



Alfa·Bank

SPB
DOT
NET

Молчанов Николай

Руководитель центра компетенции .NET
Альфа-Банк

Проповедник **MongoDB** в
высоконагруженных приложениях





Елисеев Дмитрий

SQL Team Lead в ЦРТ

Нейромонах **SQL Server** в OLTP
высоконагруженных приложениях



DocumentDB

VS

RelationDB













ПЛАН

1. Чем отличается **DocumentDB** от **RelationDB**.
2. Что такое **MongoDB** и что такое **SQL Server\PostgreSQL**
3. Рассмотрим философию проектирования хранения в 2 подхода
4. Сравним производительности в разных БД
5. Сравнение инструментов
6. Выводы

СТАТИСТИКА

Rank			DBMS	Database Model	Score		
Aug 2019	Jul 2019	Aug 2018			Aug 2019	Jul 2019	Aug 2018
1.	1.	1.	Oracle	Relational, Multi-model	1339.48	+18.22	+27.45
2.	2.	2.	MySQL	Relational, Multi-model	1253.68	+24.16	+46.87
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	1093.18	+2.35	+20.53
4.	4.	4.	PostgreSQL	Relational, Multi-model	481.33	-1.94	+63.83
5.	5.	5.	MongoDB	Document	404.57	-5.36	+53.59
6.	6.	6.	IBM Db2	Relational, Multi-model	172.95	-1.19	-8.89
7.	7.	8.	Elasticsearch	Search engine, Multi-model	149.08	+0.27	+10.97
8.	8.	7.	Redis	Key-value, Multi-model	144.08	-0.18	+5.51
9.	9.	9.	Microsoft Access	Relational	135.33	-1.98	+6.24
10.	10.	10.	Cassandra	Wide column	125.21	-1.80	+5.63

СТАТИСТИКА

Rank			DBMS	Database Model	Score		
Aug 2019	Jul 2019	Aug 2018			Aug 2019	Jul 2019	Aug 2018
1.	1.	1.	MongoDB 	Document	404.57	-5.36	+53.59
2.	2.	2.	Amazon DynamoDB 	Multi-model 	56.57	+0.15	+4.91
3.	3.	3.	Couchbase 	Document, Multi-model 	33.83	+0.12	+0.88
4.	4.	4.	Microsoft Azure Cosmos DB 	Multi-model 	29.94	+0.85	+10.41
5.	5.	5.	CouchDB	Document	19.76	+0.08	+1.33
6.	6.	6.	MarkLogic 	Multi-model 	14.46	+0.65	+3.25
7.	7.	7.	Firebase Realtime Database	Document	11.64	+0.36	+4.13
8.	8.	8.	OrientDB	Multi-model 	6.28	+0.59	+1.37
9.	9.	 16.	Google Cloud Firestore	Document	6.26	+1.03	+4.04
10.	10.	 11.	Google Cloud Datastore	Document	5.90	+0.88	+2.58

КТО ИСПОЛЬЗУЕТ MONGODB

8

70% слышали

10% использовали

1% 1T+

GITHUB

<https://github.com/kroniak/mongodb-benchmark-samples>



В ЧЕМ СИЛА?

В ЧЕМ СИЛА?

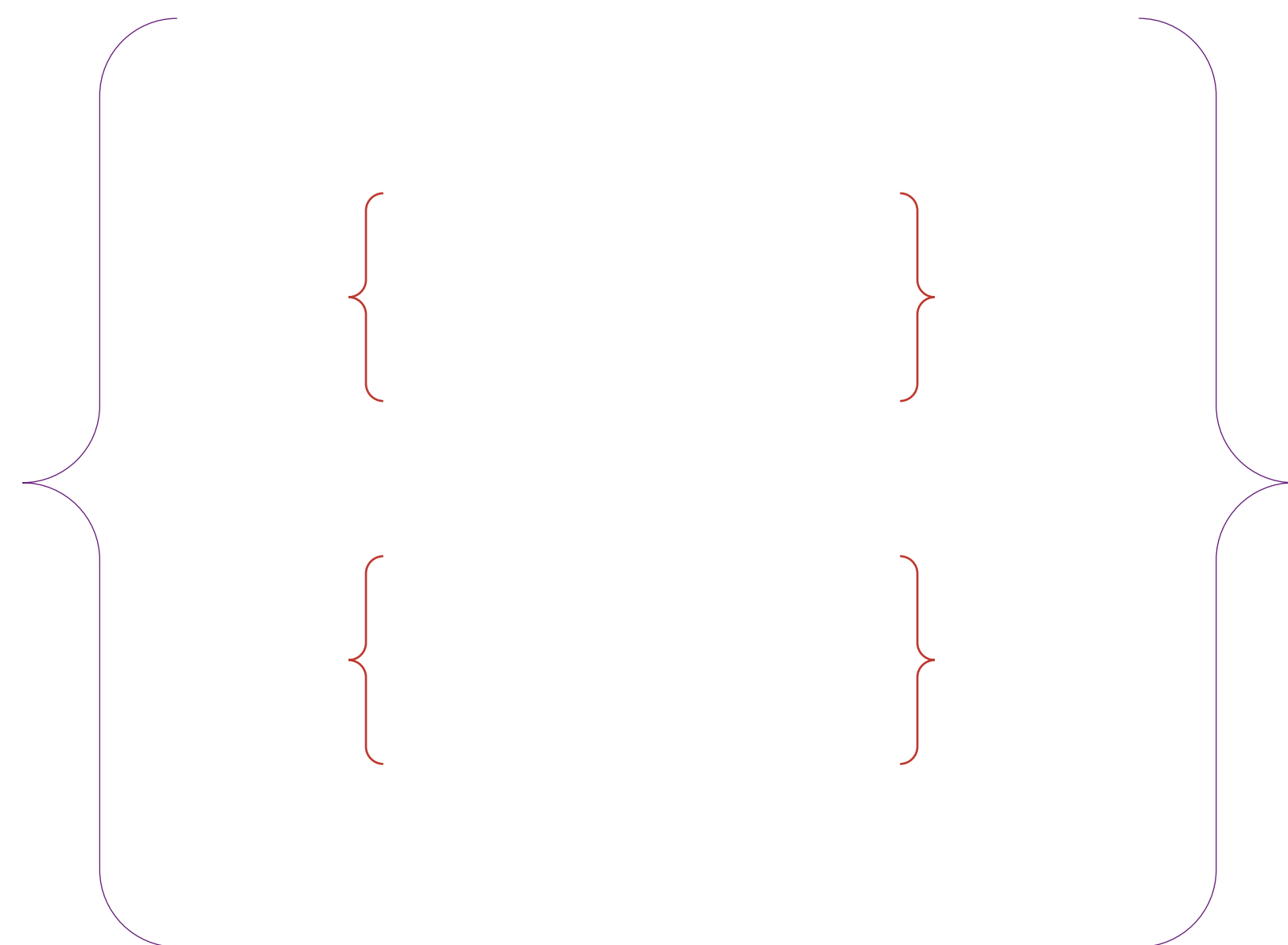
1. DocumentDB хранит набор документов, а RelationDB набор строк в таблицах

В ЧЕМ СИЛА?

1. DocumentDB хранит набор документов, а RelationDB набор строк в таблицах
2. DocumentDB и noSQL не предполагает связи между разными документами

В ЧЕМ СИЛА?

1. DocumentDB хранит набор документов, а RelationDB набор строк в таблицах
2. DocumentDB и noSQL не предполагает связи между разными документами





D: n баз данных

n коллекций

n *любых*
документов



D: n баз данных

n коллекций

n *любых*
документов

R: n баз данных

n таблиц

n *фикс*
строк

DOCUMENT BSON : JSON DOCUMENT

DOCUMENT BSON : JSON DOCUMENT

В JSON НЕТ МНОГИХ ТИПОВ ДАННЫХ

BSON DOCUMENT

```
{
  "_id" : "b5aa33132561-1531670109414",
  "hostname" : "b5aa33132561",
  "startTime" : ISODate("2018-07-15T15:55:09.000+0000"),
  "cmdLine" : {
    "net" : {
      "bindIpAll" : true
    }
  },
  "pid" : NumberLong(1),
  "buildinfo" : {
    "modules" : [

  ],
    "versionArray" : [
      NumberInt(4),
      NumberInt(0),
      NumberInt(0),
      NumberInt(0)
    ]
  }
}
```

< 16 MB

BSON DOCUMENT

```
{
  "_id" : "b5aa33132561-1531670109414",
  "hostname" : "b5aa33132561",
  "startTime" : ISODate("2018-07-15T15:55:09.000+0000"),
  "cmdLine" : {
    "net" : {
      "bindIpAll" : true
    }
  },
  "pid" : NumberLong(1),
  "buildInfo" : {
    "modules" : [

  ],
  "versionArray" : [
    NumberInt(4),
    NumberInt(0),
    NumberInt(0),
    NumberInt(0)
  ]
}
```

ObjectId

Примитив

Примитив

BSON DOCUMENT

```
{
  "_id" : "b5aa33132561-1531670109414",
  "hostname" : "b5aa33132561",
  "startTime" : ISODate("2018-07-15T15:55:09.000+0000"),
  "cmdLine" : {
    "net" : {
      "bindIpAll" : true
    }
  },
  "pid" : NumberLong(1),
  "buildinfo" : {
    "modules" : [

    ],
    "versionArray" : [
      NumberInt(4),
      NumberInt(0),
      NumberInt(0),
      NumberInt(0)
    ]
  }
}
```

Вложенный документ

Вложенный документ

BSON DOCUMENT

```
{
  "_id" : "b5aa33132561-1531670109414",
  "hostname" : "b5aa33132561",
  "startTime" : ISODate("2018-07-15T15:55:09.000+0000"),
  "cmdLine" : {
    "net" : {
      "bindIpAll" : true
    }
  },
  "pid" : NumberLong(1),
  "buildinfo" : {
    "modules" : [

    ],
    "versionArray" : [
      NumberInt(4),
      NumberInt(0),
      NumberInt(0),
      NumberInt(0)
    ]
  }
}
```

Вложенный массив



SHARDING

SHARDING POSTGRESQL

Shard 1

Stores

id	name
1	my book store
5	my other store

Products

id	name	store_id
1	foo	1
2	bar	1
3	baz	1

Purchases

id	product_id	store_id	price
1	2	1	1000
2	1	1	1200
3	3	1	1199

Shard 2

Stores

id	name
2	my sock store
6	old things

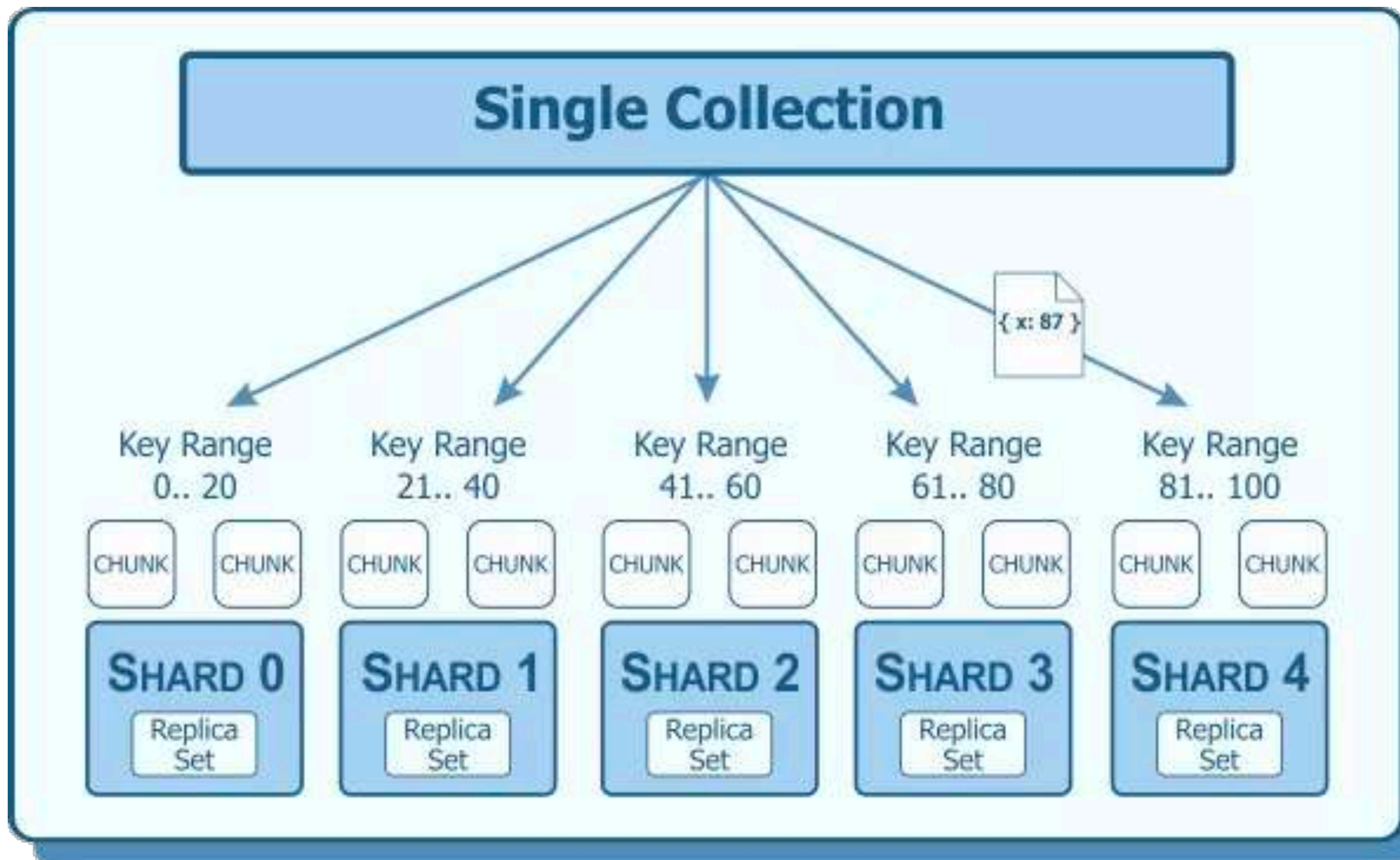
Products

id	name	store_id
33	new socks	2
34	old socks	6
35	old tie	6

Purchases

id	product_id	store_id	price
102	35	6	600
43	33	2	800

SHARDING MONGODB



NO SHARDING

MongoDB

Формат данных

Бинарный без схемы
< 16mb

Отношения

Масштабирование

RelationDB

Фиксированные строки
+ бинарный без схемы
без ограничений

MongoDB

Формат данных

Бинарный без схемы
< 16mb

Отношения

Отсутствуют между
разными документами

Масштабирование

RelationDB

Фиксированные строки
+ бинарный без схемы
без ограничений

Есть связи в виде
foreign key

MongoDB

Формат данных

Бинарный без схемы
< 16mb

Отношения

Отсутствуют между
разными документами

Масштабирование

Простое горизонтальное
масштабирование

RelationDB

Фиксированные строки
+ бинарный без схемы
без ограничений

Есть связи в виде
foreign key

Есть, но с причудами

MongoDB

Формат данных

Бинарный без схемы
< 16mb

Отношения

Отсутствуют между
разными документами

Масштабирование

Простое горизонтальное
масштабирование



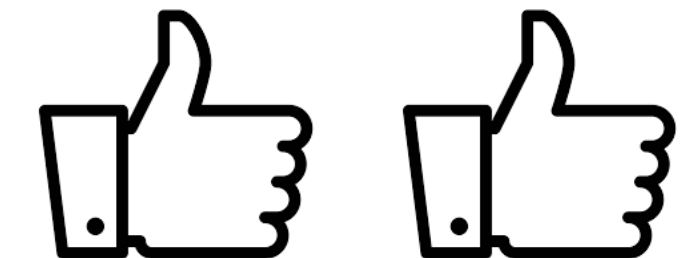
RelationDB

30

Фиксированные строки
+ бинарный без схемы
без ограничений

Есть связи в виде
foreign key

Есть, но с причудами





СТРАХИ

СТРАХИ

где ACID?

где ACID?

нет схемы

где ACID?

нет схемы

где данные?

MONGODB

было

NO transaction, NO sharding, SIMPLE replica nodes,
FIRST memory storage MMAP
Cross-Platforms

было

NO transaction, NO sharding, SIMPLE replica nodes,
FIRST memory storage MMAP
Cross-Platforms



стало

YES transaction, YES sharding, YES replica sets NEW protocol
ROW storage WiredTiger, ZIP data, ADD linearizable, ADD schema validation

Method	Count	Mean	Error	StdDev	Median	Ratio	RatioSD
Save	10	848.9 us	16.92 us	44.57 us	838.9 us	1.00	0.00
SaveBulk	10	831.3 us	16.40 us	26.49 us	830.8 us	0.97	0.07
SaveAsync	10	900.6 us	17.57 us	32.13 us	894.7 us	1.06	0.08
SaveParallel	10	2,055.7 us	41.79 us	110.81 us	2,035.2 us	2.43	0.17
SaveWithTransaction	10	835.1 us	16.70 us	32.57 us	839.0 us	0.99	0.08
SaveBulkWithTransaction	10	846.7 us	16.21 us	15.93 us	850.0 us	1.00	0.06
SaveWithTransactionAsync	10	913.5 us	18.11 us	35.33 us	907.2 us	1.08	0.07
Save	50	1,180.6 us	23.37 us	35.68 us	1,182.6 us	1.00	0.00
SaveBulk	50	1,199.8 us	23.87 us	37.85 us	1,201.7 us	1.02	0.05
SaveAsync	50	1,289.6 us	25.61 us	67.92 us	1,277.5 us	1.12	0.06
SaveParallel	50	8,420.7 us	189.05 us	554.46 us	8,408.1 us	7.15	0.41
SaveWithTransaction	50	1,212.2 us	23.84 us	41.75 us	1,208.9 us	1.02	0.04
SaveBulkWithTransaction	50	1,196.7 us	23.83 us	57.09 us	1,191.1 us	1.01	0.06
SaveWithTransactionAsync	50	1,246.9 us	24.81 us	47.81 us	1,249.3 us	1.04	0.05

Method	Count	Mean	Error	StdDev	Median	Ratio	RatioSD
Save	10	848.9 us	16.92 us	44.57 us	838.9 us	1.00	0.00
SaveBulk	10	831.3 us	16.40 us	26.49 us	830.8 us	0.97	0.07
SaveAsync	10	900.6 us	17.57 us	32.13 us	894.7 us	1.06	0.08
SaveParallel	10	2,055.7 us	41.79 us	110.81 us	2,035.2 us	2.43	0.17
SaveWithTransaction	10	835.1 us	16.70 us	32.57 us	839.0 us	0.99	0.08
SaveBulkWithTransaction	10	846.7 us	16.21 us	15.93 us	850.0 us	1.00	0.06
SaveWithTransactionAsync	10	913.5 us	18.11 us	35.33 us	907.2 us	1.08	0.07
Save	50	1,180.6 us	23.37 us	35.68 us	1,182.6 us	1.00	0.00
SaveBulk	50	1,199.8 us	23.87 us	37.85 us	1,201.7 us	1.02	0.05
SaveAsync	50	1,289.6 us	25.61 us	67.92 us	1,277.5 us	1.12	0.06
SaveParallel	50	8,420.7 us	189.05 us	554.46 us	8,408.1 us	7.15	0.41
SaveWithTransaction	50	1,212.2 us	23.84 us	41.75 us	1,208.9 us	1.02	0.04
SaveBulkWithTransaction	50	1,196.7 us	23.83 us	57.09 us	1,191.1 us	1.01	0.06
SaveWithTransactionAsync	50	1,246.9 us	24.81 us	47.81 us	1,249.3 us	1.04	0.05

BSON VALIDATOR

```
{
  required: [ "hostname", "statTime", "pid", "buildinfo" ],
  properties: {
    name: {
      bsonType: "hostname",
      description: "must be a string and is required"
    },
    pid: {
      bsonType: "long",
      minimum: 1,
      maximum: 65536,
      description: "must be an integer in [ 1, 65536 ] and is required"
    }
  }
}
```

BSON VALIDATOR

41

BenchmarkDotNet=v0.11.5, OS=macOS Mojave 10.14.6 (18G87) [Darwin 18.7.0]
Intel Core i7-8850H CPU 2.60GHz (Coffee Lake), 1 CPU, 12 logical and 6 physical cores
.NET Core SDK=2.2.401
[Host] : .NET Core 2.2.6 (CoreCLR 4.6.27817.03, CoreFX 4.6.27818.02), 64bit RyuJIT
Core : .NET Core 2.2.6 (CoreCLR 4.6.27817.03, CoreFX 4.6.27818.02), 64bit RyuJIT

Job=Core Runtime=Core

Method	Mean	Error	StdDev	Ratio	RatioSD
Save	1.316 s	0.0261 s	0.0490 s	1.00	0.00
SaveWithValidation	1.413 s	0.0278 s	0.0331 s	1.08	0.05

SQL SERVER

было в 1989

YES transaction, YES indexes, NO sharding, NO cluster

было в 1989

YES transaction, YES indexes, NO sharding, NO cluster

BISM

DQS

SSIS

.NET

Java

Python

ПУТЬ РАЗВИТИЯ

45

было в 1989

YES transaction, YES indexes, NO sharding, NO cluster

BISM

SSIS

DQS

.NET

Java

Python

стало в 2019

NO sharding, AlwaysOn, JSON, ZIP data, ADD JSON schema validation,
Always Encrypted, GraphDB, AutoTuning, Cross-Platforms

POSTGRESQL

было в 1970...

YES transaction, YES indexes, NO sharding, NO cluster, nix

было в 1970...

YES transaction, YES indexes, NO sharding, NO cluster, nix

Cluster

Custom types and indexes

Array in table

BSD\MIT Licenses

Script plugins

SP in C\C++

было в 1970...

YES transaction, YES indexes, NO sharding, NO cluster, nix

Cluster

Array in table

Script plugins



Custom types and indexes

SP in C\C++

BSD\MIT Licenses

стало в 2019

YES sharding, Clusters, JSONb, ZIP data, Inheritance, Encrypted, Cross-Platforms

Constraints

Есть

Есть

MongoDB

RelationDB

Constraints

Есть

Есть

Транзакции

Есть, распределенные
по нодам

Есть, есть уровни
изоляции

MongoDB

RelationDB

Constraints

Есть

Есть

Транзакции

Есть, распределенные
по нодам

Есть, есть уровни
изоляции

Отказоустойчивость

Есть

Есть

MongoDB

Constraints

Есть

Транзакции

Есть, распределенные
по нодам

Отказоустойчивость

Есть

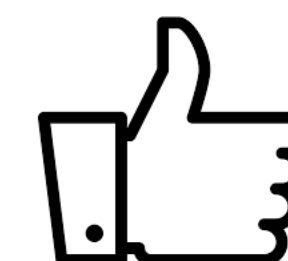


RelationDB

Есть

Есть, есть уровни
изоляции

Есть



ПЛАНИРОВАНИЕ ХРАНИЛИЩА

TOOLTIP

забудьте реляционный подход

Model => ORM => DB

Model => ORM => DB

ΦT => queries = 1 JOIN

Model => ORM => DB

ΦT => queries = 1 JOIN

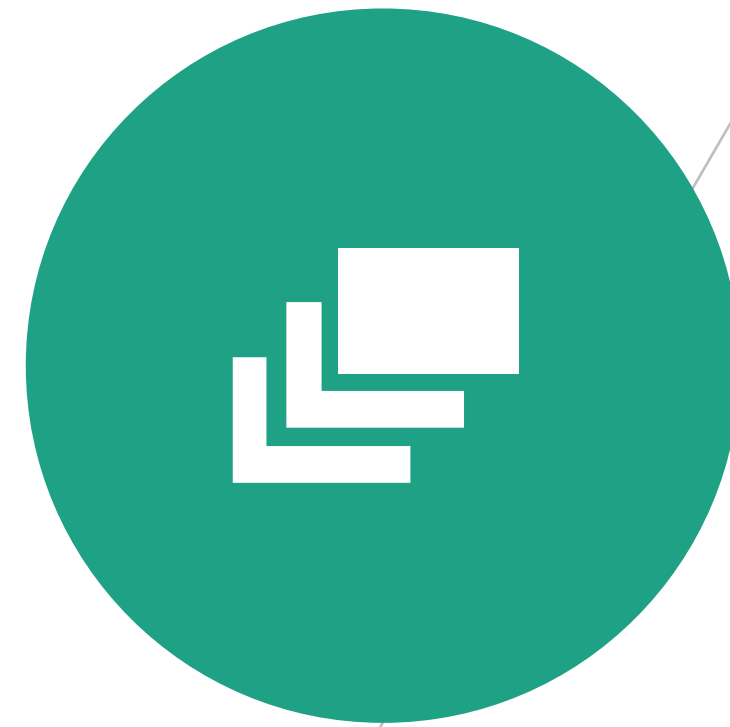
ΦT => 2 queries = 3 JOIN

...

ДВА ПОДХОДА

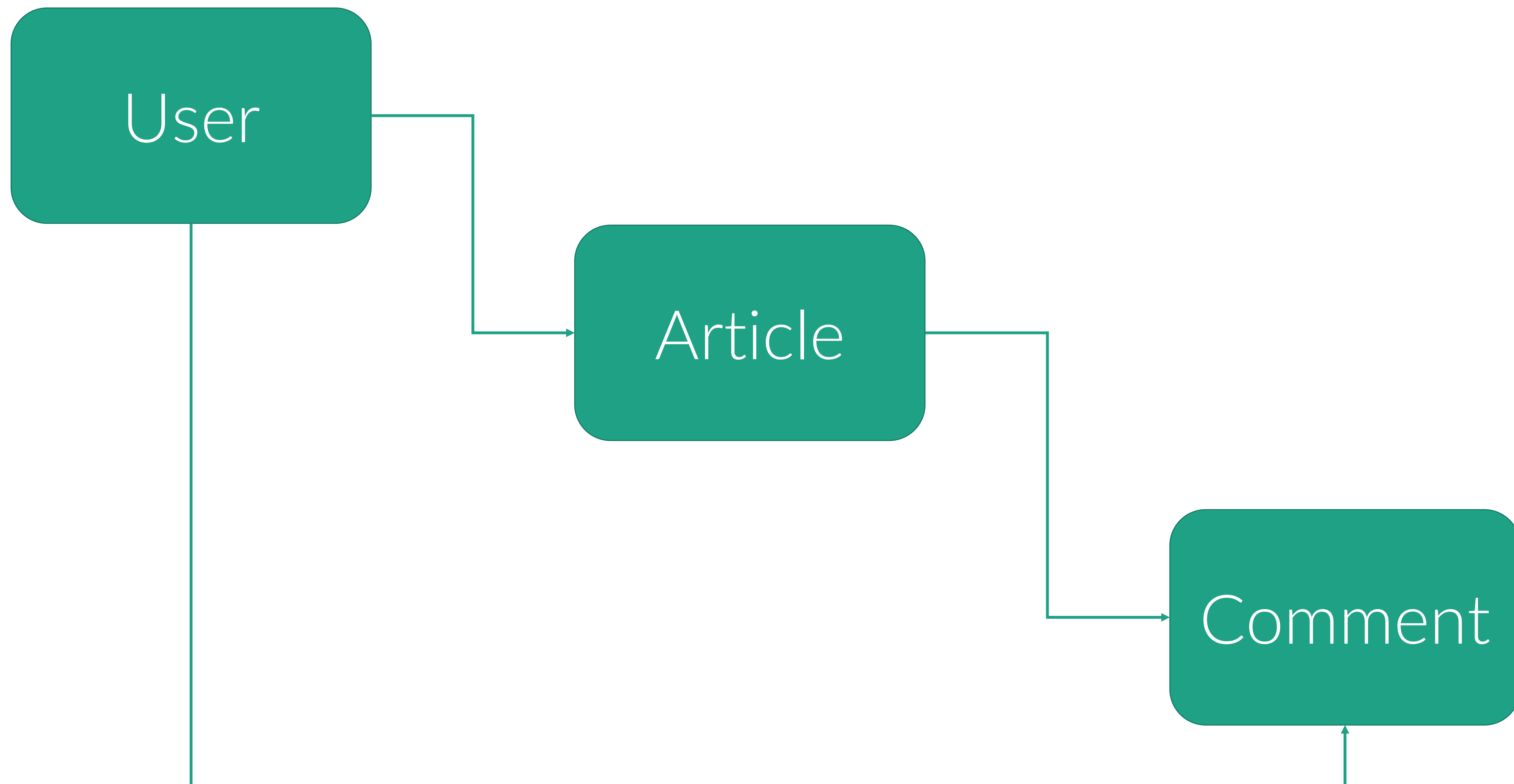
59

от модели



от запросов

что
эффективнее?



ВАРИАНТ 1 ВСТРОЕННЫЕ ДОКУМЕНТЫ

```
{
  "_id" : ObjectId("5b4e59275c005013739f1908"),
  "name" : "Maria",
  "created" : ISODate("2018-07-01T15:00:00.000+0000"),
  "url" : "http://blog-name.ru/maria/",
  "articles" : [
    {
      "name" : "How to learn mongodb",
      "created" : ISODate("2018-07-01T22:00:00.000+0000"),
      "url" : "http://blog-name.ru/maria/how-to-learn-mongodb",
      "comments" : [
        {
          "created" : ISODate("2018-07-01T23:00:00.000+0000"),
          "userId" : ObjectId("5b4e58f65c005013739f1904"),
          "text" : "Wow! You are amazing! I have learn all ongodb for 4 days"
        },
        {
          "created" : ISODate("2018-07-01T23:10:00.000+0000"),
          "userId" : ObjectId("5b4e58f65c005013739f1907"),
          "text" : "No, this is pain!"
        }
      ]
    }
  ]
}
```

Плюсы:

- сохранение за 1 операцию
- поиск возвращает **корневой** документ

Плюсы для SQL\PG:

- большой объем хранения json\jsonb

Минусы в MongoDB:

- ограничение хранения в 16мб
- глубина вложенности важна для индекса
- СЛОЖНО писать обновления

Минусы в SQL\PG:

- низкая скорость поиска


```
{  
  "_id" : ObjectId("5b4e59275c005013739f1909"),  
  "name" : "Maria",  
  "created" : ISODate("2018-07-01T15:00:00.000+0000"),  
  "url" : "http://blog-name.ru/maria/",  
  "articles" : [  
    ObjectId("5b4e59265c005022739f1909"),  
    ObjectId("5b4e59255c005013739f1910"),  
    ObjectId("5b4e59266c005014739f1911"),  
    ObjectId("5b4e59275c005016739f1912")  
  ]  
}
```


Плюсы:

- простое добавление зависимости
- поиск возвращает `_id` всех зависимостей
- просто сделать связь **МНОГИЕ-КО-МНОГИМ**

Минусы для MongoDB:

- оптимально для **небольшого** кол-ва зависимостей
- **необходимо** следить за связями (обновление)

Минусы для SQL\PG

- это нереляционный подход
- не рекомендуется использовать

```
{
  "_id" : ObjectId("5b4e67415c005013739f1932"),
  "name" : "How to learn mongodb",
  "userId" : ObjectId("5b4e67415c005013739f1934"),
  "url" : "http://blog-name.ru/maria/how-to-learn-mongodb",
  "comments" : [
    {
      "created" : ISODate("2018-07-01T23:00:00.000+0000"),
      "userId" : ObjectId("5b4e58f65c005013739f1904"),
      "text" : "Wow! You are amazing! I have learn all ongodb for 4 days"
    },
    {
      "created" : ISODate("2018-07-01T23:10:00.000+0000"),
      "userId" : ObjectId("5b4e58f65c005013739f1907"),
      "text" : "No, this is pain!"
    }
  ]
}
```

Плюсы:

- простое добавление зависимости
- это проще для понимания

Минусы:

- усложнение кол-ва связей
- **удаление** связанных документов **усложнено**
- **увеличение** IOPs

СПРАВНЕНИЕ

Сценарий: 2 типа документа: статья и пользователь

- комментарии встроены в статью
- ссылка на пользователя есть в статье и комментарии
- 1к пользователей
- всего 100к статей
- всего примерно 500к комментариев

Сценарий: 2 типа документа: статья и пользователь

- комментарии встроены в статью
- ссылка на пользователя есть в статье и комментарии
- 1к пользователей
- всего 100к статей
- всего примерно 500к комментариев

Задача: найти все комментарии пользователя по его имени с данными статьи

Объект статья

```
{
  "_id" : ObjectId("5d5b2463dc9e6f16530597e8"),
  "Name" : "animi-aliquid-aut",
  "Url" : "https://тамапа.net",
  "Created" : ISODate("2019-08-19T05:12:03.696+0000"),
  "Text" : "Ab quidem similique ad quo voluptatem corrupti. Blanditiis enim illo mol",
  "Comments" : [
    {
      "Created" : ISODate("2019-08-19T08:59:14.353+0000"),
      "Text" : "Consectetur voluptates ipsa enim optio sunt ut repudiandae moles",
      "UserId" : ObjectId("5d5b2463dc9e6f16530597e7")
    },
    {
      "Created" : ISODate("2019-08-19T11:40:01.146+0000"),
      "Text" : "Minus vitae dolores et quas sunt veritatis alias nihil non. Est",
      "UserId" : ObjectId("5d5b2463dc9e6f16530597e7")
    }
  ]
}
```



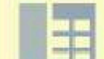


Объект пользователь

```
{
  "_id" : ObjectId("5d5b2463dc9e6f16530598b1"),
  "Name" : "Валентина Зуева 2",
  "Created" : ISODate("2019-08-19T08:31:32.990+0000"),
  "Url" : "https://мария.рф",
  "Articles" : [
    ObjectId("5d5b2463dc9e6f165305990c"),
    ObjectId("5d5b2463dc9e6f165305990d"),
    ObjectId("5d5b2463dc9e6f165305990e"),
    ObjectId("5d5b2463dc9e6f165305990f"),
    ObjectId("5d5b2463dc9e6f1653059910"),
    ObjectId("5d5b2463dc9e6f1653059911"),
    ObjectId("5d5b2463dc9e6f1653059912"),
    ObjectId("5d5b2463dc9e6f1653059913"),
    ObjectId("5d5b2463dc9e6f1653059914"),
    ObjectId("5d5b2463dc9e6f1653059915")
  ]
}
```








СЦЕНАРИЙ 1 ВСТРОЕННЫЕ ССЫЛКИ

75

Объект пользователь и статьи

users	
 id	bigint
 name	varchar(50)
 created	date
 url	varchar(255)

userid:id

articles_comments	
 id	bigint
 name	varchar(50)
 url	varchar(255)
 created	date
 text	varchar(1024)
 userid	bigint
 comments	jsonb

Запросы MongoDB

1. получить по имени id пользователя 2 операции
2. отфильтровать статьи по комментариям и убрать лишние комментарии $1 + N$ операций

Запросы Postgresql, 2 сценария

отфильтровать статьи по комментариям и убрать лишние комментарии + join на пользователей

1 index seek + 1 index seek + N операций

1. получить по имени id пользователя 2 операции
2. отфильтровать статьи по комментариям и убрать лишние комментарии 1 + N операций

ИТОГИ

Method	Mean	Error	StdDev
MongoSelectUserByUserName	843.2 us	14.45 us	21.18 us
PostgresSelectUserByUserName	3,136.4 us	216.99 us	311.20 us
MongoSelectCommentsByUserName	11,112.4 us	756.00 us	1,108.14 us
PostgresSelectCommentsByUserName	462,410.4 us	17,378.27 us	24,923.40 us
PostgresSelectCommentsByUserName2	79,333.7 us	11,560.26 us	17,302.86 us

Сценарий: 3 типа документа: статья и пользователь и комментарий

- комментарии расположены **отдельно** от статьи
- ссылка на пользователя есть в статье и комментарии
- 1к пользователей
- всего 10к статей
- всего примерно 50к комментариев

Сценарий: 3 типа документа: статья и пользователь и комментарий

- комментарии расположены отдельно от статьи
- ссылка на пользователя есть в статье и комментарии
- 1к пользователей
- всего 10к статей
- всего примерно 50к комментариев

Задача: найти все комментарии пользователя по его имени с данными статьи

Объект статья

```
{
  "_id" : ObjectId("5d5fa9df839b3279e5d8fe02"),
  "UserId" : "5d5fa9df839b3279e5d8fdf9",
  "Name" : "autem-autem-expedita",
  "Url" : "https://александр.org",
  "Created" : ISODate("2019-08-22T18:10:01.216+0000"),
  "Text" : "Dolore autem autem impedit. Necessitatibus qui impedi
  "Comments" : null
}
// -----
{
  "_id" : ObjectId("5d5fa9df839b3279e5d8fe01"),
  "UserId" : "5d5fa9df839b3279e5d8fdf9",
  "Name" : "ducimus-omnis-velit",
  "Url" : "http://ярослав.com",
  "Created" : ISODate("2019-08-22T18:33:59.738+0000"),
  "Text" : "Laboriosam neque neque consequatur est. Doloribus in
  "Comments" : null
}
```


Объект комментариев

```
{
  "_id" : ObjectId("5d5fa9de839b3279e5d8f880"),
  "Created" : ISODate("2019-08-22T13:59:36.265+0000"),
  "Text" : "Consequuntur necessitatibus et facilis tempora placeat.\nDolores fugiat quidem",
  "ArticleId" : "5d5fa9de839b3279e5d8f859",
  "UserId" : "5d5fa9db839b3279e5d8d97c"
}
// -----
{
  "_id" : ObjectId("5d5fa9de839b3279e5d8f87f"),
  "Created" : ISODate("2019-08-23T04:25:00.075+0000"),
  "Text" : "Dolores quia magnam reiciendis molestias.\nExcepturi non repellendus velit.",
  "ArticleId" : "5d5fa9de839b3279e5d8f859",
  "UserId" : "5d5fa9d9839b3279e5d8bd64"
}
```

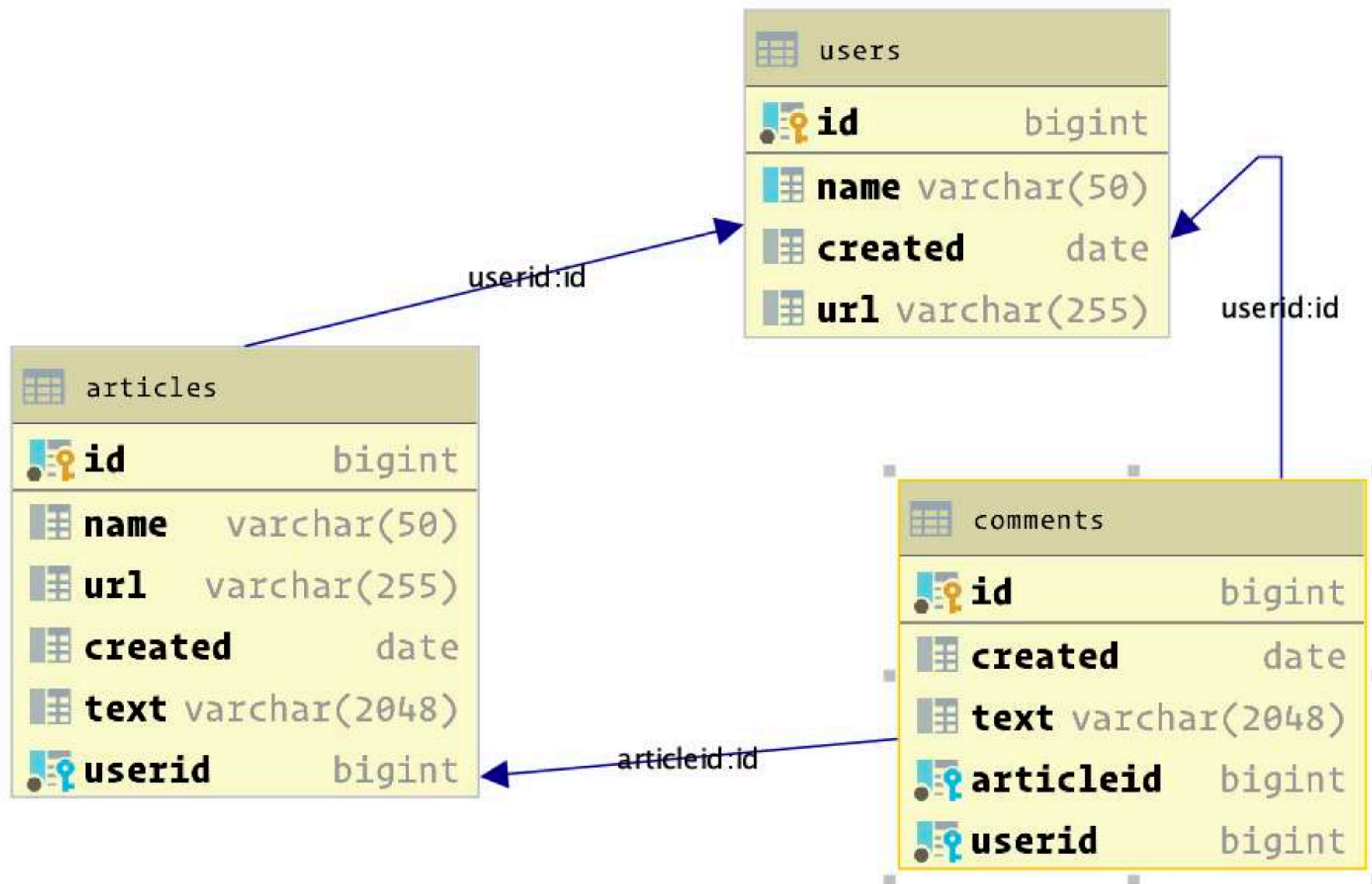

Объект пользователь

```
{
  "_id" : ObjectId("5d5fa9df839b3279e5d8fdc2"),
  "Name" : "Даниил Бирюков 992",
  "Created" : ISODate("2019-08-23T05:02:47.959+0000"),
  "Url" : "https://георгий.com",
  "Articles" : null
}
// -----
{
  "_id" : ObjectId("5d5fa9df839b3279e5d8fd91"),
  "Name" : "Денис Пономарев 991",
  "Created" : ISODate("2019-08-23T00:52:36.744+0000"),
  "Url" : "http://алина.рф",
  "Articles" : null
}
```


СЦЕНАРИЙ 2 ОТДЕЛЬНЫЕ ДОКУМЕНТЫ

85

Объект пользователь



Запросы MongoDB

1. получить по имени id пользователя 2 операции
2. получить по id пользователя все комментарии 2 + N операций
3. получить по каждому комментарию статью N операций

Запросы SQL

```
select a.name, a.url, c.text, c.created  
from comments as c  
      join users u on c.userid = u.id  
      join articles a on c.articleid = a.id  
where u.name = 'Лука Костин 976';
```

1 index user seek + 1 index comments seek + 1 index article seek
+ 1 HASH JOIN + N операций

ИТОГИ

Method	Mean	Error	StdDev
MongoSelectCommentsByUserName	11.160 ms	0.8713 ms	1.2772 ms
PostgresSelectCommentsByUserName	4.796 ms	0.2024 ms	0.2967 ms

NOSQL

ПРИМЕР

SERVICE PIPE



```
{
  "_id" : ObjectId("5b4ee8335c005013739f1944"),
  "userId" : ObjectId("5b4e58f65c005013739f1907"),
  "updated" : ISODate("2018-07-01T15:00:00.000+0000"),
  "url" : "http://service-name.ru/api/v1/add/",
  "pipe" : [
    {
      "name" : "auth",
      "isLogged" : 1.0,
      "type" : "oAuth"
    }
  ]
}
```

SERVICE PIPE



```
"pipe" : [  
  {  
    "name" : "auth",  
    "isLogged" : 1.0,  
    "type" : "oAuth"  
  },  
  {  
    "name" : "validate",  
    "isValid" : 1.0,  
    "validateClass" : "...",  
    "nextService" : "notification"  
  }  
]
```


SERVICE PIPE



```
"pipe" : [  
  {  
    "name" : "auth",  
    "isLogged" : 1.0,  
    "type" : "oAuth"  
  },  
  {  
    "name" : "validate",  
    "isValid" : 1.0,  
    "validateClass" : "...",  
    "nextService" : "notification"  
  },  
  {  
    "name" : "notification",  
    "email" : "user@user.ru",  
    "date" : ISODate("2018-07-01T16:00:00.000+0000"),  
    "nextService" : "billing"  
  }  
]
```

SERVICE PIPE



```
"pipe" : [  
  {  
    "name" : "auth",  
    "isLogged" : 1.0,  
    "type" : "oAuth"  
  },  
  {  
    "name" : "validate",  
    "isValid" : 1.0,  
    "validateClass" : "...",  
    "nextService" : "notification"  
  },  
  {  
    "name" : "notification",  
    "email" : "user@user.ru",  
    "date" : ISODate("2018-07-01T16:00:00.000+0000"),  
    "nextService" : "billing"  
  },  
  {  
    "name" : "billing",  
    "email" : "user@user.ru",  
    "billingDate" : ISODate("2018-07-01T16:10:00.000+0000"),  
    "accountId" : ObjectId("5b4e58f65c005013739f1907")  
  }  
]
```

КАК ЭТО ХРАНИТЬ
В РЕЛЯЦИОННОЙ БД?

NO GOD PLEASE

NOOOOOOOO

- денормализованные таблицы с **n колонками**

- денормализованные таблицы с **n колонками**
- таблица **на каждый тип** записи

- денормализованные таблицы с **n колонками**
- таблица **на каждый тип** записи
- таблица с общими данными и добавкой в **json**

- денормализованные таблицы с **n колонками**
- таблица **на каждый тип** записи
- таблица с общими данными и добавкой в **json**
- key-value pairs

СПРАВНЕНИЕ

Сценарий: 1 тип документа: пакет












- данные пакета встроены в пакет
- для sql данные лежат в денормализованной таблице
- 100к пакетов
- всего примерно 500к данных

Сценарий: 1 тип документа: пакет

- данные пакета встроены в пакет
- для sql данные лежат в денормализованной таблице
- 100к пакетов
- всего примерно 500к данных

Задача: найти все пакеты где есть нужные данные

```
{
  "_id" : ObjectId("5d60600929dc28fce46a5a1"),
  "Created" : ISODate("2019-08-23T04:17:10.510+0000"),
  "InitSystem" : "Бобров Трейд",
  "Records" : [
    {
      "_t" : "AuthService",
      "Name" : "AuthService",
      "Created" : ISODate("2019-08-23T07:40:51.565+0000"),
      "IsAuth" : false,
      "UserName" : "Valentin8@ya.ru"
    },
    {
      "_t" : "BillingService",
      "Name" : "BillingService",
      "Created" : ISODate("2019-08-23T10:43:11.926+0000"),
      "IsBilled" : false,
      "Provider" : "Калинин, Терентьева and Куликов"
    },
    {
      "_t" : "PartnerService",
      "Name" : "PartnerService",
      "Created" : ISODate("2019-08-23T07:38:48.869+0000"),
      "IsDone" : false,
      "Partner" : "Елисеева, Воронцов and Беляков"
    }
  ]
}
```


packages	
 id	bigint
 created	date
 init_system	varchar(30)
 record_name	varchar(30)
 record_created	date
 record_is_auth	boolean
 record_username	varchar(30)
 record_is_billed	boolean
 record_provider	boolean
 record_is_done	boolean
 record_partner	varchar(30)

Method	Mean	Error	StdDev
MongoSelectAuthAndBilledRecords	43.67 ms	2.113 ms	3.435 ms
PostgresSelectAuthAndBilledRecords	65.12 ms	6.453 ms	15.657 ms
SqlSelectAuthAndBilledRecords	41.40 ms	1.396 ms	2.945 ms

ИТОГИ

MongoDB

Иерархическая связанная
структура

Отлично на чтение
под запросы

Отлично на запись
вместе

Плохо на обновление

RelationDB

Нормально на чтение

Хорошо на запись

Отлично на обновление

MongoDB

Иерархическая связанная структура

Отлично на чтение
под запросы

Отлично на запись
вместе

Плохо на обновление

Сырые данные

Отлично на чтение
Отлично на запись

RelationDB

Нормально на чтение

Хорошо на запись

Отлично на обновление

Отлично на чтение
Неудобно на запись

MongoDB

Иерархическая связанная структура

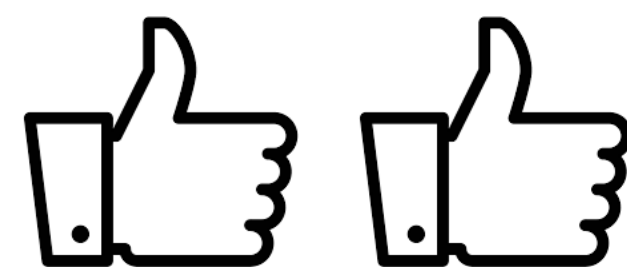
Отлично на чтение
под запросы

Отлично на запись
вместе

Плохо на обновление

Сырые данные

Отлично на чтение
Отлично на запись



RelationDB

Нормально на чтение

Хорошо на запись

Отлично на обновление

Отлично на чтение
Неудобно на запись



ИНСТРУМЕНТЫ

Mongodb

- ODM

Mongodb

- ODM

RelationDB

- ADO.NET
- Dapper
- Entity Framework

Mongodb

- ODM
 - Model mapping
 - Raw Query Language
 - Driver API
 - LINQ Query Language

СИНТАКСИС

ВСТАВКА ДАННЫХ




```
using var conn = new NpgsqlConnection(connectionString);  
  
conn.Open();  
conn.TypeMapper.UseJsonNet();  
  
using var transaction = conn.BeginTransaction();
```

```
using var cmd = new NpgsqlCommand() {  
    Connection = conn,  
    CommandText =  
        $"INSERT INTO articles (name, text, user, comments) VALUES (@n,@t,@u,@co)"  
};
```

```
cmd.Parameters.AddWithValue("c", document.Name);
cmd.Parameters.AddWithValue("n", document.Text);
cmd.Parameters.AddWithValue("t", document.User);

cmd.Parameters.Add(new NpgsqlParameter("i", NpgsqlDbType.Jsonb) {
    Value = document.Comments
});

.
.
.

cmd.ExecuteNonQuery();

transaction.Save("transaction");
```

```
using var context = new BlogContext();
using var transaction = context.Database.BeginTransaction();

try {
    var record = new RecordFactory().GetRecord();

    context.Blog.Add(record);
    context.SaveChanges();

    transaction.Commit();
}
catch () {
    transaction.Rollback();
}
```



```
using var context = new BlogContext();
using var transaction = context.Database.BeginTransaction();

try {
    var record = new RecordFactory().GetRecord();

    context.Blog.Add(record);
    context.SaveChanges();

    transaction.Commit();
}
catch () {
    transaction.Rollback();
}
```

```
using var context = new BlogContext();
using var transaction = context.Database.BeginTransaction();

try {
    var record = new RecordFactory().GetRecord();

    context.Blog.Add(record);
    context.SaveChanges();

    transaction.Commit();
}
catch () {
    transaction.Rollback();
}
```

Но **ТОЛЬКО** для полей в **SQL Server**
Json VARCHAR

```
var client = new MongoClient(connectionString);  
var collection = client  
    .GetDatabase("test")  
    .GetCollection<Article>("articles");
```

```
using var session = client.StartSession();  
session.StartTransaction();
```

```
try {  
    collection.InsertMany(documents);  
    session.CommitTransaction();  
}  
catch () {  
    session.RollbackTransaction();  
}
```



```
var client = new MongoClient(connectionString);  
var collection = client  
    .GetDatabase("test")  
    .GetCollection<Article>("articles");
```

```
using var session = client.StartSession();  
session.StartTransaction();
```

```
try {  
    collection.InsertMany(documents);  
    session.CommitTransaction();  
}  
catch () {  
    session.RollbackTransaction();  
}
```

```
var client = new MongoClient(connectionString);  
var collection = client  
    .GetDatabase("test")  
    .GetCollection<Article>("articles");
```

```
using var session = client.StartSession();  
session.StartTransaction();
```

```
try {  
    collection.InsertMany(documents);  
    session.CommitTransaction();  
}  
catch () {  
    session.RollbackTransaction();  
}
```

СИНТАКСИС ПОЛУЧЕНИЕ ДАННЫХ

```
using var conn = new NpgsqlConnection(connectionString);  
conn.Open();
```

```
using var cmd = new NpgsqlCommand(  
    "select t.id, t.name, t.url, t.v  
    from (  
        select id, name, url, userid,  
            jsonb_array_elements(comments)::jsonb as v  
        from articles_comments) as t  
    join users as u on u.id = jsonb_extract_path_text(v, 'UserId')::int  
    where u.name = @name"  
    , conn));
```

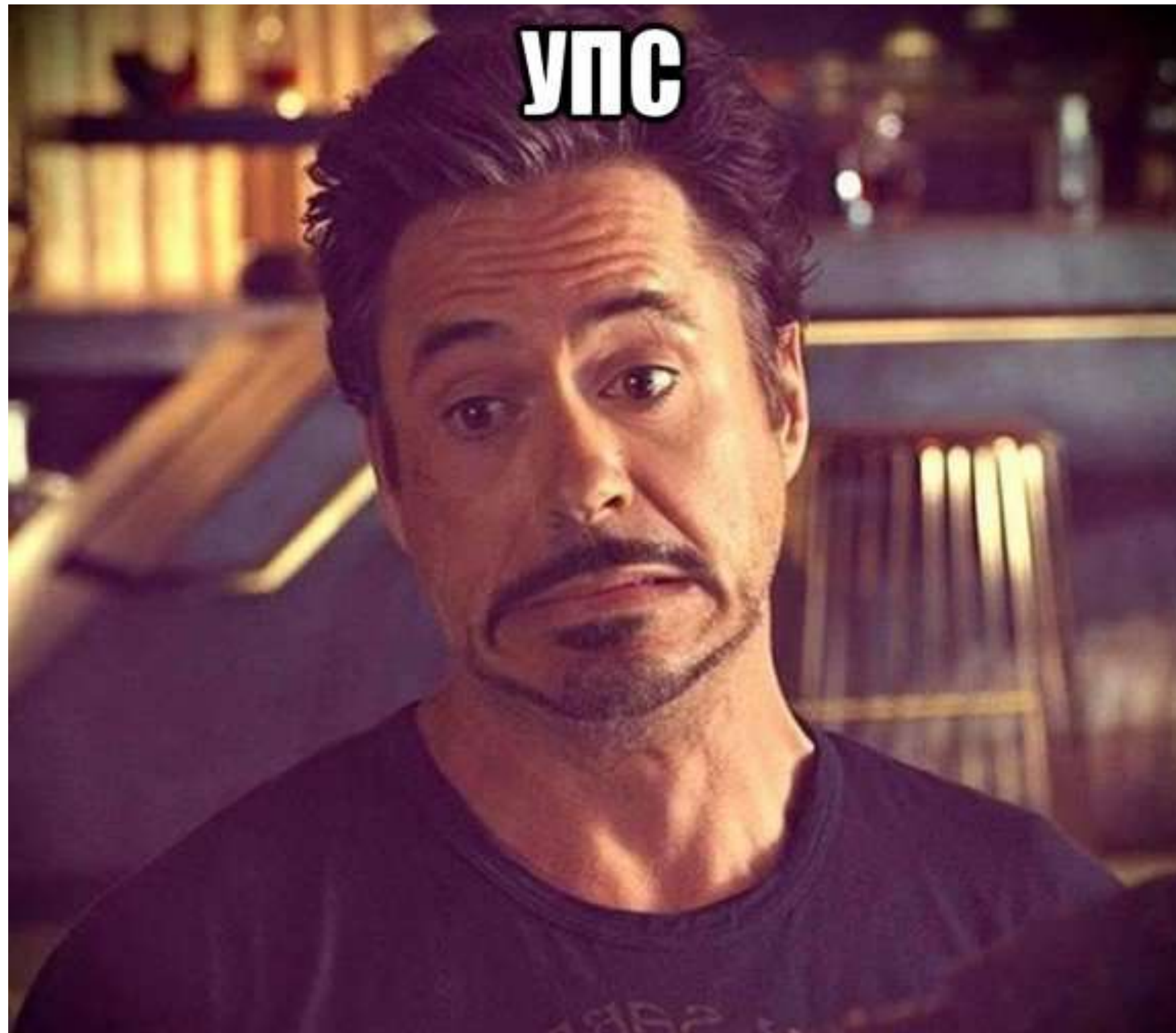


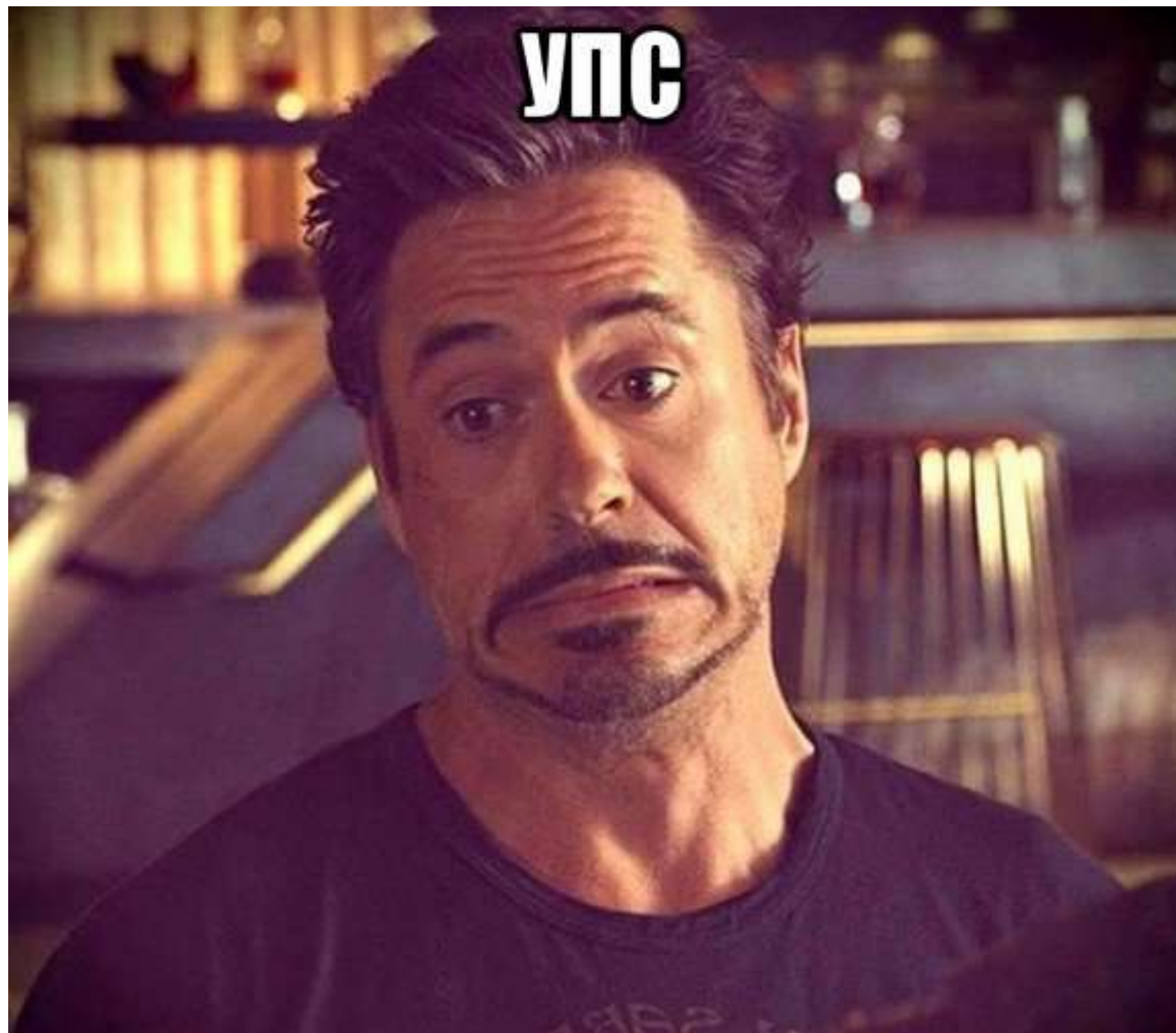
```
cmd.Parameters.AddWithValue("@name", _faker.PickRandom(_userNames));

using var reader = cmd.ExecuteReader();

var articles = reader.Select(r => new ArticleSql
{
    Id = r.GetInt64(0),
    Name = r.GetString(1),
    Url = r.GetString(2),
    Comments = JsonConvert.DeserializeObject<List<Comment>>(
        "[" + r.GetString(3) + "]")
});
```







[Npgsql.EntityFrameworkCore.PostgreSQL/issues/981](https://github.com/Npgsql.EntityFrameworkCore.PostgreSQL/issues/981)

This *does not* include:

- Calling JSON functions and operators
- Mapping to non-POCO, JSON DOM types
- PostgreSQL 12 JSONPATH support


```
var userFilter = Builders<User>.Filter.Eq(
    u => u.Name, _faker.PickRandom(_userNames));

var user = _users.Find(userFilter).FirstOrDefault();

var articlesFilter = Builders<Article>.Filter
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

var projection = Builders<Article>.Projection
    .Include(a => a.Name)
    .Include(a => a.Url)
    .Include(a => a.Comments)
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

articles.Find(articlesFilter).Project(projection).ToList();
```

```
var userFilter = Builders<User>.Filter.Eq(
    u => u.Name, _faker.PickRandom(_userNames));

var user = _users.Find(userFilter).FirstOrDefault();

var articlesFilter = Builders<Article>.Filter
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

var projection = Builders<Article>.Projection
    .Include(a => a.Name)
    .Include(a => a.Url)
    .Include(a => a.Comments)
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

articles.Find(articlesFilter).Project(projection).ToList();
```

```
var userFilter = Builders<User>.Filter.Eq(
    u => u.Name, _faker.PickRandom(_userNames));

var user = _users.Find(userFilter).FirstOrDefault();

var articlesFilter = Builders<Article>.Filter
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

var projection = Builders<Article>.Projection
    .Include(a => a.Name)
    .Include(a => a.Url)
    .Include(a => a.Comments)
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);

articles.Find(articlesFilter).Project(projection).ToList();
```

```
var userFilter = Builders<User>.Filter.Eq(  
    u => u.Name, _faker.PickRandom(_userNames));  
  
var user = _users.Find(userFilter).FirstOrDefault();  
  
var articlesFilter = Builders<Article>.Filter  
    .ElemMatch(a => a.Comments, comment => comment.UserId == user.Id);  
  
var projection = Builders<Article>.Projection  
    .Include(a => a.Name)  
    .Include(a => a.Url)  
    .Include(a => a.Comments)  
    .ElemMatch(a => a.Comments, comment => comment.UserId =  
  
articles.Find(articlesFilter).Project(projection).ToList();
```



ИТОГИ

MongoDB

Добавление
данных

Просто и удобно

Надо привыкнуть к
синтаксису

Нет хинтов

RelationDB

Просто через EF на запись
ТОЛЬКО реляционных
данных

ADO.NET

Есть хинты

MongoDB

RelationDB

Добавление
данных

Просто и удобно

Просто через EF на запись
ТОЛЬКО реляционных
данных

Надо привыкнуть к
синтаксису

ADO.NET

Нет хинтов

Есть хинты

Чтение данных

Просто и удобно
Есть LINQ

Просто через EF без JSON
Есть LINQ

MongoDB

Добавление
данных

Просто и удобно

Надо привыкнуть к
синтаксису

Нет хинтов

Чтение данных

Просто и удобно
Есть LINQ



RelationDB

Просто через EF на запись
ТОЛЬКО реляционных
данных

ADO.NET

Есть хинты

Просто через EF без JSON
Есть LINQ



ИТОГИ ИТОГОВ

MongoDB



RelationDB



TOOLTIP

пробуйте новое и процветайте

Q & A