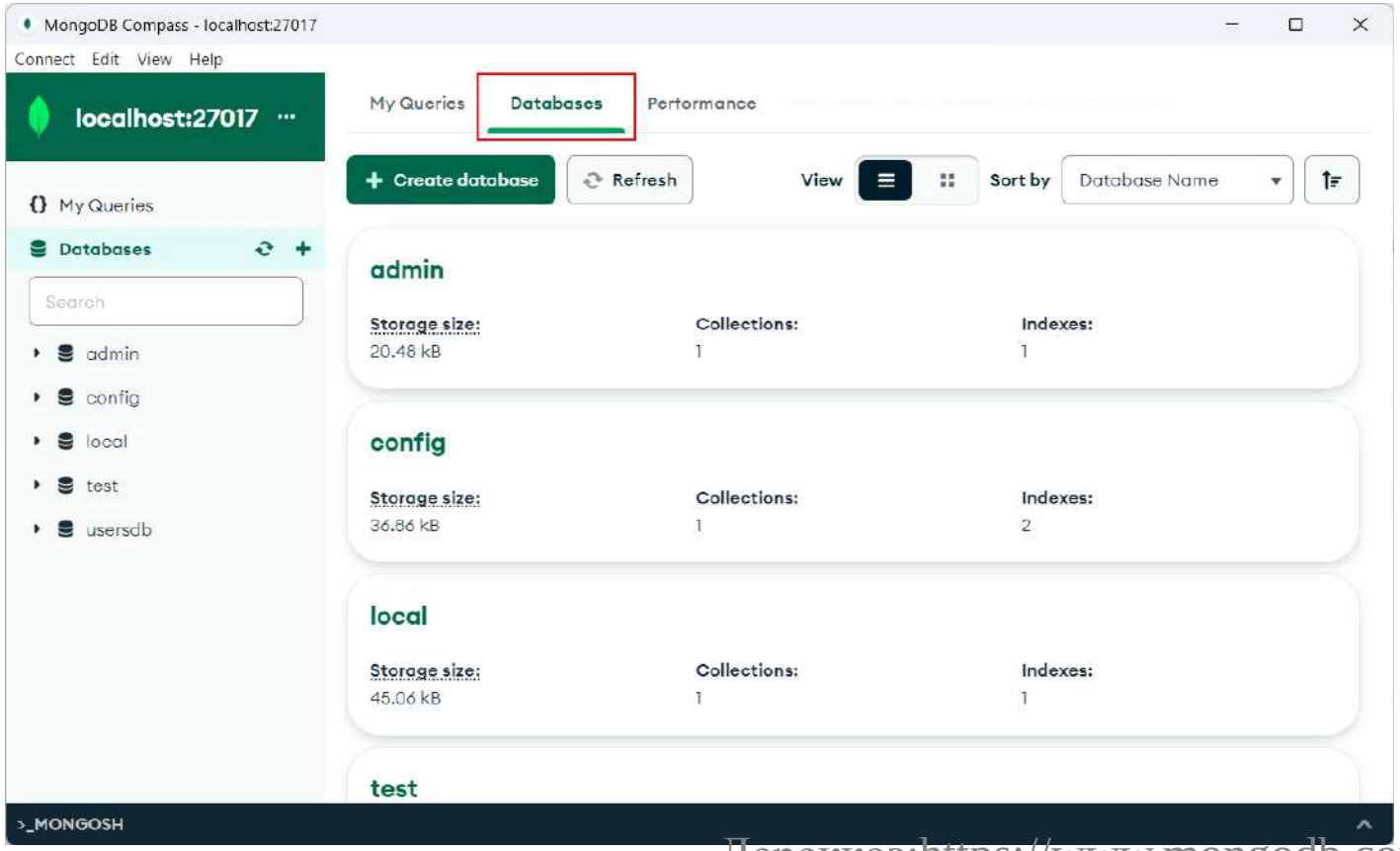
- Әдепкі бойынша жаңа қосылу құру терезесі ашылады. Онда тек бір ғана өріс бар – URI, яғни қосылу жолын енгізу керек. Көбіне бұл өрісте жергілікті серверге арналған қосылу жолы автоматты түрде көрсетіледі:mongodb://localhost:27017
- Қосылу жолы MongoDB серверінің қай жерде іске қосылғанына (жергілікті компьютерде немесе желідегі басқа ресурсқа), логин мен парольге және басқа баптауларға байланысты өзгеруі мүмкін. Бұл жағдайда біз алдыңғы тақырыпта орнатылған жергілікті MongoDB серверіне қосыламыз, сондықтан mongodb://localhost:27017 мәнін қалдырамыз (немесе өріс бос болса, қолмен енгіземіз) және Connect батырмасын басамыз.



Дереккөз:https://www.mongodb.com/try/download/compass

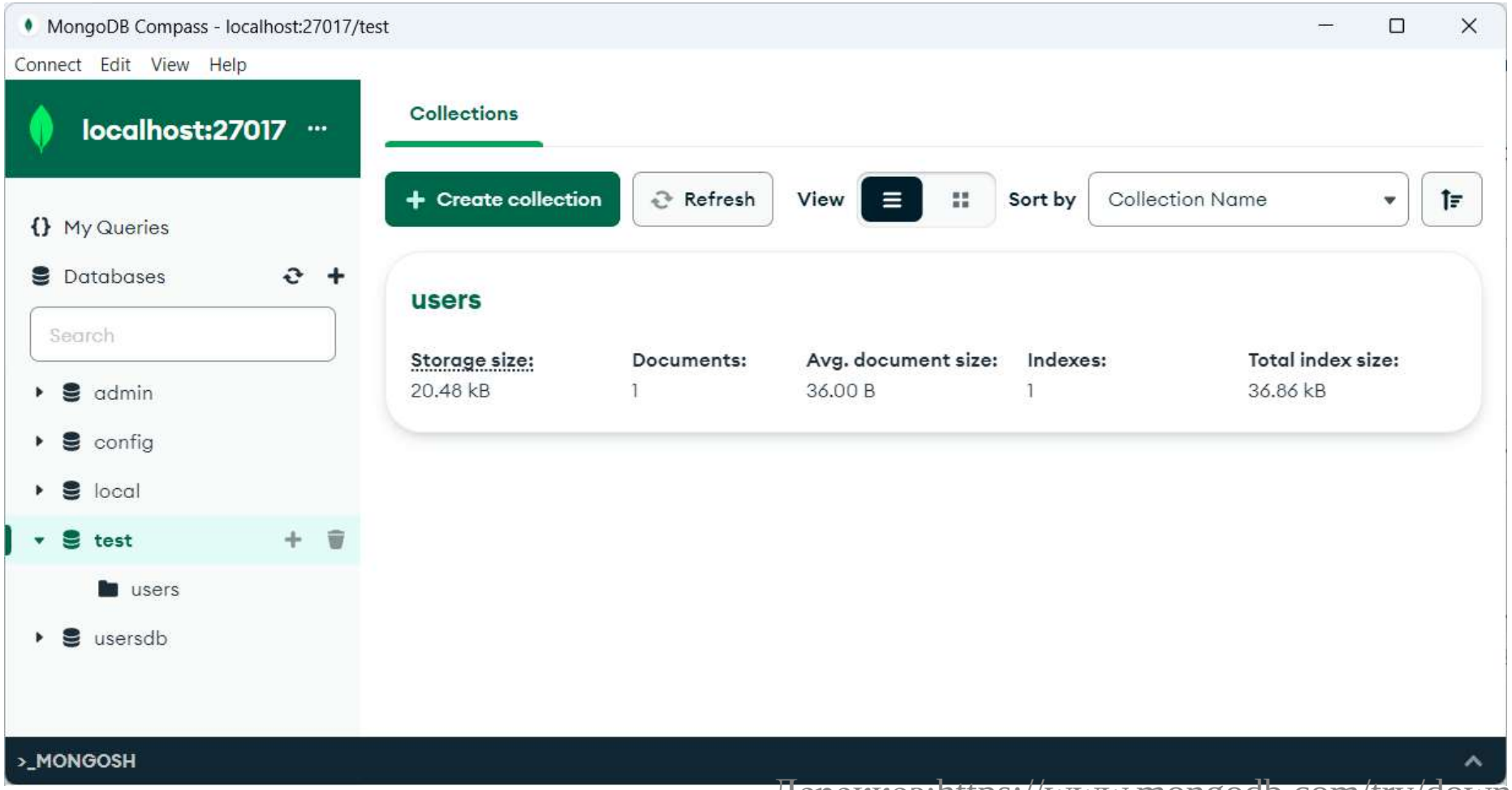
MONGODB COMPASS-ТА ДЕРЕКТЕР ҚОРЛАРЫ

- Белгілі бір деректер қорын таңдап, оның ішіндегі коллекциялар тізімін, олардың көлемін және сақталған деректер көлемін көруге болады.



Дереккөз:https://www.mongodb.com/try/download/compass

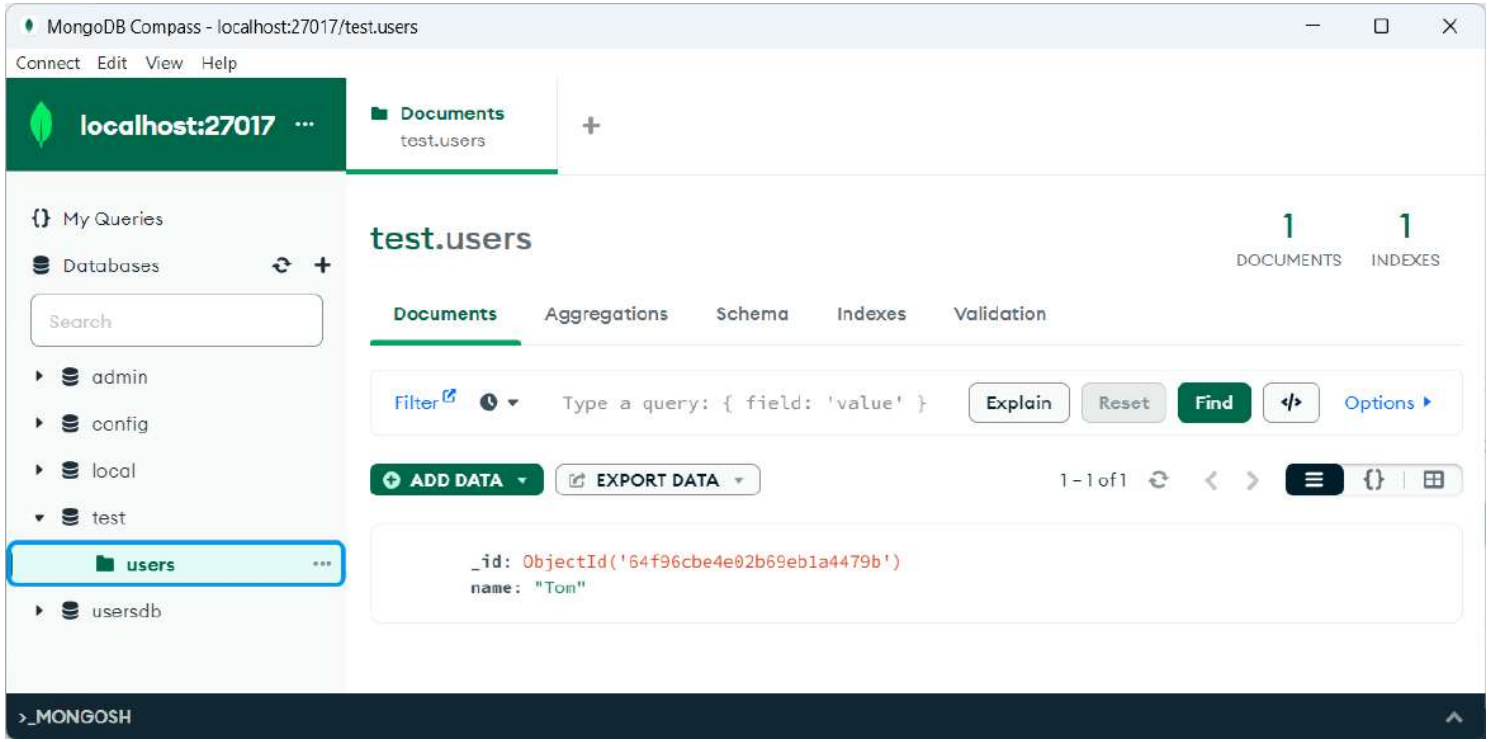
- Берілген мысалда **test** деректер қорының коллекциялары көрсетілген. Олар алдыңғы тақырыпта **mongosh** қабығы арқылы құрылған.
- Белгілі бір коллекцияны таңдаған кезде, ондағы барлық деректер графикалық түрде көрсетіледі.



Дереккөз:https://www.mongodb.com/try/download/compass

COMPASS МҮМКІНДІКТЕРІ

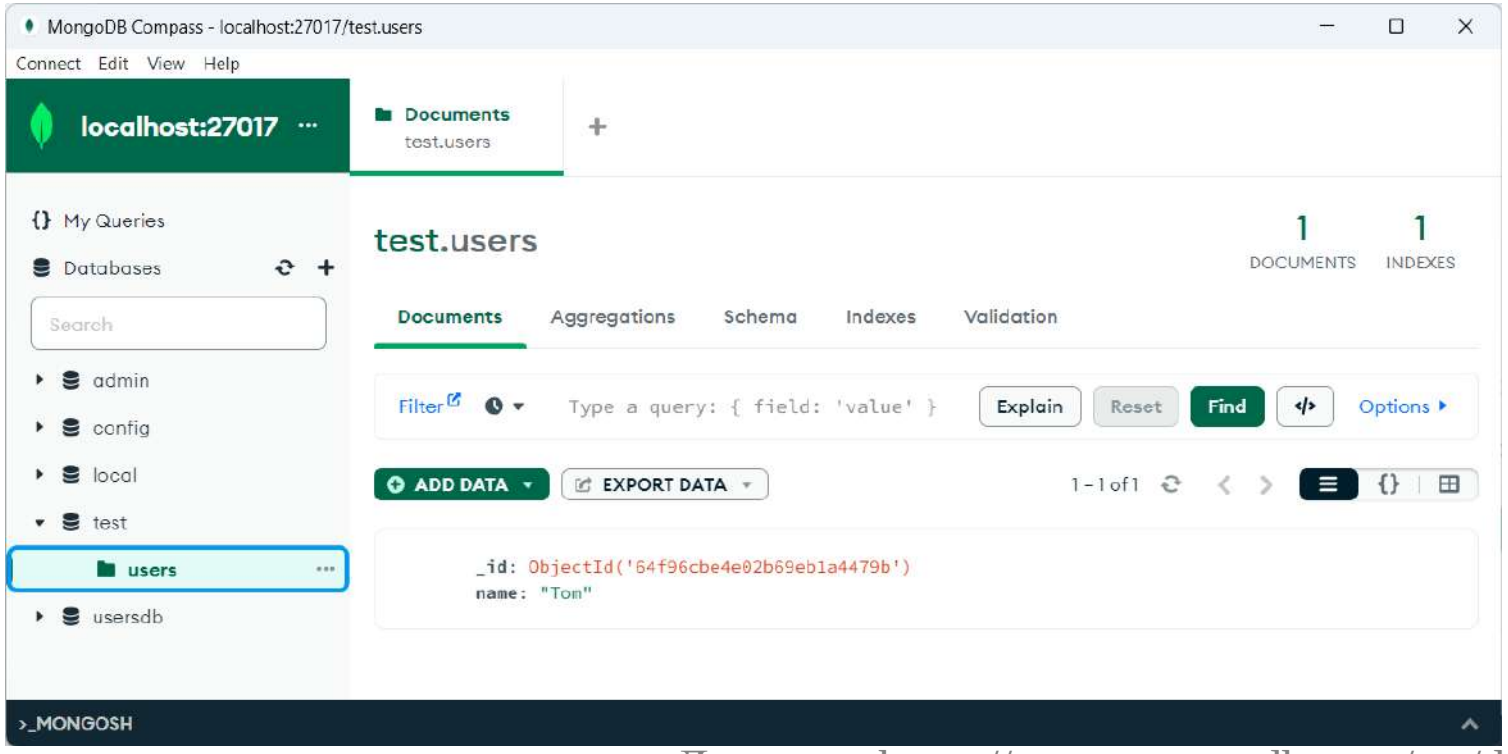
MongoDB Compass бағдарламасының графикалық интерфейсі арқылы деректерді басқаруға болады, яғни жаңа деректер қосу; бар деректерді өзгерту; деректерді жою мүмкіндігі бар.



Дереккөз:https://www.mongodb.com/try/download/compass

COMPASS МҮМКІНДІКТЕРІ

MongoDB Compass бағдарламасының графикалық интерфейсі арқылы деректерді басқаруға болады, яғни жаңа деректер қосу; бар деректерді өзгерту; деректерді жою мүмкіндігі бар.



Дереккөз:https://www.mongodb.com/try/download/compass

MONGODB ДЕРЕКТЕР ТИПТЕРІ

НЕГІЗГІ САНДЫҚ ТИПТЕР

1

Int32

32-биттік бүтін сан (Int32) - тауар саны немесе жас сияқты шағын және орташа мәндегі бүтін сандар үшін өте қолайлы.

1

Int64 (Long)

64-биттік бүтін сан (Int64) - өте үлкен бүтін сандармен жұмыс істеу үшін қажет, мысалы, идентификаторлар немесе Int32 ауқымынан асатын есептегіштер.

10

Double

Қос дәлдікті жылжымалы нүктелі сандар (Double) - баға немесе өлшемдер сияқты ондық сандарды сақтауға арналған стандартты тип, бірақ дәлдіктің белгілі бір дәрежеде жоғалуы мүмкін.

Decimal128

128-биттік ондық сандар (Decimal128) - жоғары дәлдікті толық қамтамасыз етеді. Қаржылық есептеулер үшін ең дұрыс таңдау, себебі ең аз қателікке де жол берілмейді.

Жолдық тип, логикалық тип және NULL мәні

String

мәтіндік деректерді сақтауға арналған негізгі тип. UTF-8 кодтамасын қолданады, бұл кез келген тілдің символларымен жұмыс істеуге мүмкіндік береді. Атауларды, сипаттамаларды және кез келген мәтіндік өрістерді сақтау үшін өте қолайлы.

Boolean

true немесе false мәндерін қабылдайтын логикалық мәнді білдіреді. Жалаушалар (флагтар), күйлер мен шарттарды көрсету үшін кеңінен қолданылады, мысалы, «белсенді» немесе «жарияланған».

Null

берілген өріс үшін деректердің жоқ екенін көрсететін арнайы мән. Бұл бос жолға немесе нөлге тең емес, нақты мәннің өзектілігі жоқ немесе мүлде жоқ екенін білдіреді.

Массивтер және кірістірілген объектілер

MongoDB массивтер мен кірістірілген объектілерді қолдану арқылы икемді әрі күрделі деректер құрылымдарын құруға мүмкіндік береді. Бұл — құжатқа бағытталған деректер қорларының негізгі артықшылықтарының бірі.

Array (Массив)

Массивтер бір кілттің астында мәндердің реттелген тізімін сақтайды. Бұл мәндер кез келген деректер типінде болуы мүмкін, соның ішінде басқа массивтер немесе объектілер де бар. Бұл тегтер тізімін, пайдаланушы рөлдерін, жинақтарды немесе пікірлерді оңай сақтауға мүмкіндік береді.

Object (Кірістірілген объекті)

Кірістірілген объектілер немесе ішкі құжаттар иерархиялық деректер құрылымдарын құруға мүмкіндік береді. Олар өзара байланысты деректерді бір құжаттың ішінде топтастыруға көмектеседі, бұл сұраныстарды жеңілдетіп, өнімділікті арттырады. Мысалы, пайдаланушы туралы деректердің құрамында мекенжайы немесе байланыс ақпараты бар кірістірілген объект болуы мүмкін.

MongoDB арнайы типі

ObjectId

MongoDB әрбір құжат үшін автоматты түрде жасайтын бірегей 12 байттық идентификатор.

Date (Уақыт)

Деректерді UNIX уақыт форматында сақтайды (1970 жылғы 1 қаңтардан бастап миллисекундпен есептеледі, UTC бойынша).

Timestamp (Уақыт белгісі)

Негізінен MongoDB ішкі мақсаттарында қолданылады, мысалы, репликация операциялары мен өзгерістерді бақылауға арналған.

Дереккөз:https://www.opc-router.com/what-is-mongodb

Қосымша деректер типтері

Binary Data (Бинарлы деректер)

суреттер, аудиофайлдар немесе кез келген басқа құрылымдалмаған бинарлық объектілерді сақтауға мүмкіндік береді. MongoDB бинарлық деректерді жіктеу үшін бірнеше қосалқы типтерді қолдайды.

JavaScript

құжаттардың ішінде тікелей JavaScript кодын сақтауға мүмкіндік береді. Қауіпсіздік пен өнімділік себептерімен қазіргі қосымшаларда сирек қолданылады, бірақ сервер жағында орындалатын кейбір операциялар үшін пайдалы болуы мүмкін.

Regular Expression

күрделі сұраныстарда қолдану үшін регулярлы өрнектерді сақтауға мүмкіндік береді. Бұл мәтінді үлгілер бойынша іздеу және сүзгілеудің қуатты құралы болып табылады..

Symbol (Символ)

Ескірген деректер типі - жолға ұқсас тип. Көп жағдайда оның орнына стандартты жолдарды (String) қолданған дұрыс. Бұл тип қайталанатын жолдық мәндерді сақтау үшін оңтайландыруға арналған, бірақ басқа оңтайландыру әдістерінің дамуы нәтижесінде өзектілігі азайды.

Деректер типтерімен жұмыс істеудің маңызды ерекшеліктері

- ✓ MongoDB BSON (Binary JSON)-ді қолданады - JSON құжаттарының бинарлық көрінісі. BSON жылдамдық пен өлшемге оңтайландырылған, деректерді тиімді сақтау мен беру мүмкіндігін қамтамасыз етеді. Бұл MongoDB-ге таза мәтінмен жұмыс істеуге қарағанда деректерді әлдеқайда жылдам өңдеуге мүмкіндік береді.
- ✓ MongoDB деректер типтеріне қатал қарайды. Бұл дегеніміз, сұраныстарды орындағанда "age": "30" (жол) және "age": 30 (сан) мүлде әртүрлі мәндер ретінде қарастырылады. Егер деректер қорыңызда бірдей өрістері бар, бірақ типтері әртүрлі құжаттар болса, сұраныста типті нақты көрсету немесе екі нұсқаны да өңдеу қажет болады.

Деректер типтерімен жұмыс істеудің маңызды ерекшеліктері

- ✓ MongoDB BSON (Binary JSON)-ді қолданады - JSON құжаттарының бинарлық көрінісі. BSON жылдамдық пен өлшемге оңтайландырылған, деректерді тиімді сақтау мен беру мүмкіндігін қамтамасыз етеді. Бұл MongoDB-ге таза мәтінмен жұмыс істеуге қарағанда деректерді әлдеқайда жылдам өңдеуге мүмкіндік береді.
- ✓ MongoDB деректер типтеріне қатал қарайды. Бұл дегеніміз, сұраныстарды орындағанда "age": "30" (жол) және "age": 30 (сан) мүлде әртүрлі мәндер ретінде қарастырылады. Егер деректер қорыңызда бірдей өрістері бар, бірақ типтері әртүрлі құжаттар болса, сұраныста типті нақты көрсету немесе екі нұсқаны да өңдеу қажет болады.

MONGODB-ДА ДЕРЕКТЕРДІ МОДЕЛЬДЕУ: ТИІМДІ ҰЙЫМДАСТЫРУДЫҢ НЕГІЗДЕРІ

Деректерді модельдеу дегеніміз не?

Деректерді модельдеу - бұл ақпаратты тиімді ұйымдастыру және олардың арасындағы байланыстарды анықтау процесі. Бұл сіздің қолданбаңыздың деректермен қалай жұмыс істейтінін көрсететін Blueprint іспеттес.

MongoDB-да деректер құжаттар түрінде сақталады, бұл икемділік береді және NoSQL дерекқорларының басты артықшылығы болып табылады.

MongoDB-ның икемді деректер моделі

Әртүрлі құжаттар

Бір коллекциядағы құжаттар әртүрлі өрістерге ие болуы мүмкін, бұл әртүрлі деректер түрлерін бір жерде сақтауға мүмкіндік береді.

Өзгермелі деректер типтері

Бір өрістің дерек типі құжаттар арасында өзгеруі мүмкін, бұл схеманы икемді етеді.

Қолданбаға ыңғайлы

Жиі бірге қолданылатын деректер бірге сақталады, бұл қолданбаның жұмысын жеңілдетеді және жылдамдатады.

Құжаттар мен коллекциялар: негізгі құрылымдар

Құжат

Құжат – бұл JSON тәрізді деректердің негізгі бірлігі. Ол мәліметтерді өріс-мәндер жұптары ретінде сақтайды. Мысалы, бір құжат пайдаланушының атын, электронды поштасын және жасын қамтуы мүмкін.

Коллекция

Коллекция – бұл құжаттар жиыны. Бір коллекциядағы құжаттардың құрылымы міндетті түрде бірдей болмауы мүмкін, бұл MongoDB-ның икемділігін көрсетеді.

Қатынастарды көрсету тәсілдері

MongoDB-да деректер арасындағы қатынастарды орнатудың екі негізгі тәсілі бар:

Құжаттарды біріктіру (Embedding)

Байланысты деректерді бір құжатта сақтау. Мысалы, тапсырыс құжатына клиенттің мекенжайын енгізу. Бұл оқырман операциялары үшін өте тиімді.

Сілтемелеу (Referencing)

Деректерді бөлек коллекцияларда сақтап, ID арқылы байланыстыру. Мысалы, пайдаланушы құжатына оның барлық жазбаларының ID-ларын сақтау. Бұл деректерді қайталауды азайтады.

Қолдану мысалы: қызметкерлер мен бөлімдер

Қызметкер құжатында бөлімді біріктіру

```
{
  "_id": "emp123",
  "name": "Айгүл",
  "department": {
    "name": "Маркетинг",
    ""id": "dep456"
  }
}
```

Пікірлерді бөлек коллекцияда сақтау

```
{
 "_id": "prod789",
 "name": "Жаңа өнім",
 "comments": ["com111", "com222"]
}
{
 "_id": "com111",
 "text": "Тамаша өнім!",
 "product_id": "prod789"
}
```

Модельдеуді жоспарлау және жетілдіру

Қарапайым схемадан бастаңыз

Алғашқыда деректер моделін мүмкіндігінше қарапайым етіп жасаңыз. Бұл сізге қолданбаңыздың негізгі функционалдығын жылдам іске қосуға көмектеседі.

Қажеттілікке қарай өзгертіңіз

Қолданба дамыған сайын және деректерді пайдалану үлгілері анықталған сайын, деректер моделін өзгертіп, оңтайландырыңыз.

Қолдануды анықтау

Деректерді қалай қолданатыныңызды (оқу, жазу, жаңарту) алдын ала анықтау қиындықтардан аулақ болуға көмектеседі.

Өнімділікті арттыру үшін деректерді бірге сақтау

Бір сұрауда жиі қолданылатын деректер

Егер сізге деректер жиі бірге қажет болса, оларды бір құжатта немесе коллекцияда сақтау сұрау жылдамдығын айтарлықтай арттырады. Бұл деректер базасының бірнеше сұраныс жасауын болдырмайды.

Үлкен көлемдегі деректерді бөлу

Сирек қолданылатын немесе өте үлкен көлемдегі деректерді бөлек коллекцияларға шығару негізгі коллекцияның өлшемін кішірейтіп, жад пен өткізу қабілетін үнемдейді. Бұл жүйенің жалпы өнімділігін жақсартады.

MongoDB-да деректерді модельдеудің үздік тәжірибелері

Икемді схеманы қолданыңыз

Қолданба қажеттіліктеріне сәйкес деректер құрылымын икемді өзгертіңіз. MongoDB-ның схемасыз табиғатын пайдаланыңыз.

Құжат өлшемін шектеңіз

Әр құжаттың өлшемі шектеулі екенін есте сақтаңыз (16 МБ). Оңтайлы өнімділік үшін құжаттарды тым үлкен етпеңіз.

Индекстерді дұрыс пайдаланыңыз

Жиі сұрау жасалатын өрістерге индекстер қосыңыз. Бұл сұрау жылдамдығын едәуір арттырады.

Қатынастарды дұрыс таңдаңыз

Біріктіру немесе сілтемелеу арасында дұрыс таңдау жасау қолданбаның өнімділігі мен масштабталуына әсер етеді.

MongoDB-да деректерді модельдеудің үздік тәжірибелері

Икемді схеманы қолданыңыз

Қолданба қажеттіліктеріне сәйкес деректер құрылымын икемді өзгертіңіз. MongoDB-ның схемасыз табиғатын пайдаланыңыз.

Құжат өлшемін шектеңіз

Әр құжаттың өлшемі шектеулі екенін есте сақтаңыз (16 МБ). Оңтайлы өнімділік үшін құжаттарды тым үлкен етпеңіз.

Индекстерді дұрыс пайдаланыңыз

Жиі сұрау жасалатын өрістерге индекстер қосыңыз. Бұл сұрау жылдамдығын едәуір арттырады.

Қатынастарды дұрыс таңдаңыз

Біріктіру немесе сілтемелеу арасында дұрыс таңдау жасау қолданбаның өнімділігі мен масштабталуына әсер етеді.