Yandex

2016 CEE-SEC(R) Software Engineering Conference in Russia

YoctoDB @ Yandex.Classifieds

Vadim Tsesko
incubos@

About

Backend infrastructure team

- Services
- Libraries
- Frameworks

@ Yandex.Classifieds

- auto.ru
- auto.yandex.ru
- rabota.yandex.ru
- realty.yandex.ru
- travel.yandex.ru

https://stat.yandex.ru

Target audience

Target audience

› Software engineers / architects

Target audience

- Software engineers / architects
- Developing high-load systems

Target audience

- Software engineers / architects
- Developing high-load systems
- In JVM world (optionally)

Introduction

YoctoDB

- Embedded library in Java for
- horizontally partitioned
- immutable-after-construction databases

Yandex.Auto / Auto.ru

Yandex.Auto / Auto.ru

- 6M monthly / 0.5M daily audience

Yandex.Auto / Auto.ru

- 6M monthly / 0.5M daily audience
- 1M documents reindexed every minute

Yandex.Auto / Auto.ru

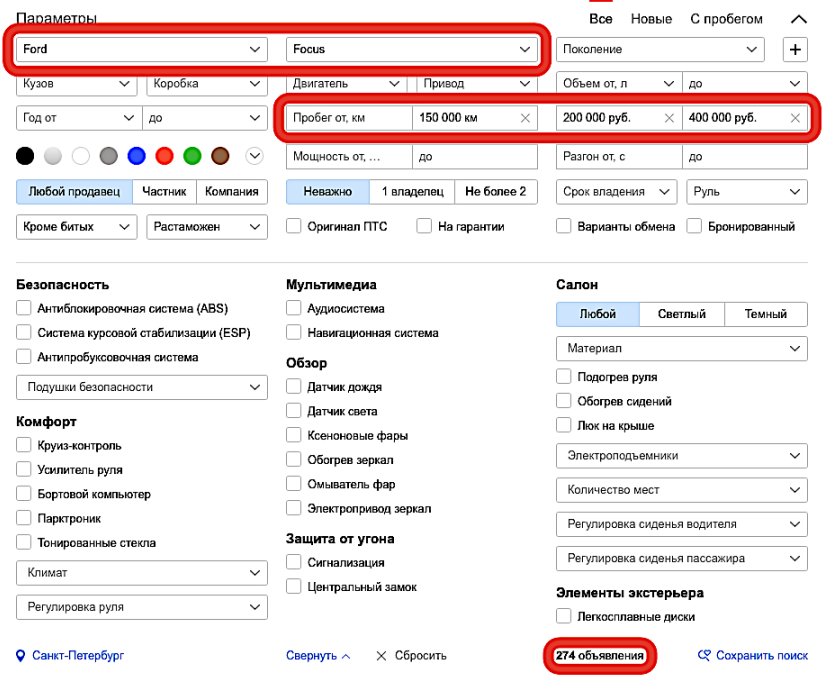
- 6M monthly / 0.5M daily audience
- 1M documents reindexed every minute
- >500 rps per node

Yandex.Auto/Auto.ru

- 6M monthly/0.5M daily audience
- 1M documents reindexed every minute
- >500 rps per node
- 99.9% < 200 ms

Yandex.Auto/Auto.ru

- 6M monthly/0.5M daily audience
- 1M documents reindexed every minute
- >500 rps per node
- 99.9% < 200 ms
- Rich search queries


Параметры
Все
Новые
С пробегом
Ford
Focus
Поколение
Кузов
Коробка
Двигатель
Привод
Объем от, л
до
Год от
до
Пробег от, км
150 000 км
200 000 руб.
400 000 руб.
Мощность от, ...
до
Разгон от, с
до
Любой продавец
Частник
Компания
Неважно
1 владелец
Не более 2
Срок владения
Руль
Кроме битых
Растаможен
Оригинал ПТС
На гарантии
Варианты обмена
Бронированный
Безопасность
Антиблокировочная система (ABS)
Система курсовой стабилизации (ESP)
Антипробуксовочная система
Подушки безопасности
Комфорт
Круиз-контроль
Усилитель руля
Бортовой компьютер
Парктроник
Тонированные стекла
Климат
Регулировка руля
Мультимедиа
Аудиосистема
Навигационная система
Обзор
Датчик дождя
Датчик света
Ксеноновые фары
Обогрев зеркал
Омыватель фар
Электропривод зеркал
Защита от угона
Сигнализация
Центральный замок
Салон
Любой
Светлый
Темный
Материал
Подогрев руля
Обогрев сидений
Люк на крыше
Электроподъемники
Количество мест
Регулировка сиденья водителя
Регулировка сиденья пассажира
Элементы экстерьера
Легкосплавные диски
Санкт-Петербург
Свернуть
Сбросить
274 объявления
Сохранить поиск

Ford Focus II Рестайлинг

1.6 AT (100 л.с.) бензин, передний привод, седан, серебристый

Санкт-Петербург

399 000 ₽ | 2008 | 116 000 км

RAISE

Ford Focus II Рестайлинг

1.6 MT (100 л.с.) бензин, передний привод, хэтчбек 5 дв., серый

Санкт-Петербург, 1 час назад

399 000 ₽ | 2011 | 106 750 км

FILTER

ЛЮКСАР
28 объявлений

Ford Focus II

2.0 MT (145 л.с.) бензин, передний привод, хэтчбек 5 дв., голубой

Санкт-Петербург, 3 часа назад

255 000 ₽ | 2007 | 131 000 км

JOIN WITH CATALOG

Ford Focus II Рестайлинг

1.6 MT (90 л.с.) дизель, передний привод, универсал 5 дв., чёрный

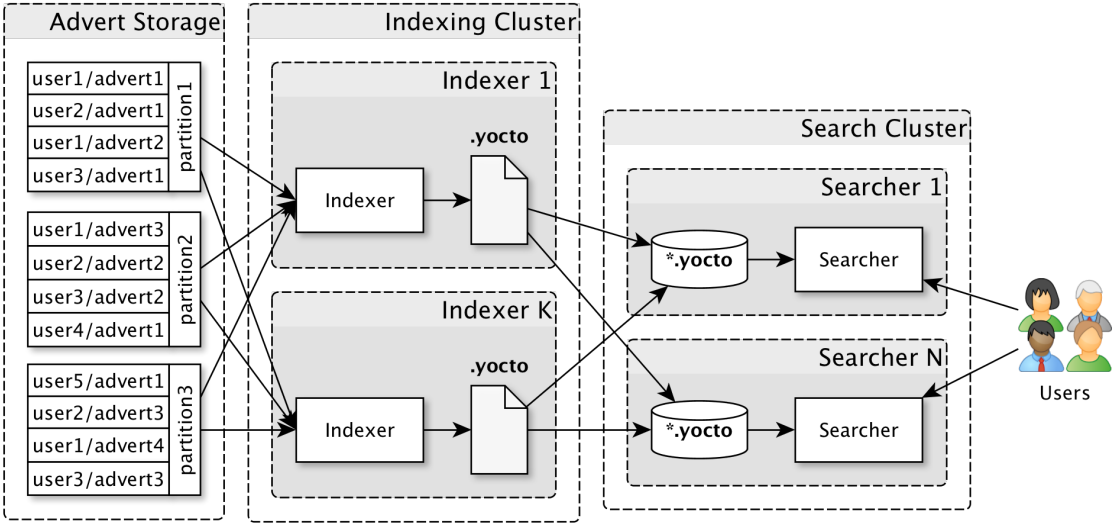
Санкт-Петербург, 3 часа назад

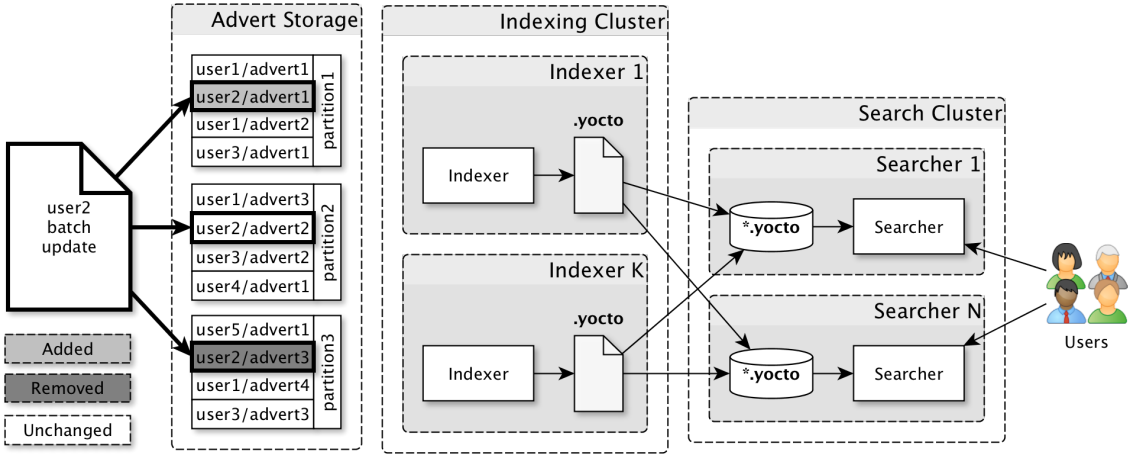
398 000 ₽ | 2009 | 92 000 км

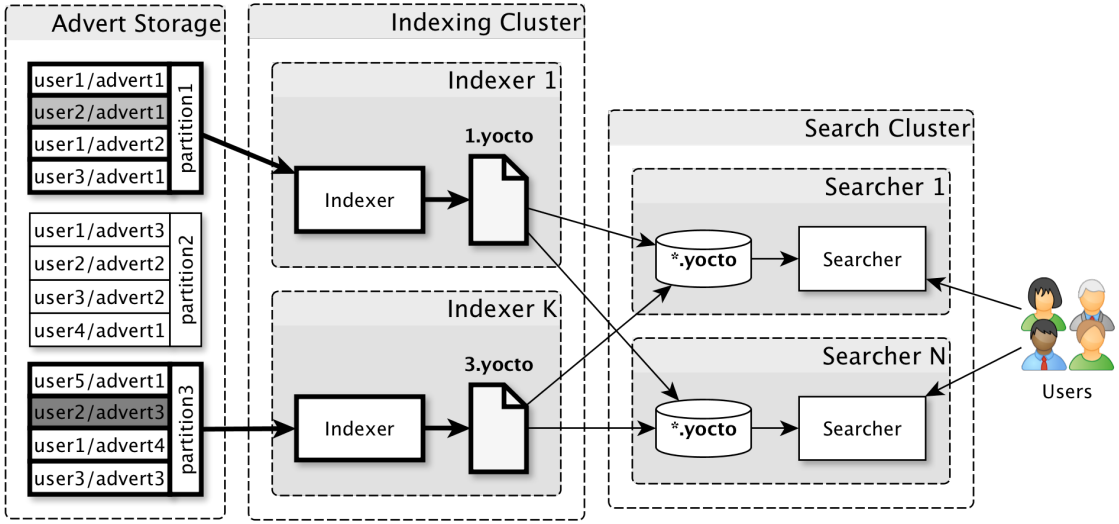
Автосалон
10 объявлений

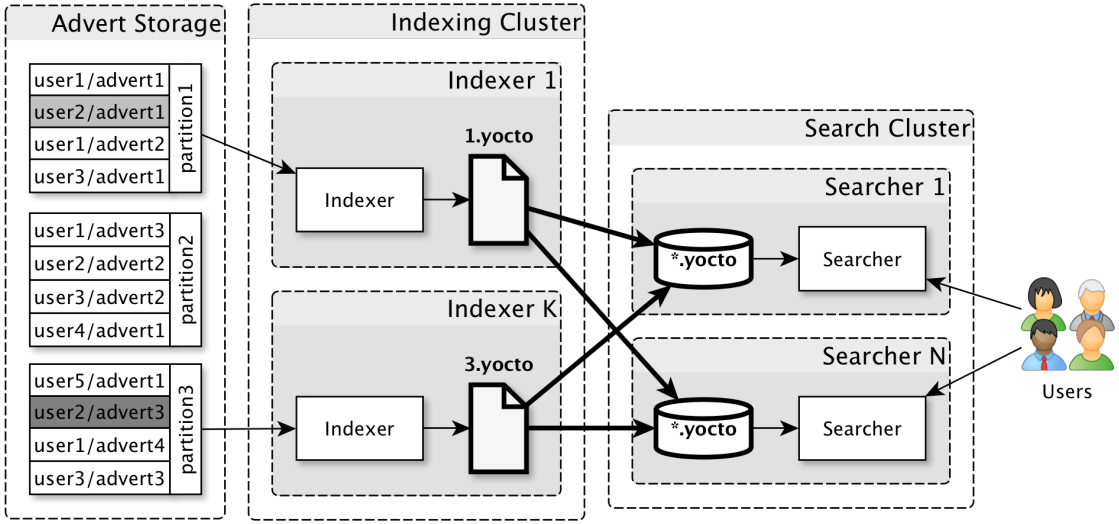
ORDER BY DATE

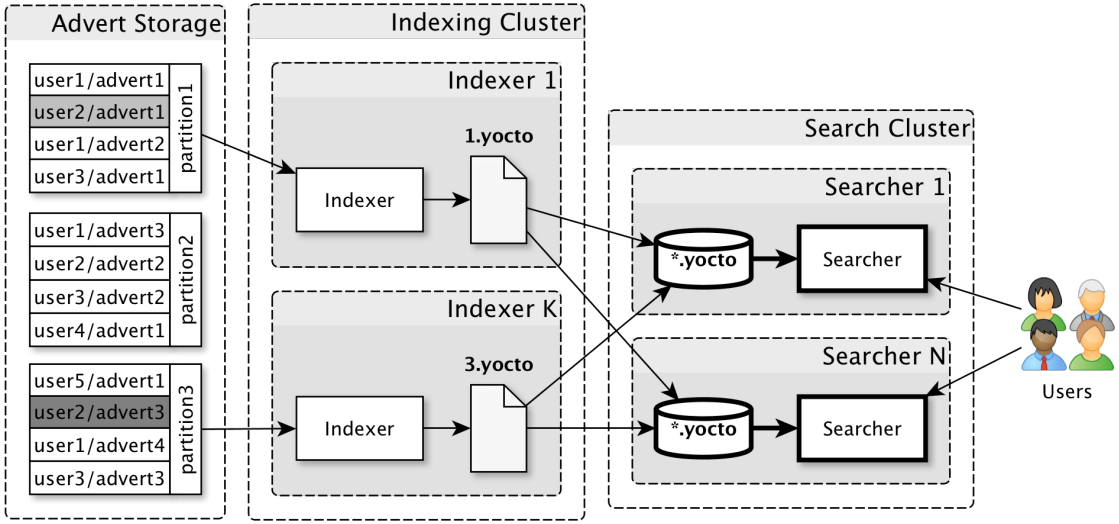
Motivation


Advert Storage
user1/advert1
user2/advert1
user1/advert2
user3/advert1
partition1
user1/advert3
user2/advert2
user3/advert2
user4/advert1
partition2
user5/advert1
user2/advert3
user1/advert4
user3/advert3
partition3
Indexing Cluster
Indexer 1
Indexer
.yocto
Indexer K
Indexer
.yocto
Search Cluster
Searcher 1
*.yocto
Searcher
Searcher N
*.yocto
Searcher
Users


Advert Storage
user1/advert1
user2/advert1
user1/advert2
user3/advert1
partition1
user1/advert3
user2/advert2
user3/advert2
user4/advert1
partition2
user5/advert1
user2/advert3
user1/advert4
user3/advert3
partition3
user2
batch
update
Added
Removed
Unchanged
Indexing Cluster
Indexer 1
.yocto
Indexer
Indexer K
.yocto
Indexer
Search Cluster
Searcher 1
*.yocto
Searcher
Searcher N
*.yocto
Searcher
Users

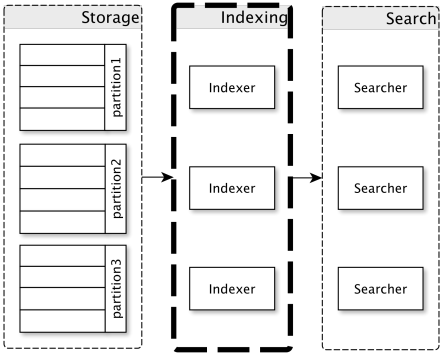

Advert Storage
user1/advert1
user2/advert1
user1/advert2
user3/advert1
partition1
user1/advert3
user2/advert2
user3/advert2
user4/advert1
partition2
user5/advert1
user2/advert3
user1/advert4
user3/advert3
partition3
Indexing Cluster
Indexer 1
Indexer
1.yocto
Indexer K
Indexer
3.yocto
Search Cluster
Searcher 1
*.yocto
Searcher
Searcher N
*.yocto
Searcher
Users


Advert Storage
user1/advert1
user2/advert1
user1/advert2
user3/advert1
partition1
user1/advert3
user2/advert2
user3/advert2
user4/advert1
partition2
user5/advert1
user2/advert3
user1/advert4
user3/advert3
partition3
Indexing Cluster
Indexer 1
Indexer
1.yocto
Indexer K
Indexer
3.yocto
Search Cluster
Searcher 1
*.yocto
Searcher
Searcher N
*.yocto
Searcher
Users


Advert Storage
user1/advert1
user2/advert1
user1/advert2
user3/advert1
partition1
user1/advert3
user2/advert2
user3/advert2
user4/advert1
partition2
user5/advert1
user2/advert3
user1/advert4
user3/advert3
partition3
Indexing Cluster
Indexer 1
1.yocto
Indexer
Indexer K
3.yocto
Indexer
Search Cluster
Searcher 1
*.yocto
Searcher
Searcher N
*.yocto
Searcher
Users

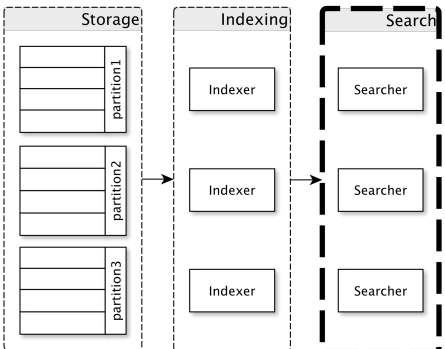
Indexing Cluster

- Stateless
- High throughput
- Horizontal scalability
- (Soft) availability



Search Cluster

- Low latency
- Quick index reopening
- High availability
- Self-sufficient



State of things

- 99% < 500 ms
- 99.9% < 1 s
- 10 nodes/datacenter
- Slow index reopening
- Periodic CMS tuning

State of things

- 99% < 500 ms
- 99.9% < 1 s
- 10 nodes/datacenter
- Slow index reopening
- Periodic CMS tuning

Goals

- 99% < 300 ms
- 99.9% < 500 ms
- 2 nodes/datacenter
- Fast index reopening
- No GC tuning

```
SELECT COUNT(id) FROM test WHERE f = ?
```

- `id` — PK
- `f` — indexed `md5(id % 10)`

```
SELECT COUNT(id) FROM test WHERE f = ?
```

› `id` — PK

› `f` — indexed `md5(id % 10)`

Table: Synthetic key-value performance (µs)

Database	Mean	Max	95%
SQLite	1930	2597	2186
H2	1858	2342	2001
Lucene	156	505	174

```
SELECT COUNT(id) FROM test WHERE f = ?
```

› `id` — PK

› `f` — indexed `md5(id % 10)`

Table: Synthetic key-value performance (µs)

Database	Mean	Max	95%
SQLite	1930	2597	2186
H2	1858	2342	2001
Lucene	156	505	174
YoctoDB (prototype)	30	162	47

Data Model

Composite Database

A collection of **simple databases.**

Composite Database

A collection of **simple databases.**

Simple Database

A collection of numbered **documents.**

Composite Database

A collection of **simple databases.**

Simple Database

A collection of numbered **documents.**

Document

Optional **payload** + a set of named **fields.**

Composite Database

A collection of **simple databases.**

Simple Database

A collection of numbered **documents.**

Document

Optional **payload** + a set of named **fields.**

Field

- **Filterable** (≥0 values per document)
- **Sortable** (exactly one value per document)
- **Full** (sortable + filterable)

Field Value

- Opaque comparable array of bytes
- Conversions from/to `String` and integers
  - Fixed length
  - Big endian
- Custom conversions supported

Example Document

- Payload — serialized advert (description, URLs, etc.)
- `region_id` — **filterable** integer (partition key)
- `mark` — **filterable** string
- `model` — **filterable** string
- `date` — **sortable** integer
- `price` — **full** integer

Example Queries

Example Queries

- Count all the adverts having
  - Specified `region_id` value

Example Queries

- Count all the adverts having
  - Specified `region_id` value
- Select top 10 adverts having
  - Specified `region_id` value
  - `mark` equal to `BMW`
  - `price` in range `[$10000, $20000]`
  - Order the adverts by `date` descending

Isomorphic SQL Schema

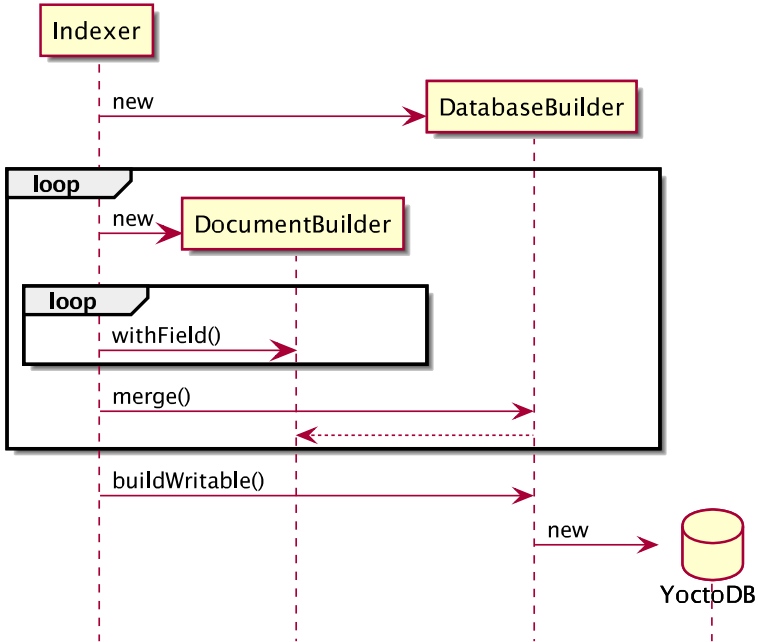
```
CREATE TABLE adverts(
  id INT AUTO_INCREMENT,
  payload BLOB,
  region_id INT,
  mark TEXT,
  model TEXT,
  date TIMESTAMP,
  price BIGINT);
CREATE INDEX region_id_idx ON adverts(region_id);
CREATE INDEX ...
```

Isomorphic SQL Queries

```
SELECT COUNT(*) FROM adverts
  WHERE region_id = :region_id;

SELECT payload FROM adverts
  WHERE
    region_id = :region_id AND
    mark = "BMW" AND
    price BETWEEN (1000000, 2000000)
  ORDER BY date DESC
  LIMIT 10;
```

API


Indexer
new
DatabaseBuilder
loop
new
DocumentBuilder
loop
withField()
merge()
buildWritable()
new
YoctoDB

Create Database (Java)

```
// Get current database format
final DatabaseFormat format =
    DatabaseFormat.getCurrent();

// Create a mutable database
final DatabaseBuilder dbBuilder =
    format.newDatabaseBuilder();
```

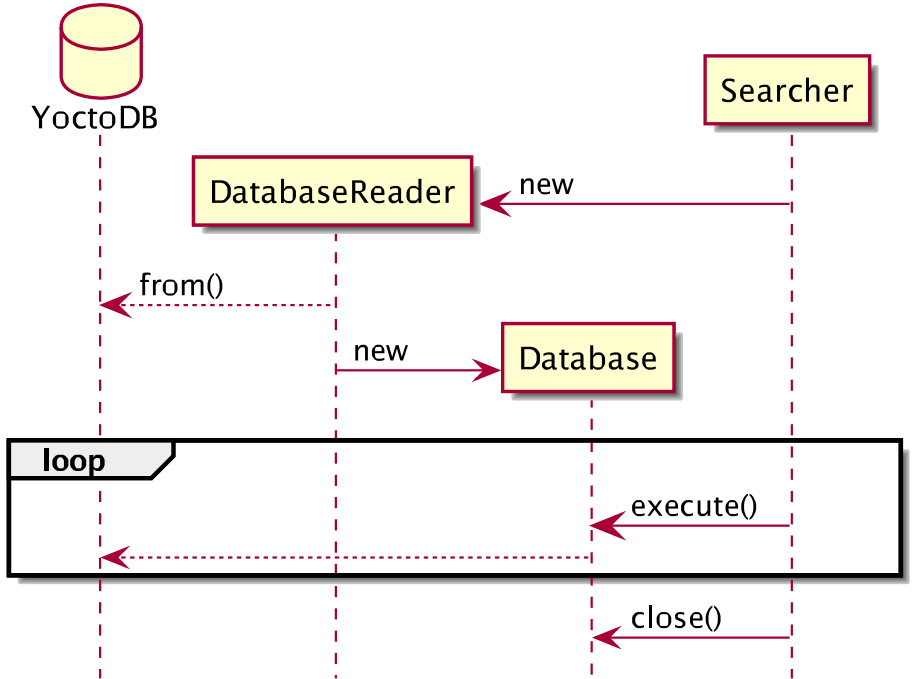
Add Documents (Java)

```
dbBuilder.merge(
  format.newDocumentBuilder()
    .withField("id", 1, FILTERABLE)
    .withField("score", 0, SORTABLE)
    .withPayload("payload1".getBytes()));
dbBuilder.merge(
  format.newDocumentBuilder()
    .withField("id", 2, FILTERABLE)
    .withField("score", 1, SORTABLE)
    .withPayload("payload2".getBytes()));
```

Serialize Database (Java)

```
final ByteArrayOutputStream os =
    new ByteArrayOutputStream();

dbBuilder.buildWritable().writeTo(os);
```


YoctoDB
Searcher
DatabaseReader
new
from()
new
Database
loop
execute()
close()

Open Database (Java)

```
final Database db =
  format.getDatabaseReader().from(
    Buffer.from(
      os.toByteArray()));
```

Query

```
final Query doc2 =
    select().where(eq("id", from(2)));

assertEquals(1, db.count(doc2));
```

```
final Query sorted = select()
    .where(and(gte("id", from(1)), lte("id", from(2))))
    .orderBy(desc("score"));
final List<Integer> ids = new LinkedList<Integer>();
db.execute(sorted, new DocumentProcessor() {
    @Override
    public boolean process(int document, ...) {
        ids.add(document);
        return true;
    }
});
assertEquals(asList(1, 0), ids);
```

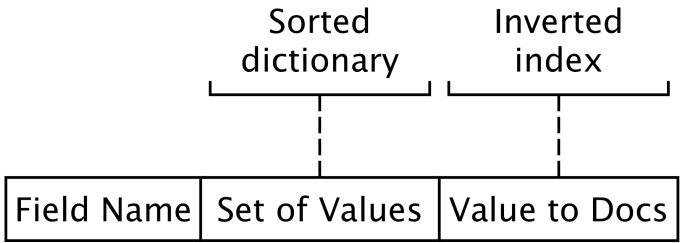
Immutable API

- `eq`, `gt`, `gte`, `lt`, `lte`, `in`, `between`
- `and`, `or`, `not`
- `orderBy` multiple fields in ascending/descending order
- `count`, `skip` and `limit`
- Apply custom callback to the filtered documents IDs in order
- Extract payload and sortable field value by document ID

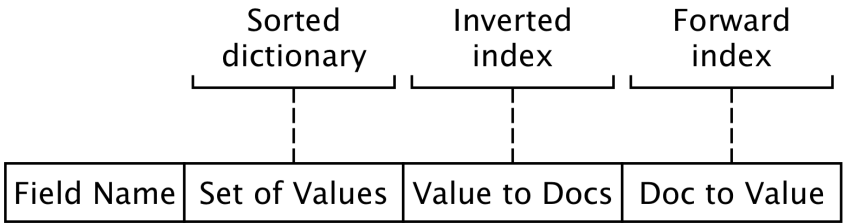
Implementation

Portable Binary Format

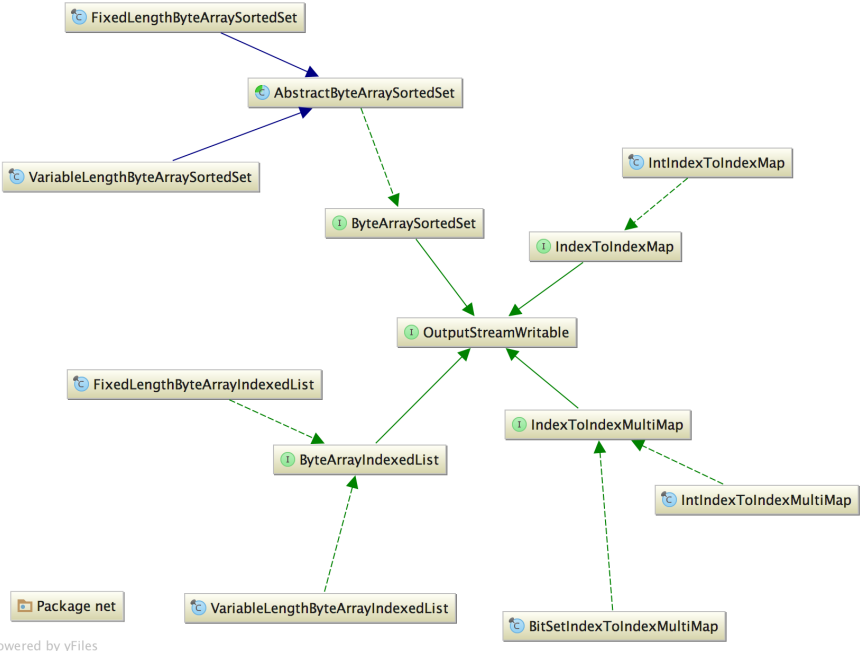
Filterable


Sorted dictionary
Inverted index
Field Name
Set of Values
Value to Docs
"mark"
["BMW", "FORD"]
{0: [0, 2], 1: [1, 3, 4]}

Sortable



"date"	[yesterday, today]	{0: [1, 4], 1: [0, 2, 3]	{0: 1, 1: 0, 2: 1, 3: 1, 4: 0}


FixedLengthByteArraySortedSet
AbstractByteArraySortedSet
VariableLengthByteArraySortedSet
IntIndexToIndexMap
ByteArraySortedSet
IndexToIndexMap
OutputStreamWritable
FixedLengthByteArrayIndexedList
IndexToIndexMultiMap
ByteArrayIndexedList
IntIndexToIndexMultiMap
Package net
VariableLengthByteArrayIndexedList
BitSetIndexToIndexMultiMap
Powered by yFiles

Persistent Collections

› YoctoDB core

Persistent Collections

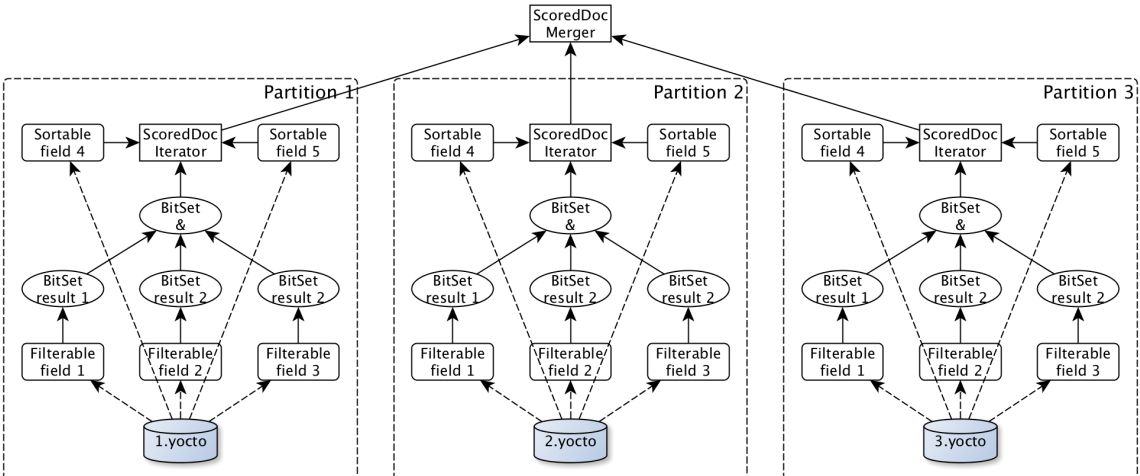
- YoctoDB core
- Separate mutable/immutable class hierarchies

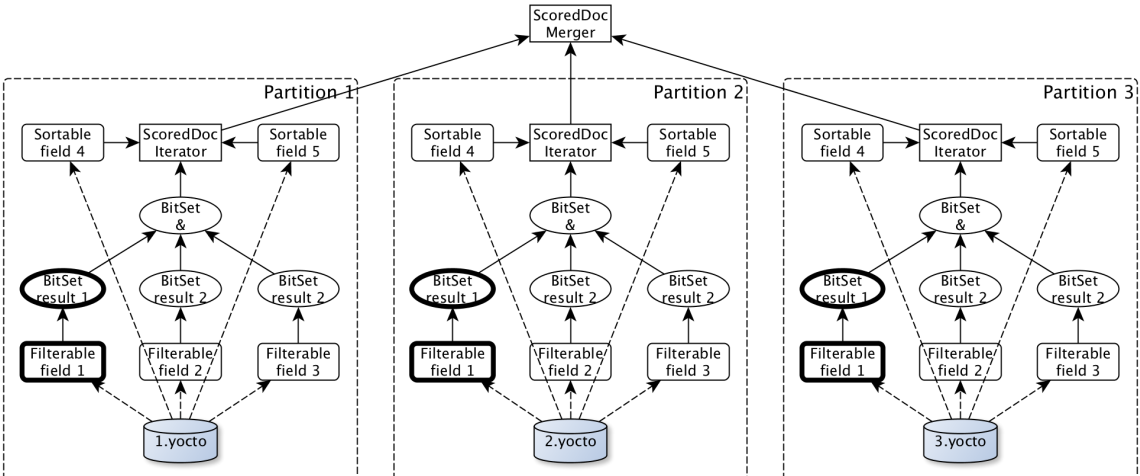
Persistent Collections

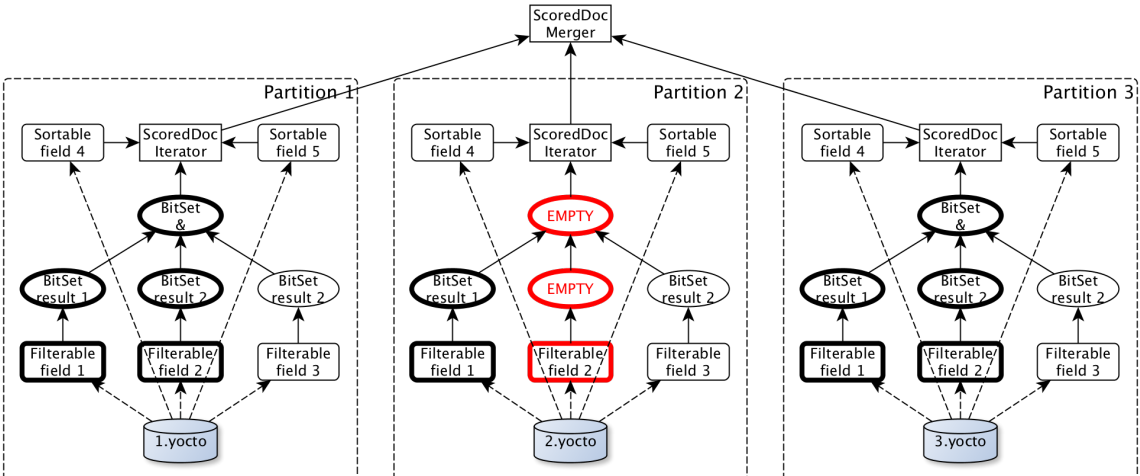
- YoctoDB core
- Separate mutable/immutable class hierarchies
- Various implementations

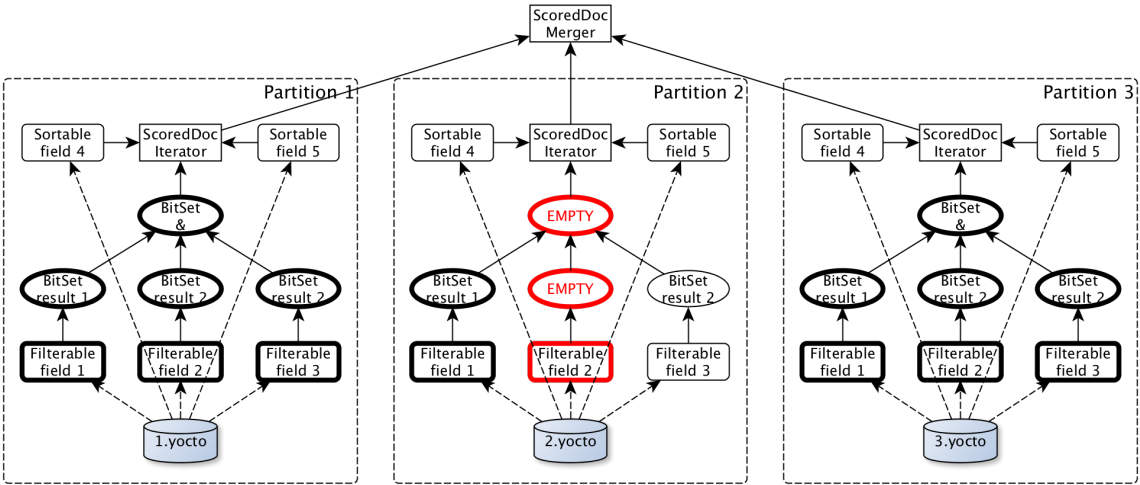
Persistent Collections

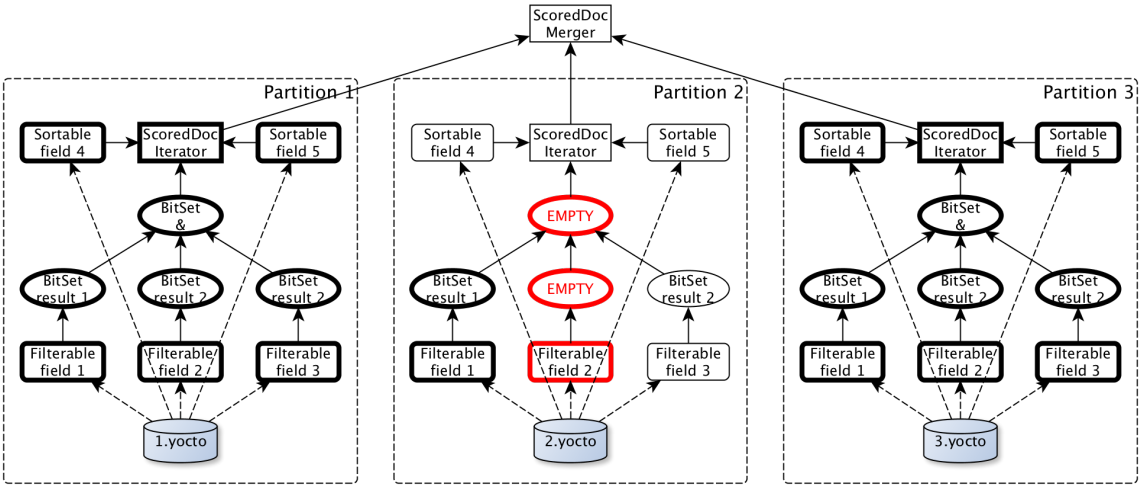
- YoctoDB core
- Separate mutable/immutable class hierarchies
- Various implementations
- Implementation choice during serializing

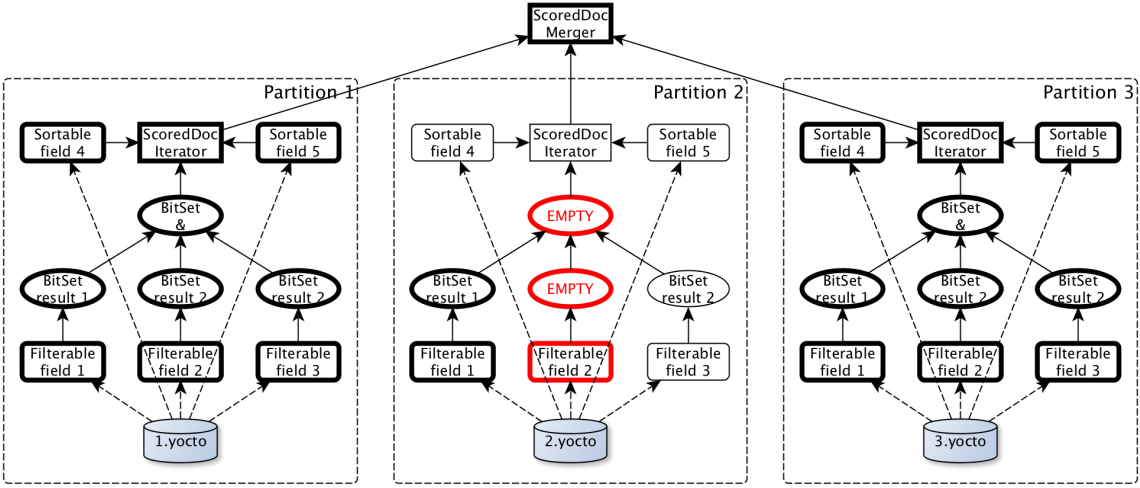

ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto


ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto


ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
EMPTY
EMPTY
BitSet result 1
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto


ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
EMPTY
EMPTY
BitSet result 1
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto


ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
EMPTY
EMPTY
BitSet result 1
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto


ScoredDoc Merger
Partition 1
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
1.yocto
Partition 2
Sortable field 4
ScoredDoc Iterator
Sortable field 5
EMPTY
BitSet result 1
EMPTY
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
2.yocto
Partition 3
Sortable field 4
ScoredDoc Iterator
Sortable field 5
BitSet &
BitSet result 1
BitSet result 2
BitSet result 2
Filterable field 1
Filterable field 2
Filterable field 3
3.yocto

Real Applications

Real Applications

Yandex.Auto

Dataset

- 3M documents in 1024 partitions on SSD
- 1.8 GB raw document payload in Protobuf
- 3.4 GB database in Lucene 3.x
- 2.3 GB database in YoctoDB
- Early YoctoDB release
- Real-world queries

Workload

Percentile	Lucene	YoctoDB
98%	300	150
95%	200	100
75%	100	45
mean	50	28

Table: 50 rps latency (ms)

Workload

Percentile	Lucene	YoctoDB
98%	300	150
95%	200	100
75%	100	45
mean	50	28

Table: 50 rps latency (ms)

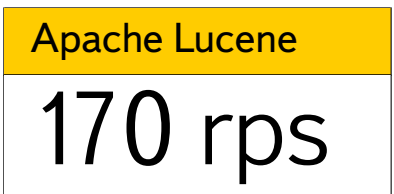
YoctoDB

- 2x lower percentiles
- 2x lower and much steadier CPU load

Extreme Load

Still conforming to SLO.

Extreme Load



Still conforming to SLO.

Extreme Load

Still conforming to SLO.

Real Applications

Auto.ru

Dataset

- 0.5 M documents in 1024 partitions on SSD
- 3-4 GB database
- Index fully rebuilt every 3-5 minutes
- Document — Protobuf payload + almost a hundred filterable/sortable fields

Workload

Table: 500 rps latency (ms)

Metric	50%	75%	90%	95%	99%	99.9%
Value	6	20	40	50	70	200

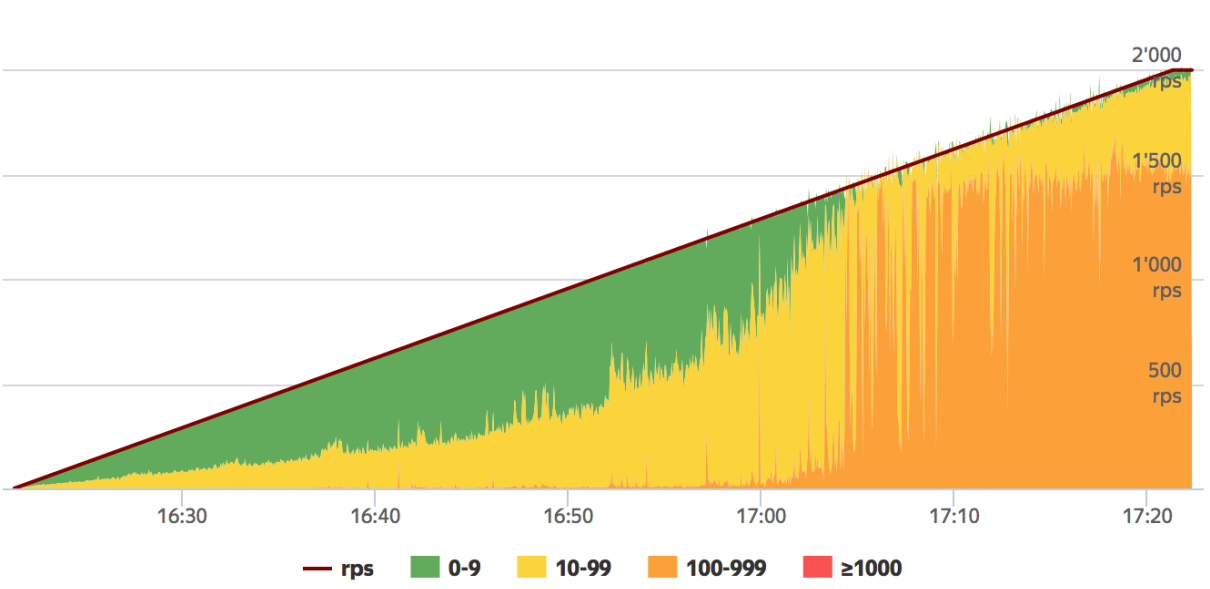
Workload

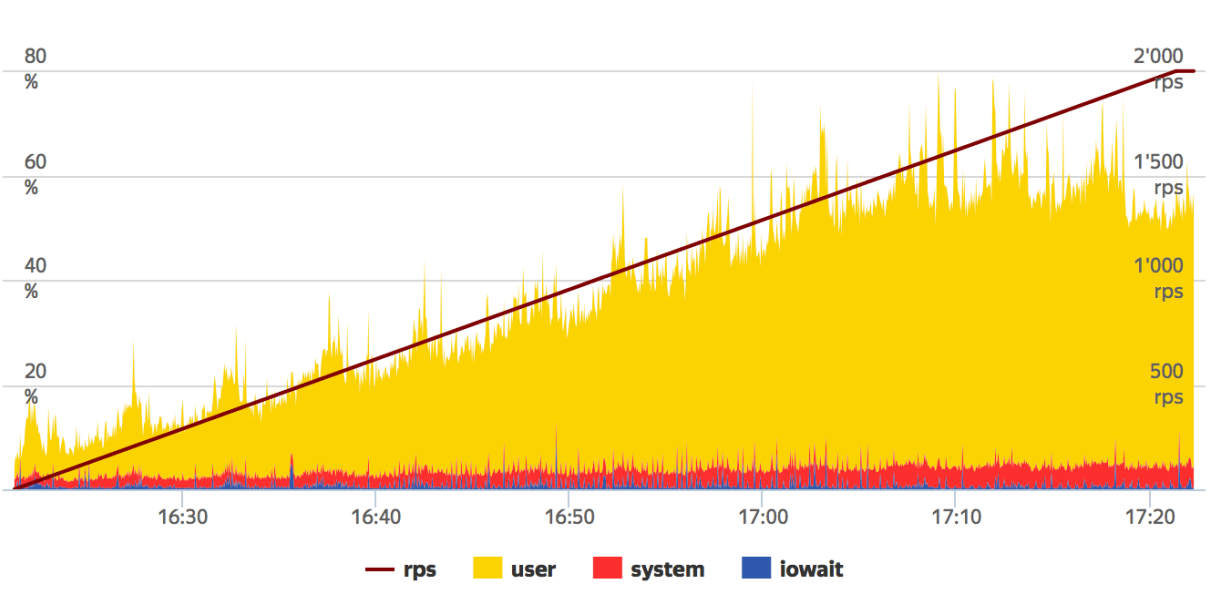
Table: 500 rps latency (ms)

Metric	50%	75%	90%	95%	99%	99.9%
Value	6	20	40	50	70	200

YoctoDB

- CPU usage under 30%
- 1200 rps still conforms to SLO


2'000 rps
1'500 rps
1'000 rps
500 rps
16:30
16:40
16:50
17:00
17:10
17:20
rps
0-9
10-99
100-999
≥1000


80 %
60 %
40 %
20 %
2'000 rps
1'500 rps
1'000 rps
500 rps
16:30
16:40
16:50
17:00
17:10
17:20
rps
user
system
iowait

Real Applications

Other

› Yandex.Realty database of buildings

› Yandex.Travel partitioned hotel database

› Read-only periodically rebuilt resources

› Hadoop Map-side `JOIN`

Conclusion

Limitations by Design

- In-memory database construction
- Updates through partition replacement
- No enforced schema
- No nested queries
- No query optimizer
- Benchmarked databases fit into disk cache

Achievements

- Drastically improved latencies
- Shrank search cluster
- Sped up index updates
- No need for GC tuning
- 100% line coverage by unit tests

Technology Scope

- If low latency and high throughput are paramount
- If your workload is stable
- If immutability after construction is acceptable
- If updates by partition reindexing are affordable
- If you implement a similar architecture

Takeaways

Takeaways

› Custom solution always wins

Takeaways

- Custom solution always wins
- Cost vs profit
  - Several man-months (over 3 years)
  - 10s of machines and energy over years

Takeaways

- Custom solution always wins
- Cost vs profit
  - Several man-months (over 3 years)
  - 10s of machines and energy over years
- Immutability simplifies everything
  - Optimal data layout (cache friendliness)
  - No need for synchronization

Future Work

github.com/yandex/yoctodb

- Data structures: tree/trie/hashmap, `RoaringBitSet`, sampling, etc.
- Parallel querying
- Aggregates
- Query tracing and database introspection
- Block compression
- Bindings to other languages

Contacts

Vadim Tsesko
Backend infrastructure

mail@incubos.org

Svyatoslav Demidov
Auto search engine

svyatoslav@my.com