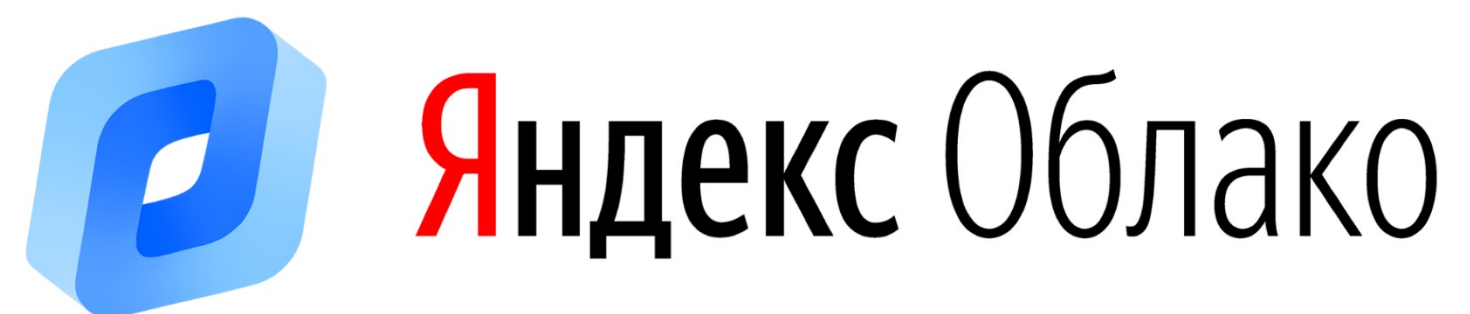


Яндекс



Новости Yandex Database

Антон Коваленко, Никита Лапков, Михаил Ситников — команда разработки Yandex Database

Содержание

- 01 | Доступ через интернет
- 02 | YDB как хранилище трейсов Jaeger
- 03 | Новые возможности: поддержка JSON
- 04 | Node.js SDK в публичном доступе
- 05 | Расширение возможностей диагностики
- 06 | Результаты запуска TPC Benchmark C

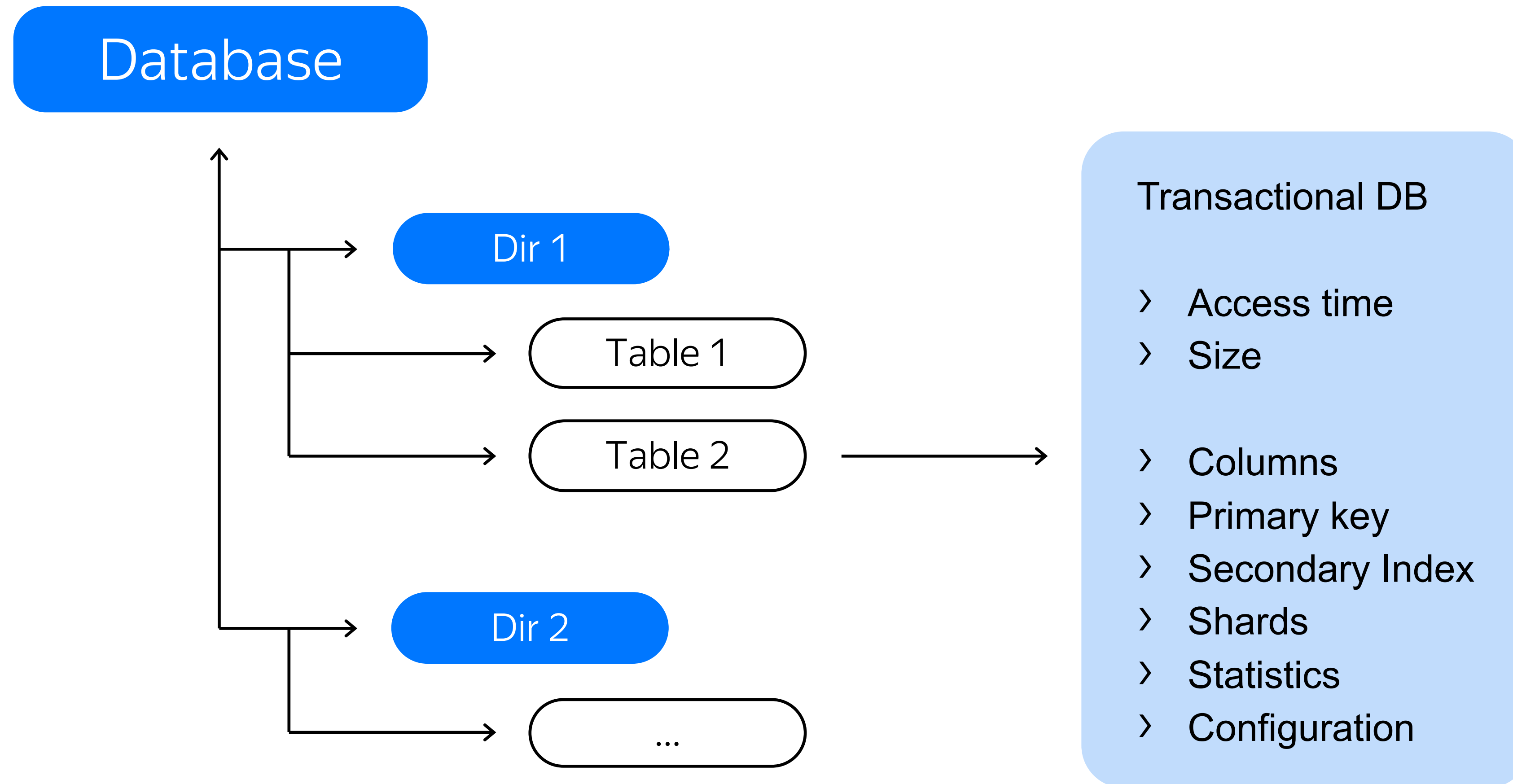
Что такое Yandex Database?

YDB — newSQL база данных масштаба Яндекса



- Нужна БД, но не простая, а:
 - › Геораспределённая и отказоустойчивая
 - › Со строгой консистентностью транзакций
 - › Горизонтально масштабируемая
 - › С высокой пропускной способностью
 - › С малым временем отклика

YDB: таблицы организованы в иерархию



YQL — диалект SQL

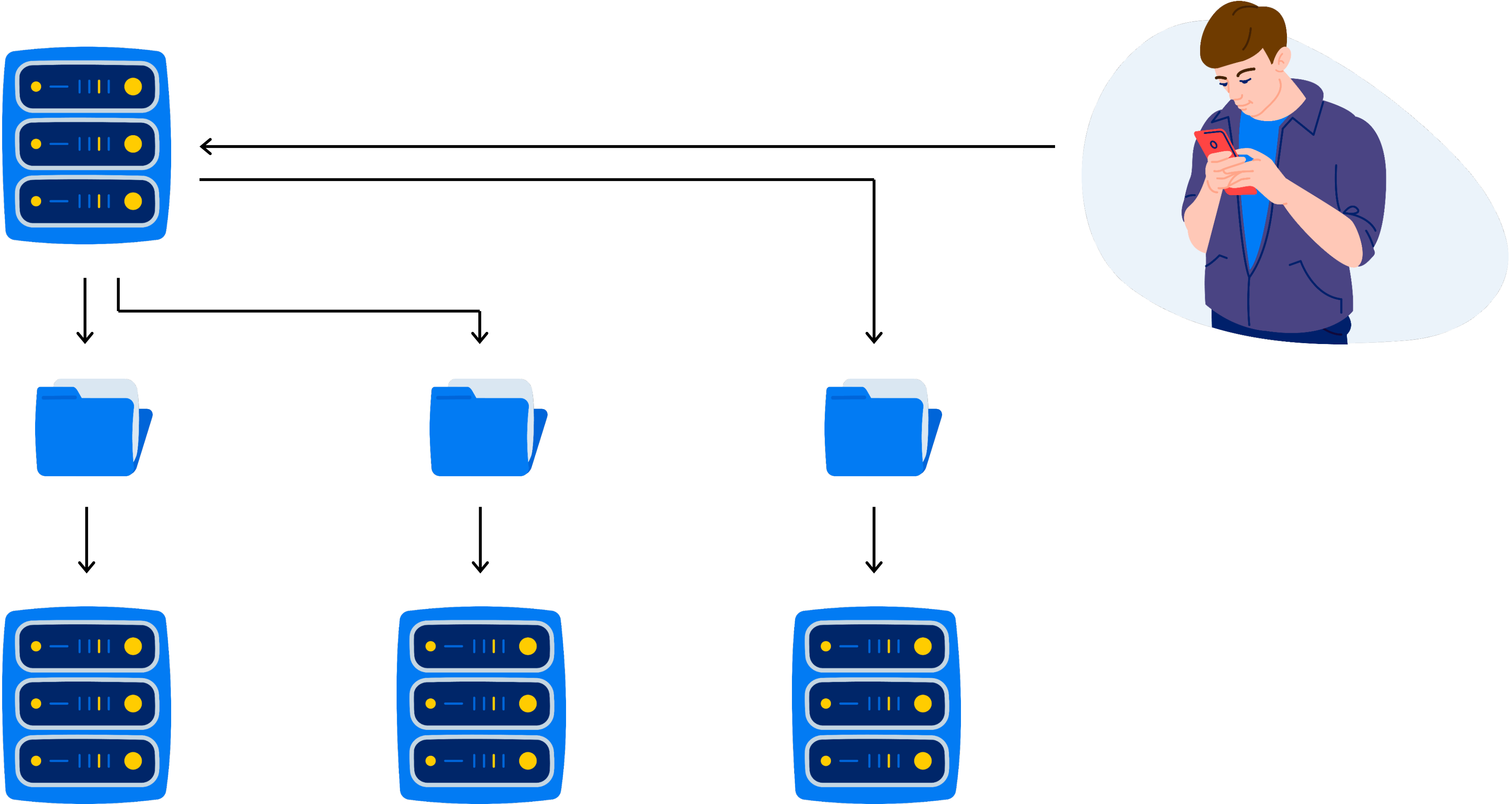
```
DECLARE $name AS String;
```

```
$name = "Foo" || $name;
```

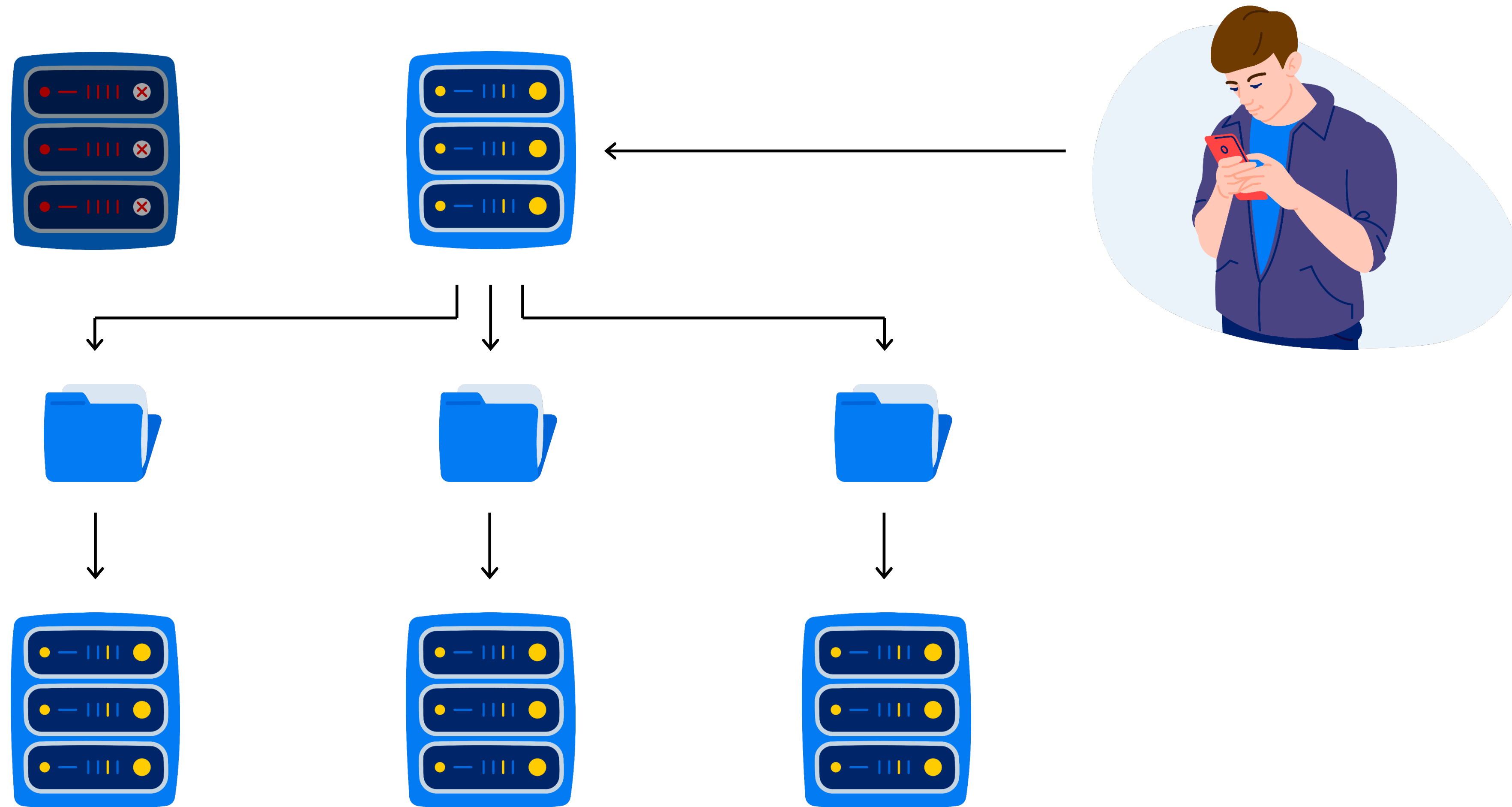
```
SELECT Id FROM Users WHERE Name = $name;
```

```
INSERT INTO Users (Id, Name) VALUES (1, "Bar");
```

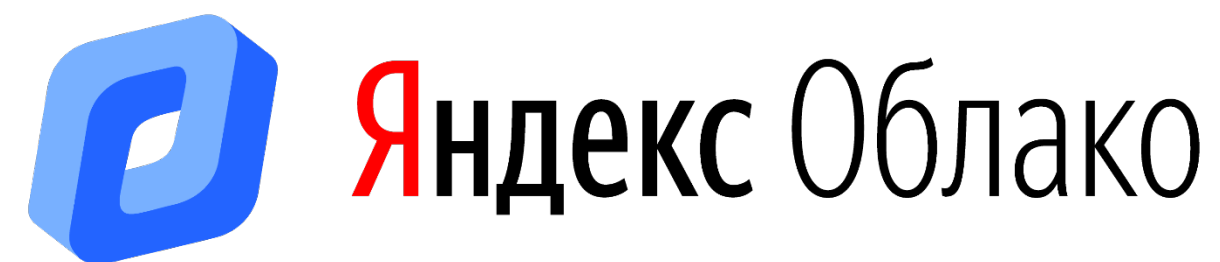
Геораспределённая система с синхронной репликацией данных



Автоматическое восстановление после сбоев



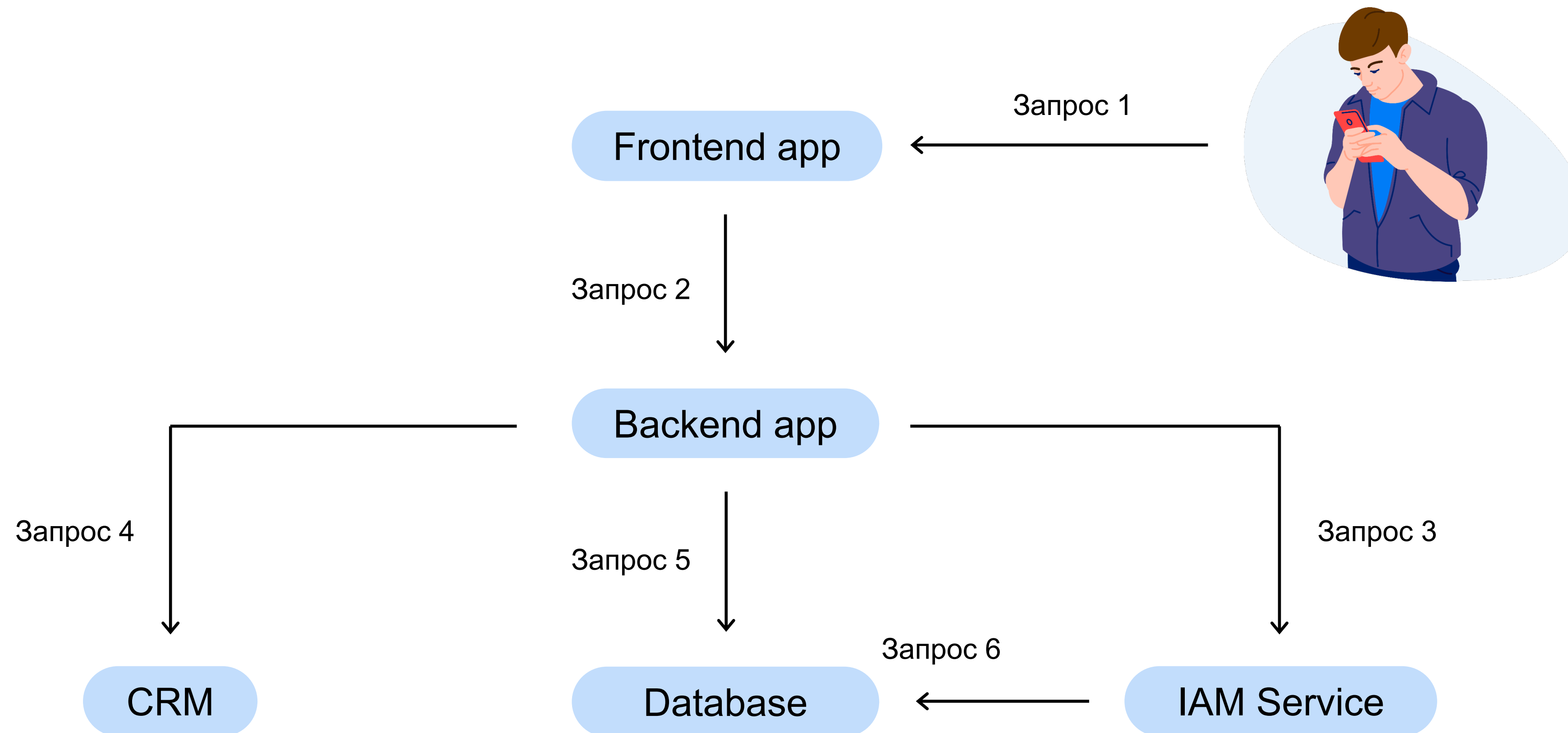
Yandex Database — сценарии использования



**YDB для хранения
трейсов Jaeger**

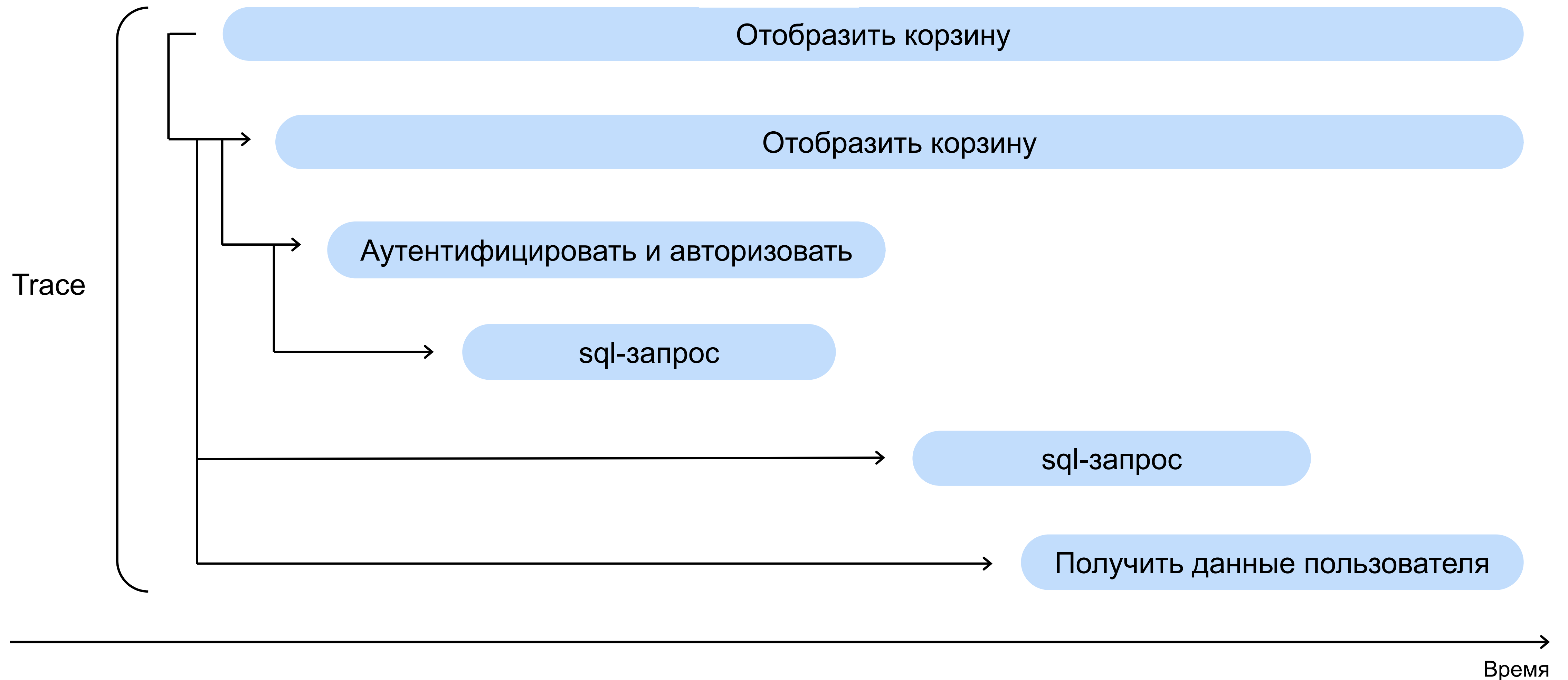
Что такое Distributed / Open Tracing

Что случилось с моим запросом?



Представление запросов в Distributed Tracing

Spans & Traces



Jaeger — удобный инструмент трассировки



- › Реализует OpenTracing
- › Позволяет собирать и анализировать трейсы
- › Разные системы хранения данных: Cassandra, Mongo
- › Используется в Auto.ru и Яндекс.Облаке

Jaeger Plugin к YDB



- › Сохранять трейсы
- › Искать по трейсам

Развернуть Jaeger в Яндекс.Облаке



Скринкаст или рассказ, как в Яндекс.Облаке можно развернуть Jaeger для X и хранить трейсы в YDB

Jaeger на YDB в Яндексе



- › Используется в Недвижимости, Auto.Ru и Яндекс.Облаке
- › Сотни тысяч спанов в секунду
- › Доклад на YaTalks 2019

Обзор **НОВЫХ ВОЗМОЖНОСТЕЙ**

Поддержка JSON

Никита Лапков, разработчик Yandex Database

Модели данных



- › Строгая схема, классические SQL СУБД
- › Документно-ориентированный подход, NoSQL-системы
- › Часто нужно что-то среднее

Пример — умный дом

DeviceID	UserID	...	SettingsID
123	456	...	789
		...	

SettingsID	BrightnessMin	BrightnessMax
789	0	100
	...	

Пример — умный дом

DeviceID	UserID	...	SettingsID
123	456	...	789
		...	

SettingsID	BrightnessMin	BrightnessMax	KelvinMin	KelvinMax
789	0	100	3000	5700
		...		

Пример — умный дом

DeviceID	UserID	...	SettingsID
123	456	...	[{ "property": "brightness", "type": "kelvin", "min": 3000, "max": 5700 }, ...]
		...	

Стандарт SQL / JSON 2017



- › Стандартизированный способ работы с JSON
- › Частично поддержан во всех популярных БД
- › Скоро в YDB

Язык JsonPath



- › Извлекает значения из JSON
- › Поддерживает фильтрацию значений

Язык JsonPath

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: \$

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: `$.friends`

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: `$.friends[0]`

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: `$.friends[0].name`

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: `$.friends[1 to last].name`

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```

Язык JsonPath

Запрос: `$.friends ? (@.age >= 18)`

JSON:

```
{  
  "name": "Nikita",  
  "friends": [  
    {"name": "Masha", "age": 21},  
    {"name": "Natasha", "age": 15},  
    {"name": "Ann", "age": 30}  
  ]  
}
```


Запросные функции



- › JSON_EXISTS
- › JSON_VALUE
- › JSON_QUERY

Проверка существования

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Проверка существования

```
SELECT COUNT (*)  
FROM logs  
WHERE JSON_EXISTS(Browser, "$.version")
```

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Проверка существования

```
SELECT COUNT(*)  
FROM logs  
WHERE JSON_EXISTS(Browser, "$.version")
```

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Извлечение скалярного значения

```
SELECT  
    JSON_VALUE(Browser, "$.version")  
FROM logs
```

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Извлечение скалярного значения

```
SELECT  
    JSON_VALUE(Browser, "$.version")  
FROM logs
```

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Извлечение скалярного значения

```
SELECT  
    JSON_VALUE(Browser, "$.version"  
        DEFAULT "IE 6.0" ON EMPTY)  
FROM logs
```

Browser

```
{"version": "Chrome 81.0.4", "cookies": [...]}
```

```
{"cookies": [...]}
```

```
{"version": "Internet Explorer 11.0"}
```

Извлечение скалярного значения

```
SELECT  
    JSON_VALUE (Browser, "$.version"  
        DEFAULT "IE 6.0" ON EMPTY)  
FROM logs
```

Output

"Chrome 81.0.4"

"IE 6.0"

"Internet Explorer 11.0"

Извлечение массивов и объектов

Browser

```
{..., "cookies": ["skynet", "is", "watching"]}
```

```
{..., "cookies": ["super", "tracking"]}
```

```
{...}
```

Извлечение массивов и объектов

```
SELECT
```

```
    JSON_QUERY (Browser, "$.cookies")
```

```
FROM logs
```

Browser

```
{..., "cookies": ["skynet", "is", "watching"]}
```

```
{..., "cookies": ["super", "tracking"]}
```

```
{...}
```

Извлечение массивов и объектов

```
SELECT  
    JSON_QUERY(Browser, "$.cookies")  
FROM logs
```

Browser

```
{..., "cookies": ["skynet", "is", "watching"]}
```

```
{..., "cookies": ["super", "tracking"]}
```

```
{...}
```

Извлечение массивов и объектов

```
SELECT  
    JSON_QUERY(Browser, "$.cookies"  
        EMPTY_ARRAY ON EMPTY)  
FROM logs
```

Browser

```
{..., "cookies": ["skynet", "is", "watching"]}
```

```
{..., "cookies": ["super", "tracking"]}
```

```
{...}
```

Извлечение массивов и объектов

```
SELECT
  JSON_QUERY(Browser, "$.cookies"
    EMPTY_ARRAY ON EMPTY)
FROM logs
```

Output

```
["skynet", "is", "watching"]
```

```
["super", "tracking"]
```

```
[]
```

Скоро в YDB



- › Извлечение и фильтрация JSON значений
- › Запросные функции с обработкой ошибок
- › Совместимо со стандартом SQL/JSON 2017

Node.js SDK

Михаил Ситников, разработчик Yandex Database

Диагностика запросов: системные таблицы



SQL для работы с метаданными

- › Размещаются в служебной директории `.sys`
- › Информация о таблицах
- › Статистика выполнения запросов

Статистика запросов



.sys/top_queries_* для поиска неоптимальных запросов

› Запросы по времени выполнения

- top_queries_by_duration_one_minute
- top_queries_by_duration_one_hour

› Запросы по размеру чтения

- top_queries_by_read_bytes_one_minute
- top_queries_by_read_bytes_one_hour

Статистика по таблицам



.sys/partition_stats для поиска перегруженных таблиц и шардов

- › Размер таблицы и партиций
- › Потребление ресурсов
- › Количество запросов на запись / чтение

Вопросы?



Заключение

Про что был вебинар?



- › Публичный доступ через интернет
- › Теперь доступна и Node.js SDK
- › Системные таблицы для диагностики
- › YDB можно хранить трейсы Jaeger

Ссылки

Yandex Database Public Preview

cloud.yandex.ru/services/ydb

Выступления и материалы

cloud.yandex.ru/docs/ydb/public_talks

Python SDK

github.com/yandex-cloud/ydb-python-sdk

Go SDK

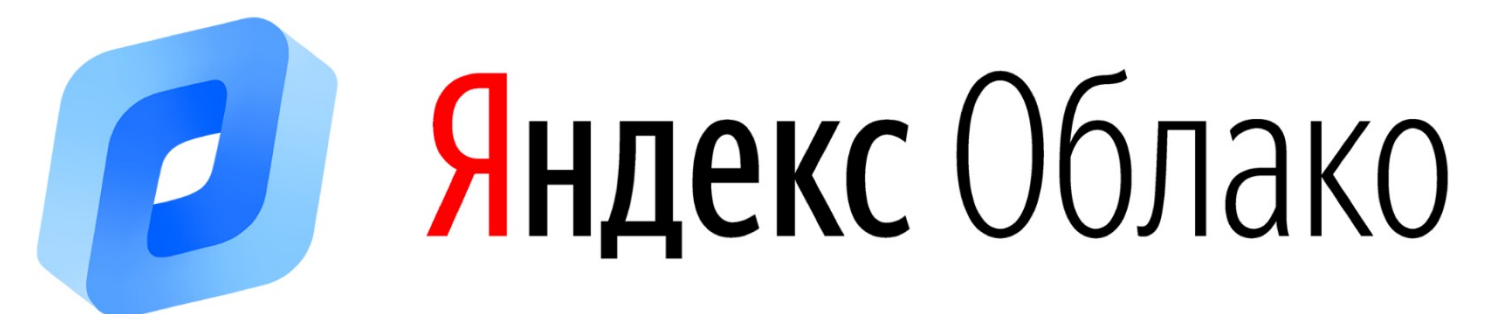
github.com/yandex-cloud/ydb-go-sdk

Java SDK

github.com/yandex-cloud/ydb-java-sdk

Node.js SDK

github.com/yandex-cloud/ydb-nodejs-sdk



Спасибо!

Антон Коваленко, Никита Лапков, Михаил Ситников

Команда разработки Yandex Database

 ydb-feedback@yandex-team.ru