Чтение с реплик в распределённых системах: опыт YDB

Александр Зевайкин

Руководитель группы разработки, YDB
Кандидат технических наук, доцент

База данных — узкое место!

1 сервер — не масштабируется для миллионов пользователей

Требуются новые архитектурные подходы

Реальность

01 **100 млн пользователей**

02 1 млн RPS на чтение

03 Горячие ключи

04 Несколько ДЦ

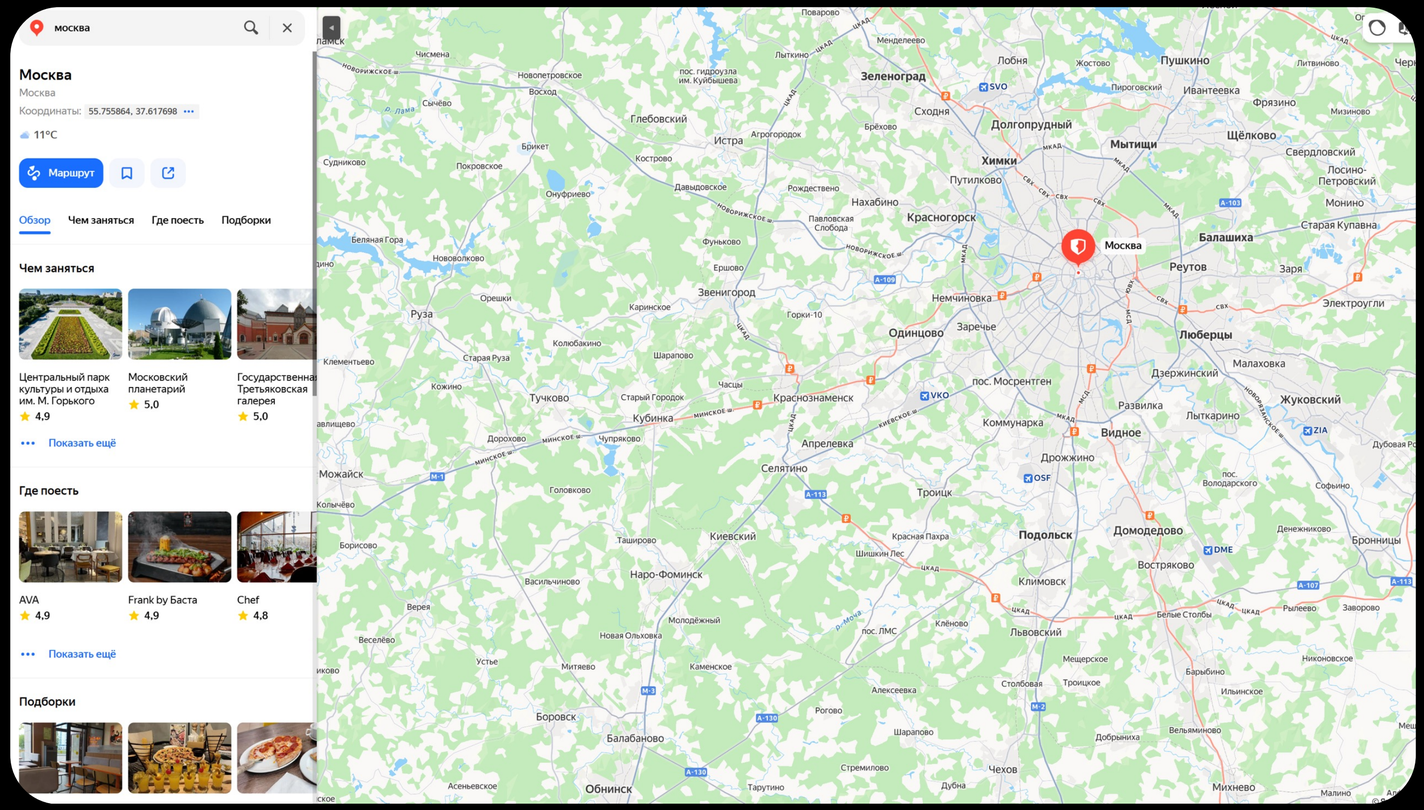
Реальность

01 100 млн пользователей

02 **1 млн RPS на чтение**

03 Горячие ключи

04 Несколько ДЦ



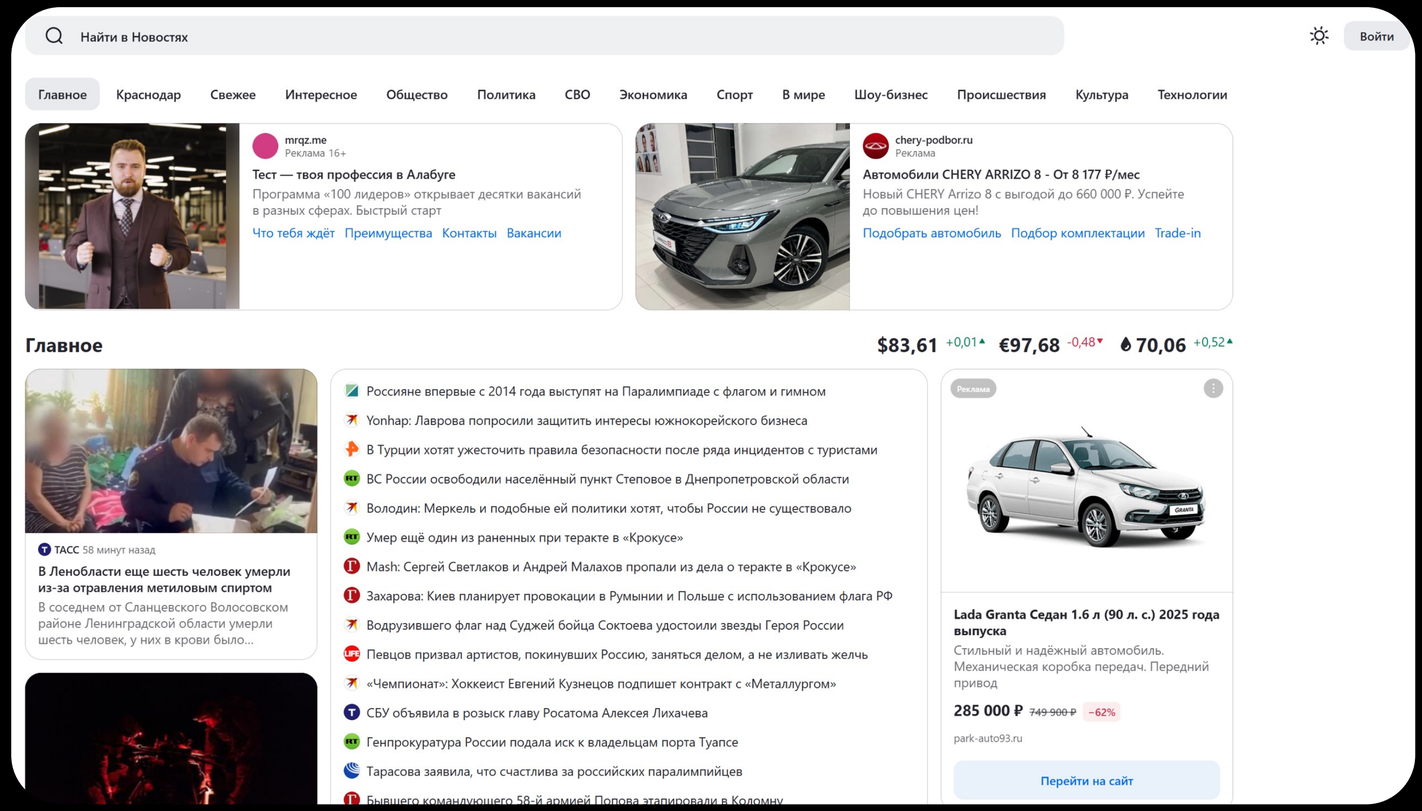
Реальность

01 100 млн пользователей

02 1 млн RPS на чтение

03 **Горячие ключи**

04 Несколько ДЦ



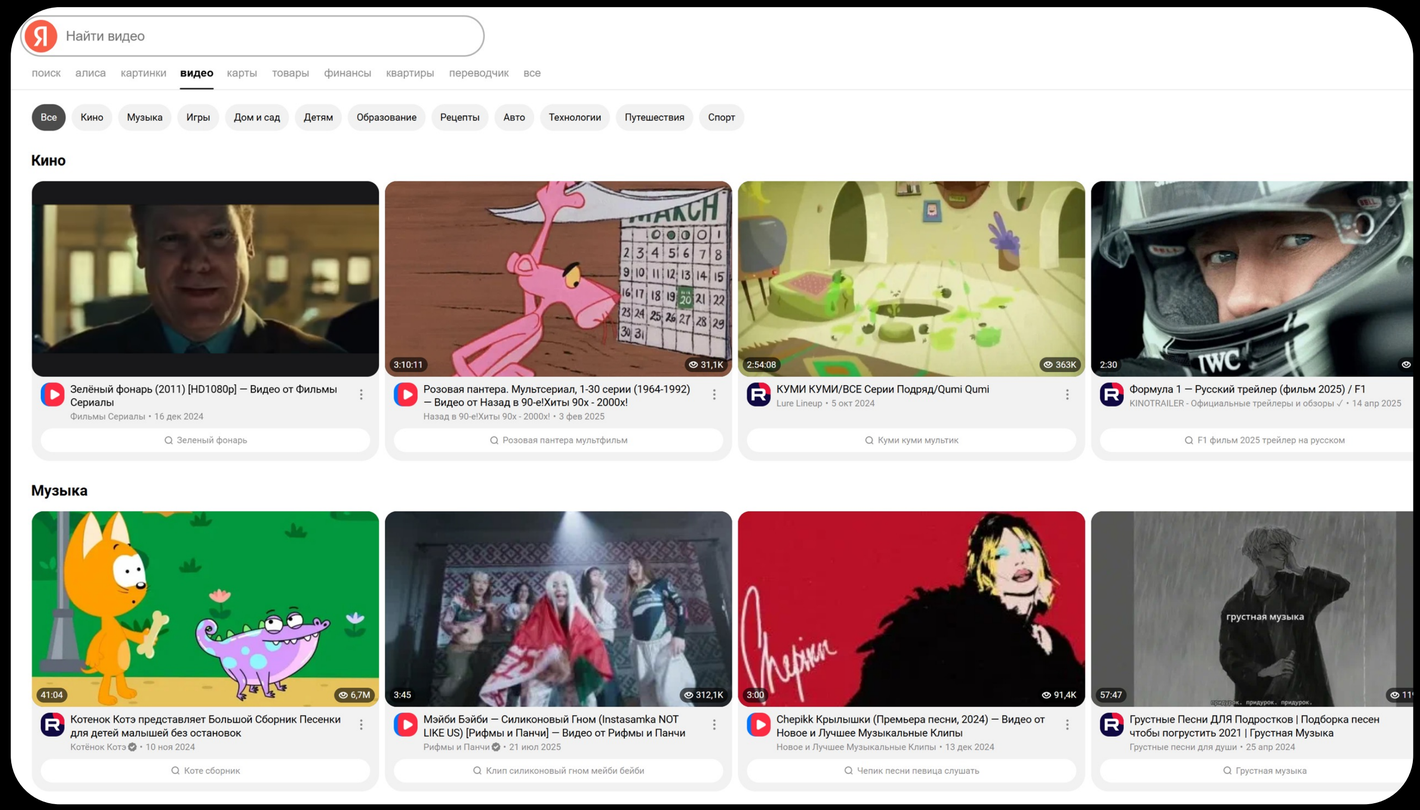
Реальность

01 100 млн пользователей

02 1 млн RPS на чтение

03 Горячие ключи

04 **Несколько ДЦ**



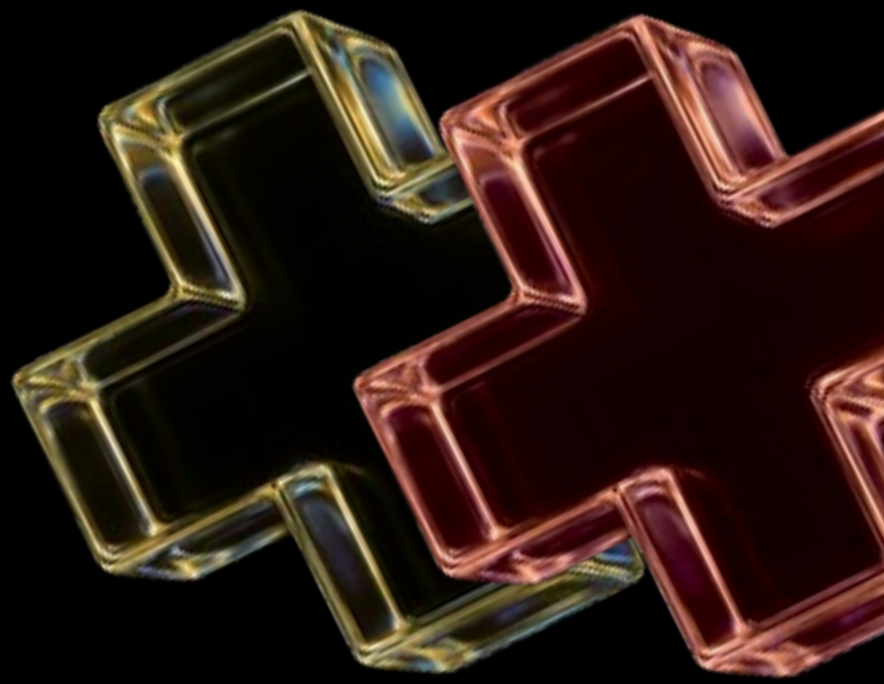
Требования

Задержка 1 мс

даже при нескольких ДЦ

Высокая пропускная способность

горизонтальное масштабирование чтения



Решение: чтение с реплик

⊕ Устранение Cross-AZ-запросов на чтение

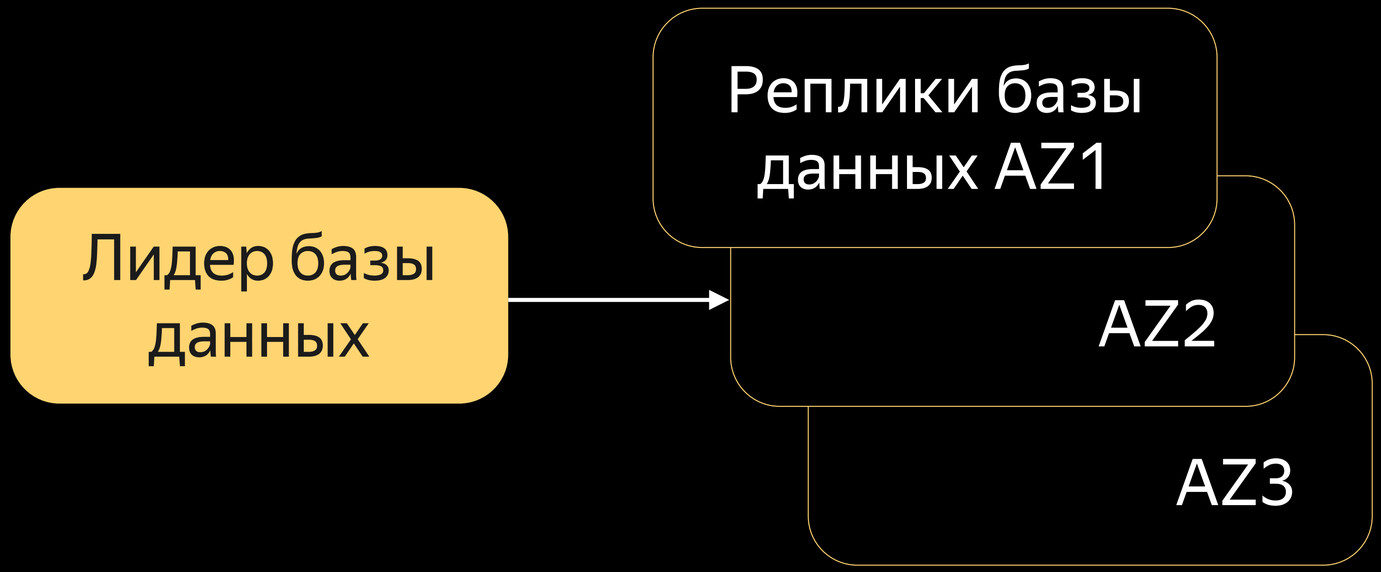
несколько реплик в каждой AZ

⊕ Снижение нагрузки на лидеров

писатели не мешают читателям

⊕ Увеличение числа одновременных чтений

много реплик



Эволюция репликации в популярных СУБД

PostgreSQL, MySQL, Redis

лидер-реплика

MongoDB

набор реплик

Cassandra

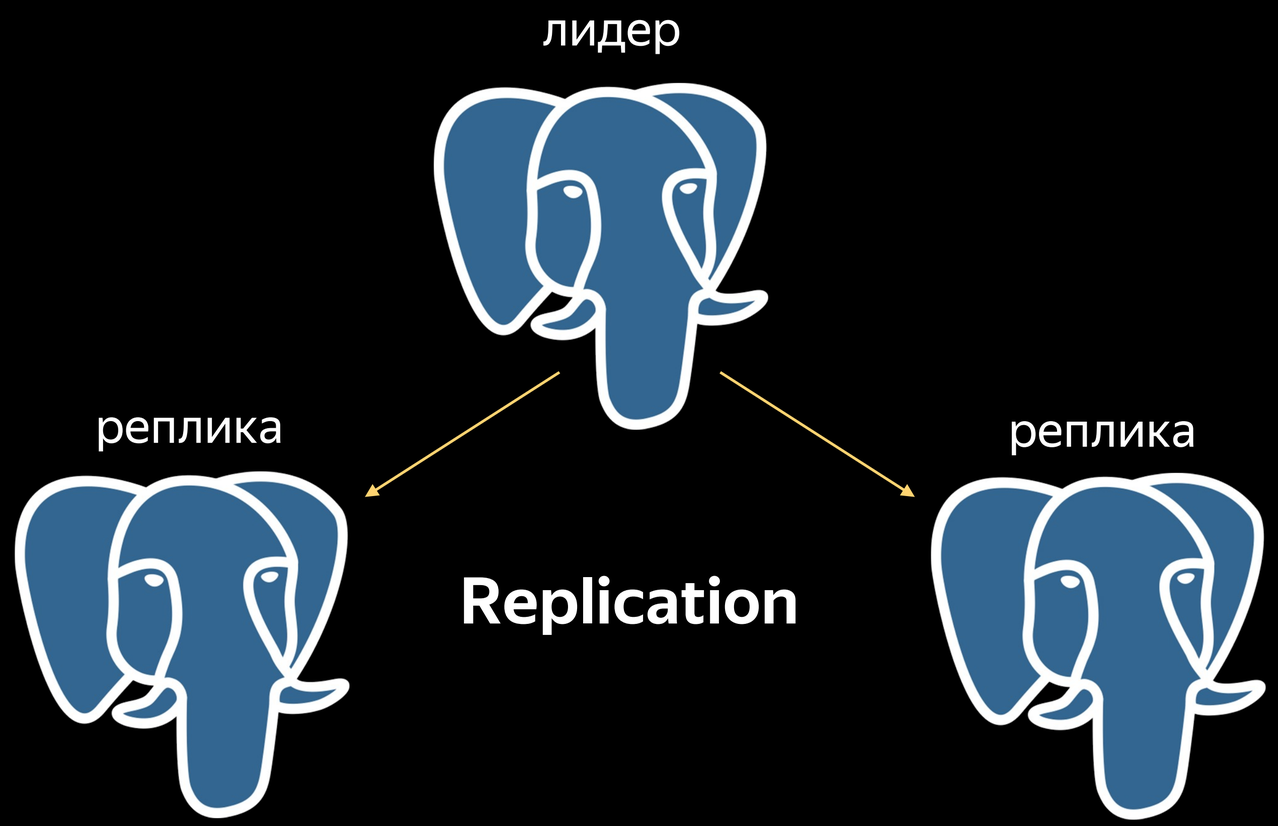
одноранговая архитектура

CockroachDB, Google Spanner, TiDB, YugabyteDB

реплики партиций

PostgreSQL, MySQL, Redis: лидер-реплика

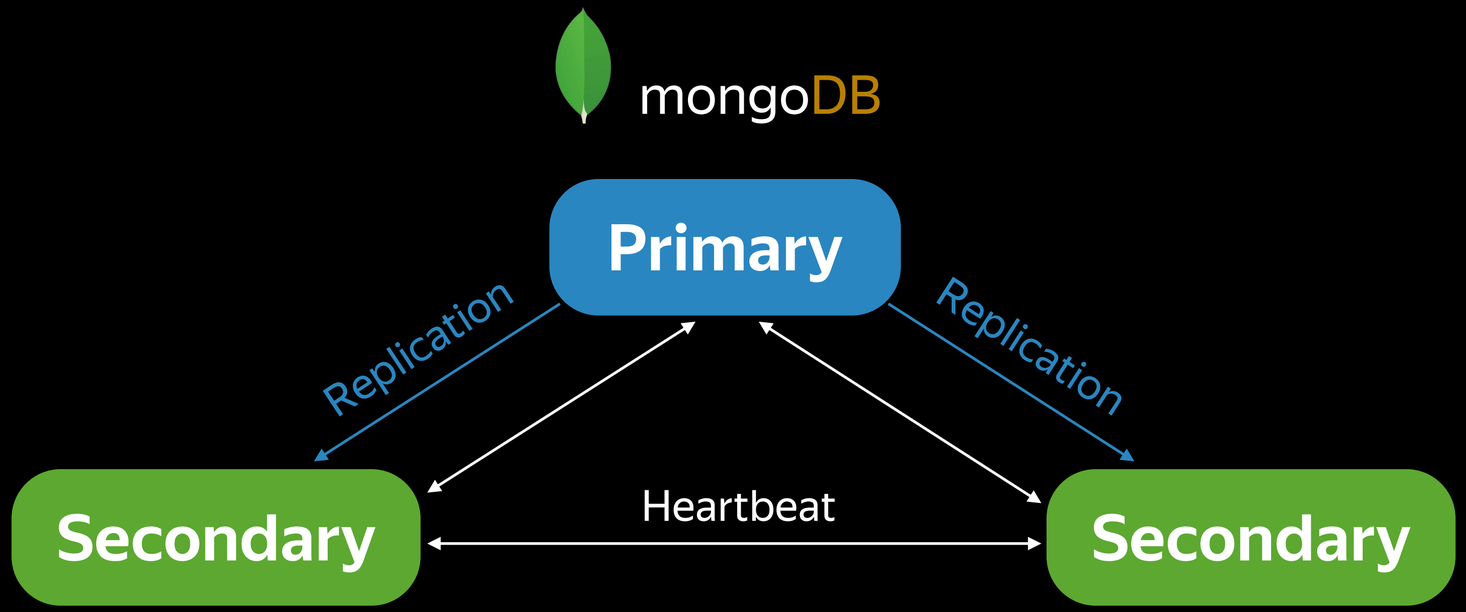
- **Лидер**
  только он принимает запись
- **Реплики**
  только чтение
- **Репликация**
  асинхронная
- **Выбор лидера**
  ручной или автоматический



MongoDB: набор реплик

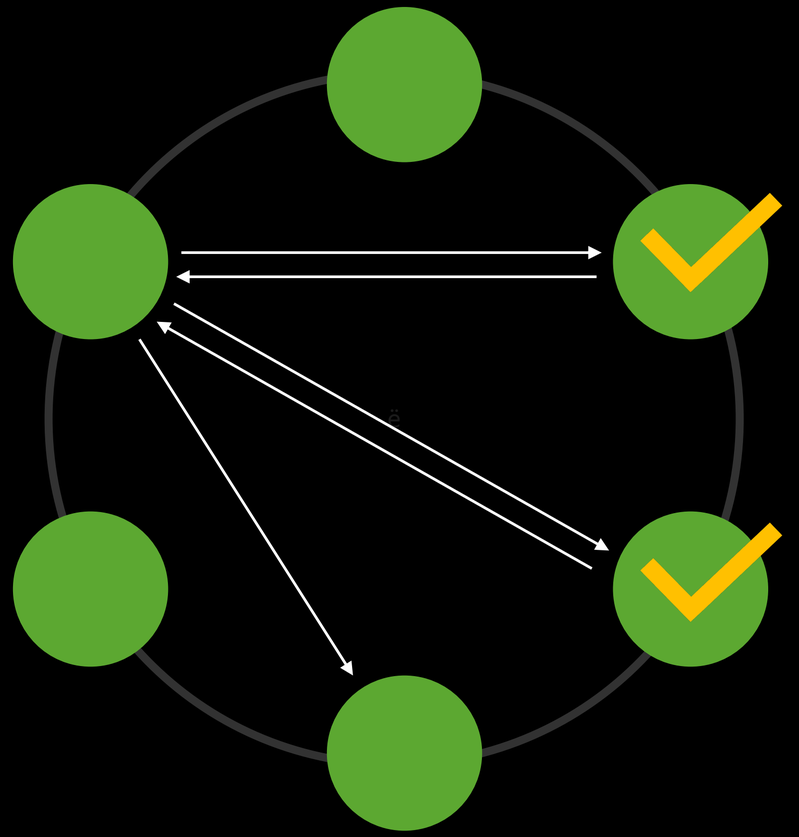
Выбор лидера

автоматически



Cassandra: одноранговая архитектура

- Нет лидеров
  все равны
- Любой узел
  чтение/запись



CockroachDB, Google Spanner, TiDB, YugabyteDB: реплики партиций

- **Партиции**
  у каждой свой лидер и свои реплики
- **Запись**
  только в лидера партиции
- **Выбор лидера**
  автоматически

01

Поиск и Рекламные технологии

YDB

Распределённая SQL-база данных

YDB

Распределённая SQL-база данных
для операционных нагрузок

ydb.tech

github.com/ydb-platform/ydb

01 SQL
для OLTP

02 Горизонтальное
масштабирование

03 Транзакции с гарантиями ACID
в нескольких AZ

04 Работоспособность и автоматическое
восстановление при отказах

05 Масштабирование на миллионы
транзакций в секунду и сотни
терабайт данных

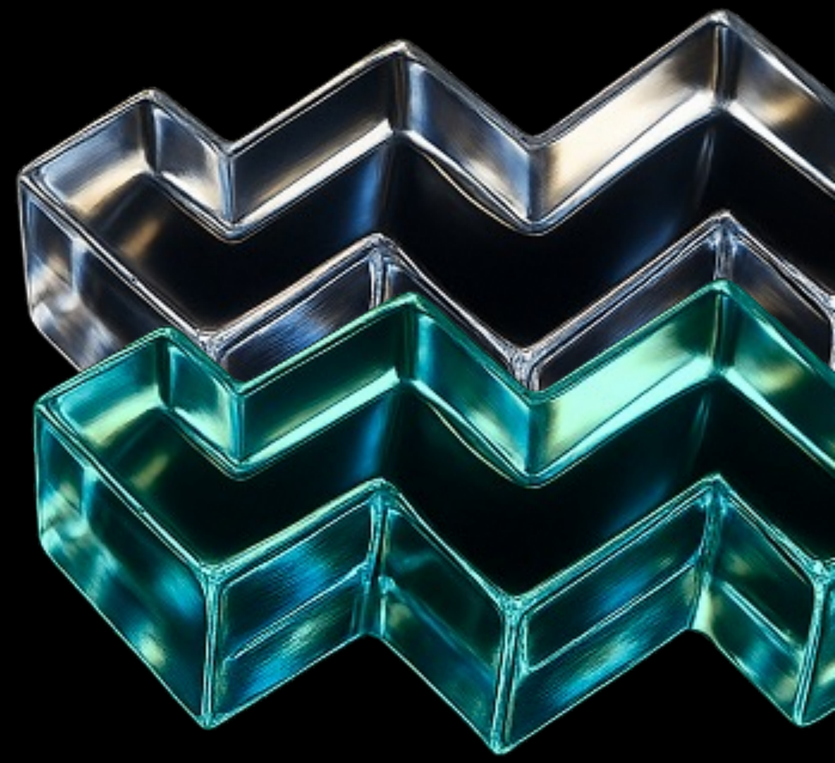
YDB: 2 режима репликации

Асинхронная репликация

как в PostgreSQL,
MySQL, Redis

Реплики партиций

как в CockroachDB, Google Spanner, TiDB, YugabyteDB



YDB (лидер-реплика)

Асинхронная репликация

- поверх CDC (Change Data Capture)
- оперирует логическими данными

Похоже на PostgreSQL,
MySQL, Redis

Выбор лидера

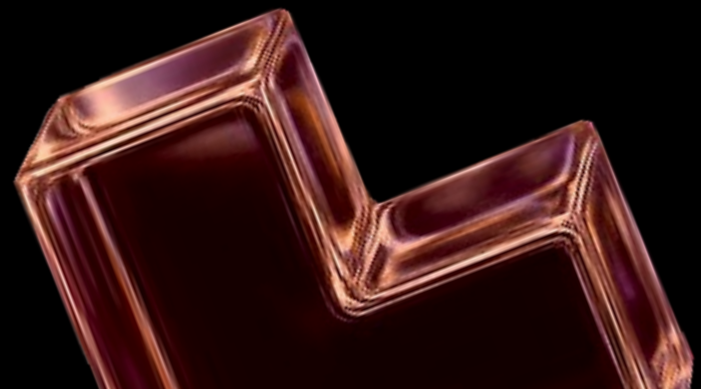
Ручной

Строковые таблицы YDB

Key	Data	
82	Иванов	Таблетка
283	Петров	
346	Сидоров	Таблетка
1273	Григорьев	
2489	Антонов	Таблетка
3578	Клюев	

Таблетка —
легковесный логический процесс

Отвечает за непрерывный диапазон первичных ключей и соответствующих им данных



Строковые таблицы YDB (несколько ДЦ)

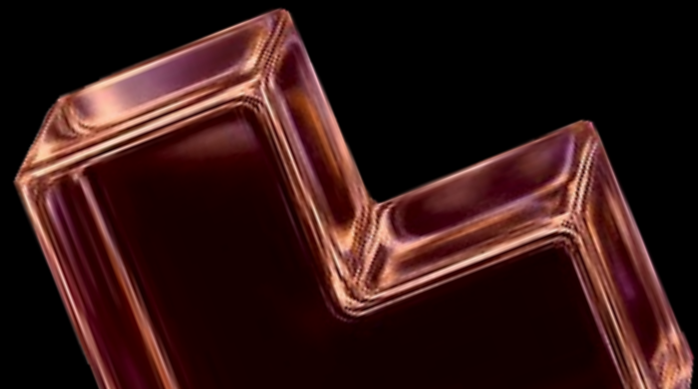
Key	Data
82	Иванов
283	Петров
346	Сидоров
1273	Григорьев
2489	Антонов
3578	Клюев

Таблетка AZ 1

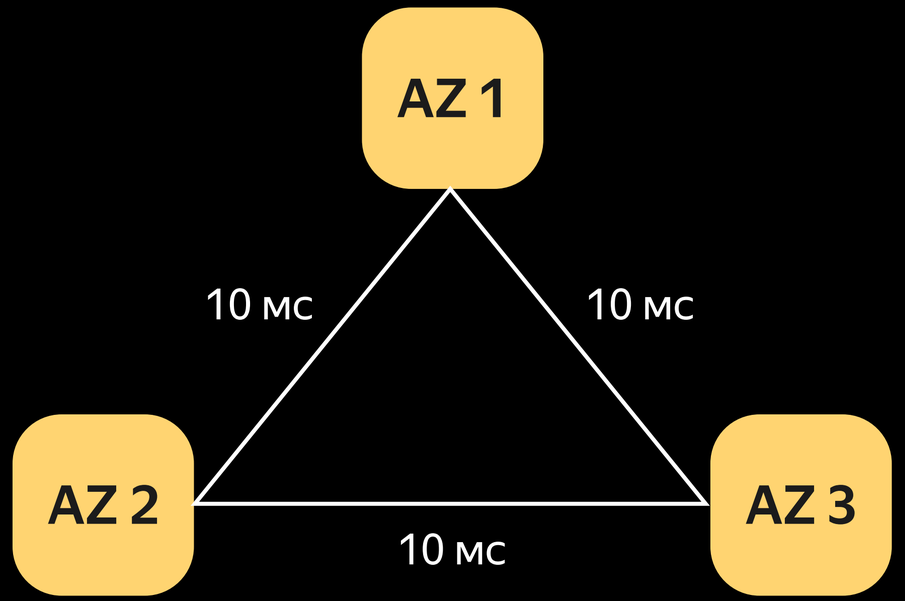
Таблетка AZ 2

Таблетка AZ 3

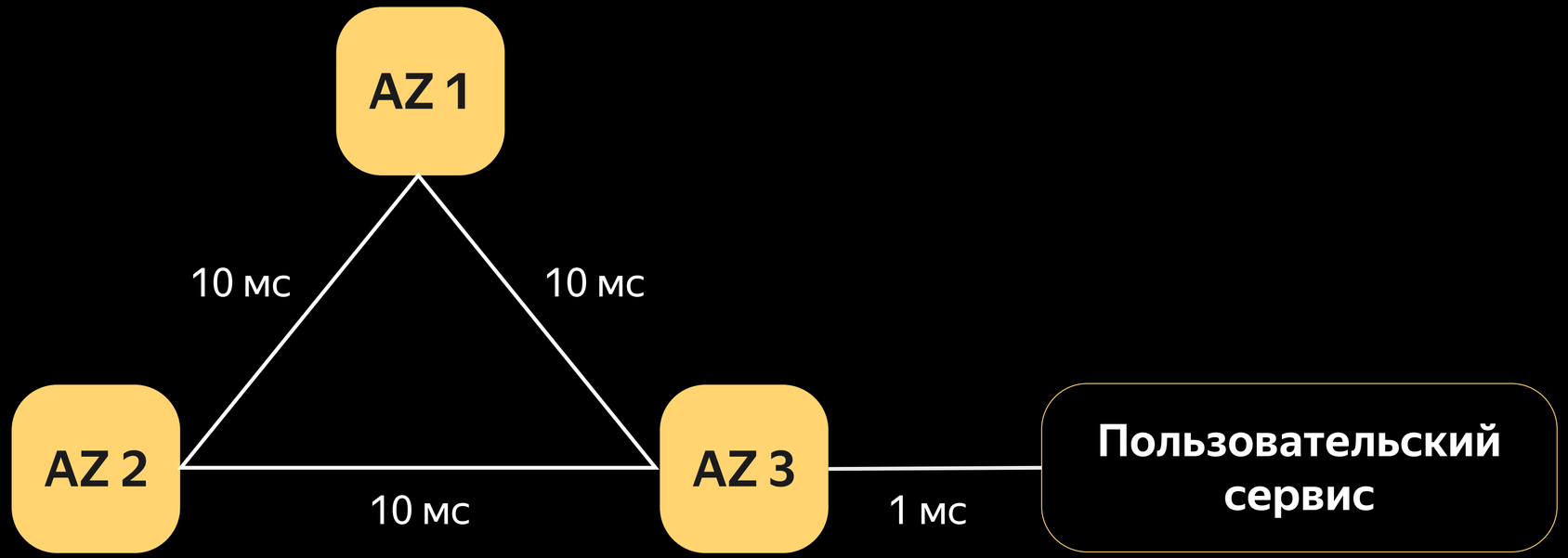
Таблетка может находиться в любой AZ и переезжать между ними



Данные в нескольких AZ

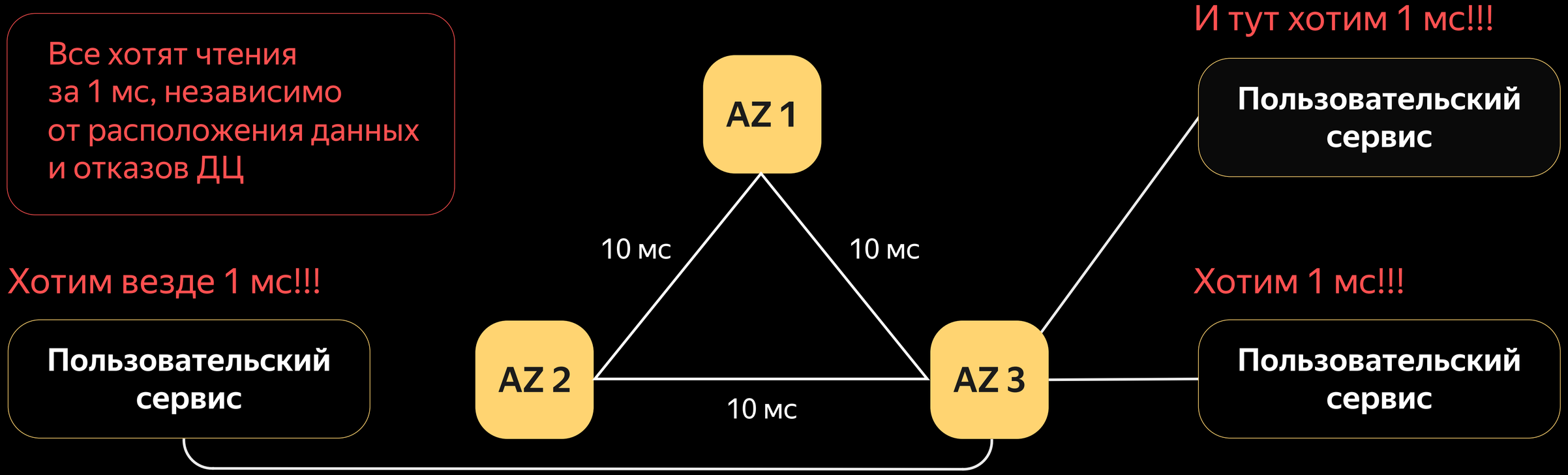


Данные в нескольких AZ + пользователь



Проблема пользователей

Все хотят чтения
за 1 мс, независимо
от расположения данных
и отказов ДЦ



Примеры таких пользователей YDB

Техплатформа
Екома и Райдтеха
Яндекса

граф fine-grained-авторизации

Поиск по видео /
Кинопоиск

информация о фильмах
и просмотрах

Яндекс Карты

пользовательские данные
на карте — лайки, фото

Авторизация Екома и Райдтеха Яндекса

Relationship-based access control (ReBAC)

Время проверки прав доступа

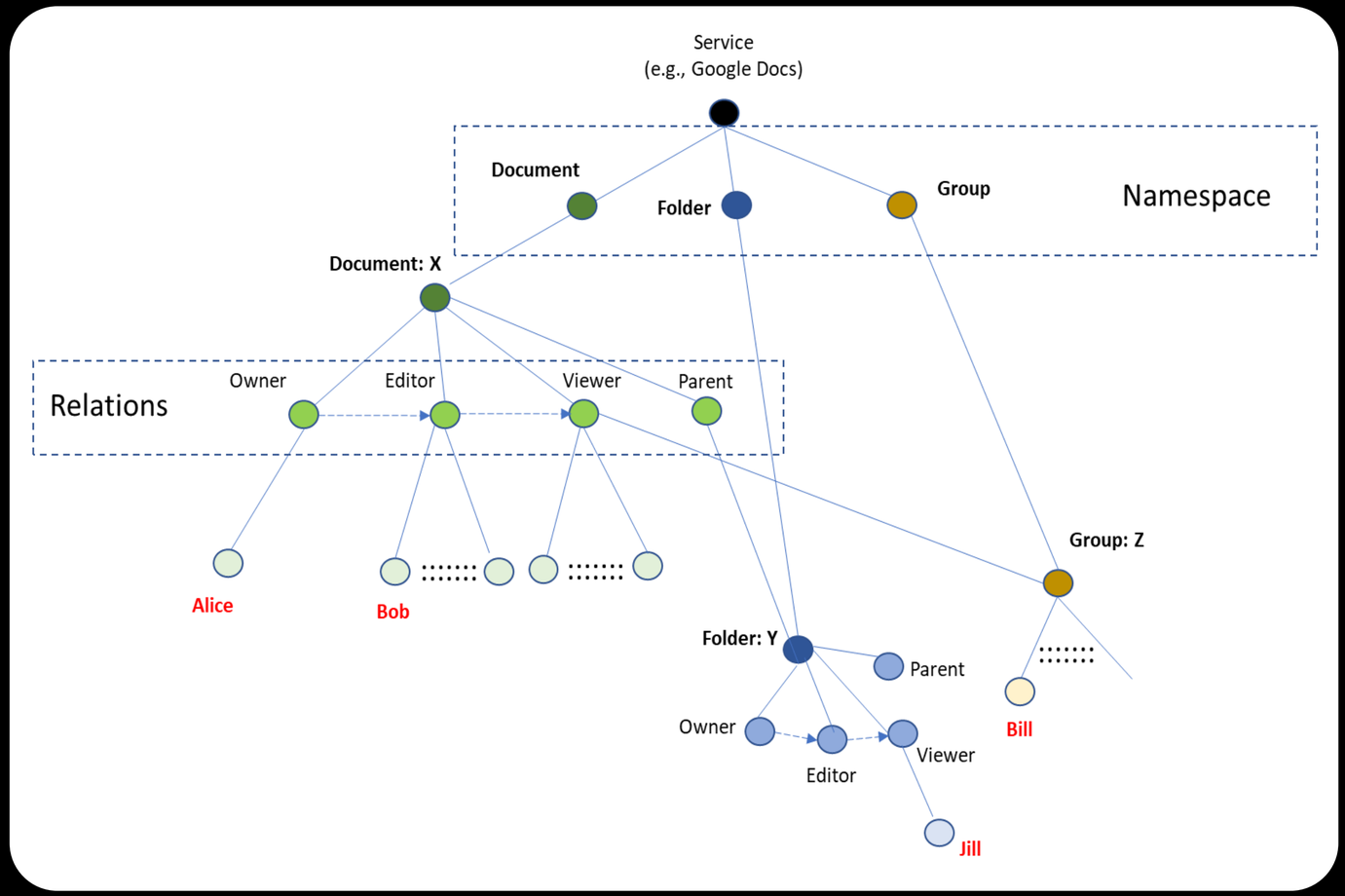
количество рёбер на пути

* время одного чтения

Длина пути

- p50 — 3 чтения
- P99 — 20 чтений

Даже 10–20 мс — это бутылочное горлышко. Требуется 1 мс!



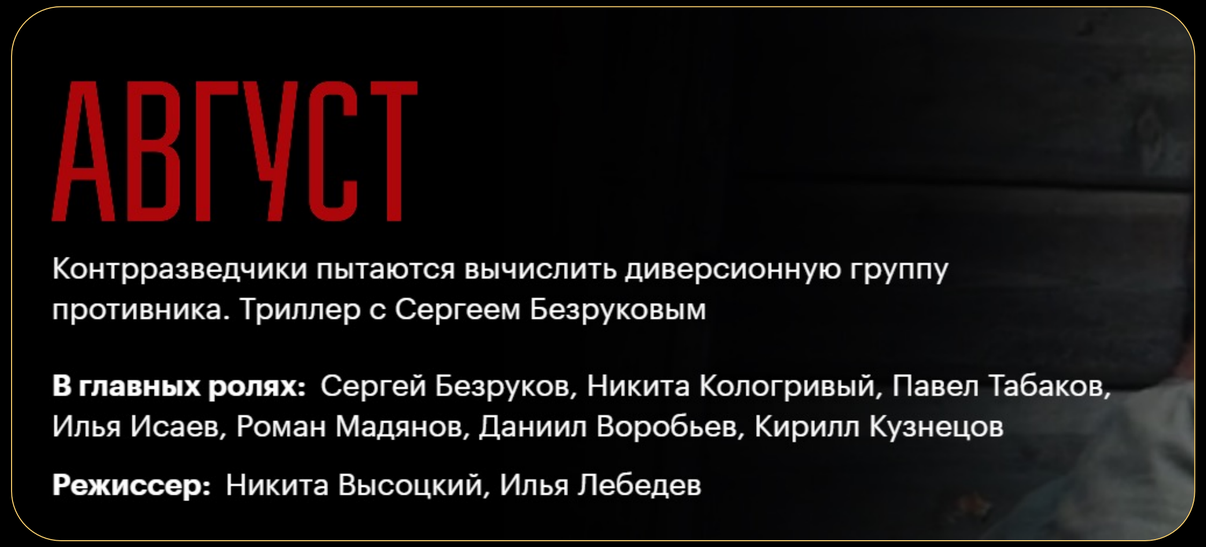
Яндекс Поиск по видео / Кинопоиск

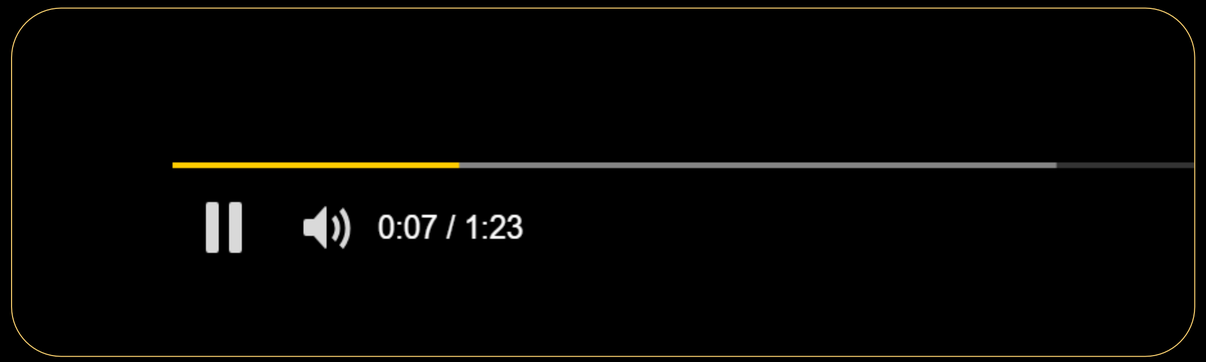
01 Метаданные сериалов и фильмов

- Минимальная нагрузка на запись
- Средний поток чтения

02 Прогресс просмотра пользователей

- Огромный поток записей
  - пользователь смотрит видео
- Высокий поток чтений
  - просмотр истории с разных устройств

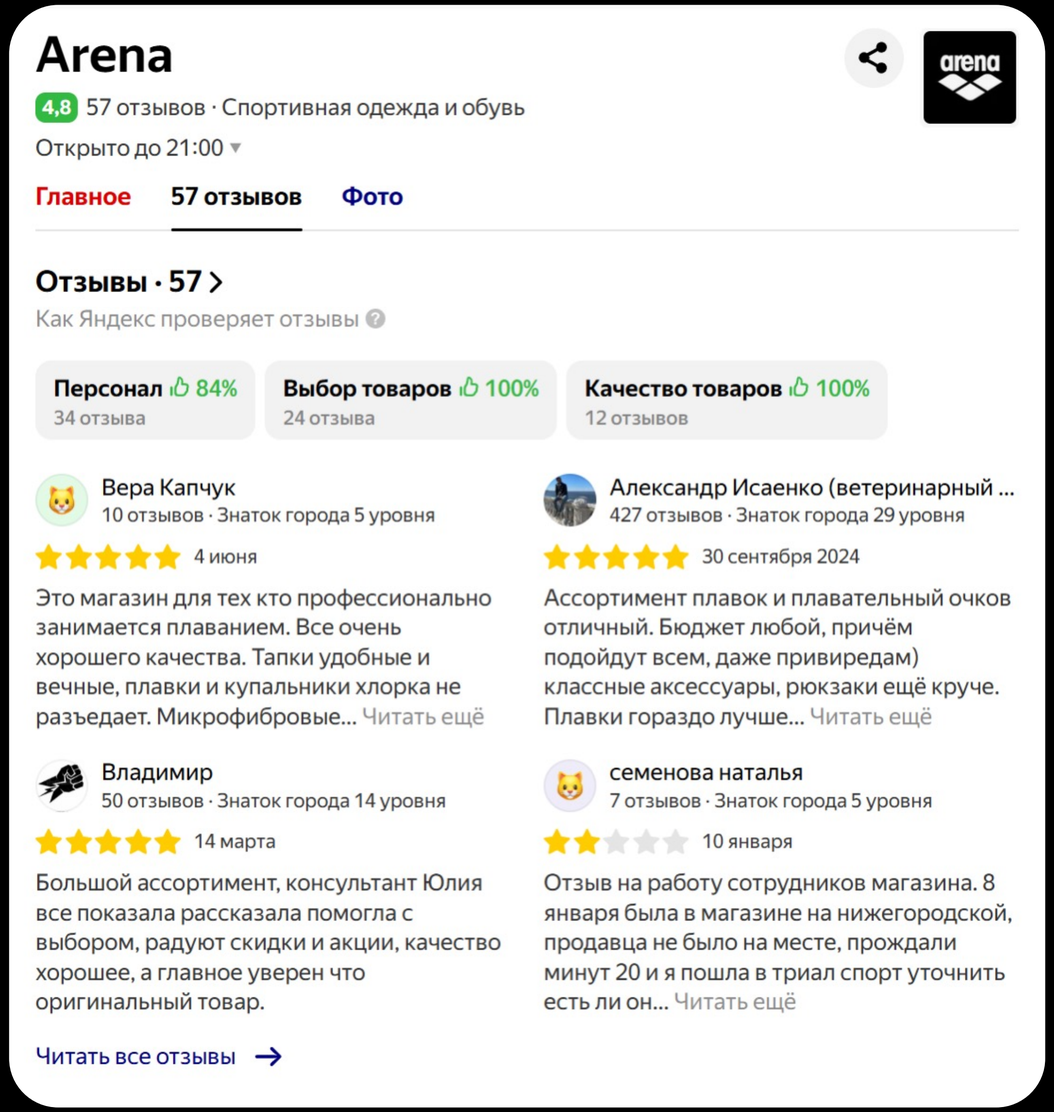




Яндекс Карты

Пользовательские данные
на карте — лайки на отзывы/фото

- Высокий поток записи
- Высокий поток чтения



02

Поиск и Рекламные технологии

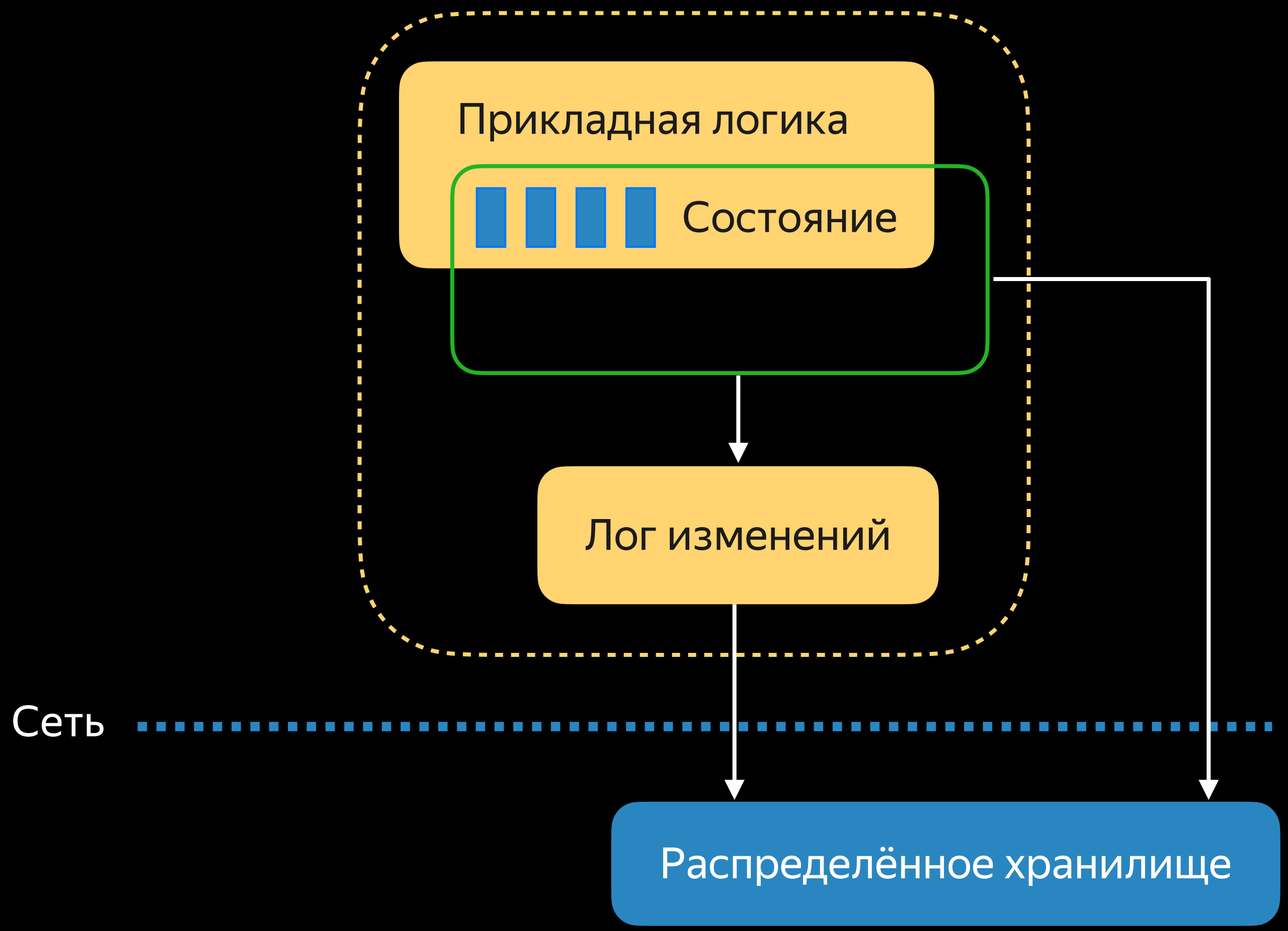
YDB

Как устроены реплики

YDB: лидер-реплика по партициям

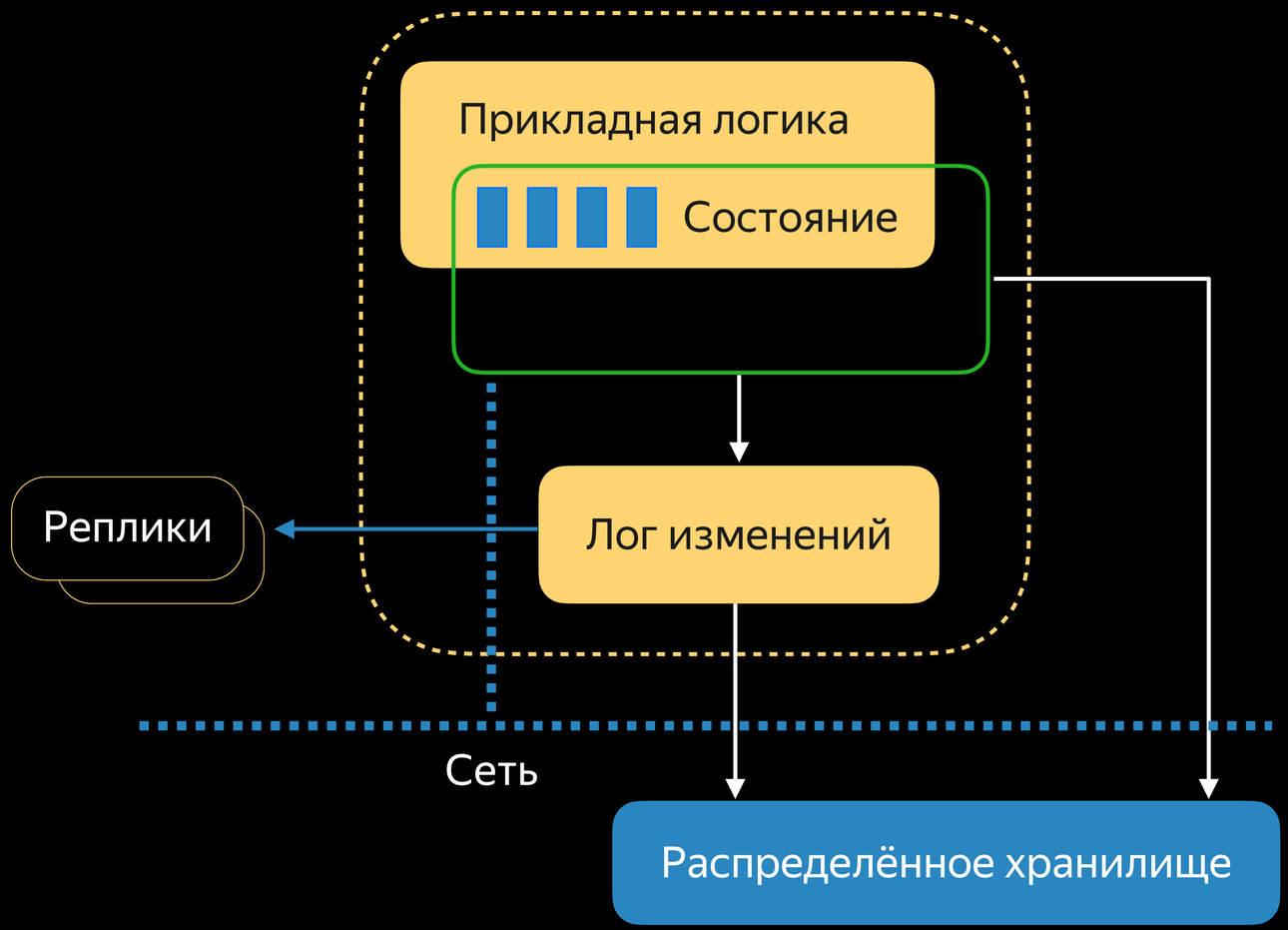
- Лидер и реплики на уровне каждой партиции
- Запись только на лидере
- Чтение с лидера или локальной реплики
- Автоматическое переключение лидера на реплику

Таблетка



- Хранит состояние в распределённом хранилище в нескольких AZ
- Пишет лог изменений в распределённое хранилище
- Запускается на любом вычислительном узле
- Существует в единственном экземпляре на партицию

Как устроены реплики (followers) в YDB



- Держат копию памяти лидера
- Получают изменения после коммита
- Не требуют нового места на диске
- Несколько реплик на партицию разрешено

Как читать свежие данные?

Простыми словами

Сервис показывает баланс на счёте пользователя:

- есть счёт, на котором есть 500K
- обрабатываем транзакцию по добавлению на счёт 1M
- пока она незакоммичена, хотим показывать всё ещё 500K

Как читать свежие данные?

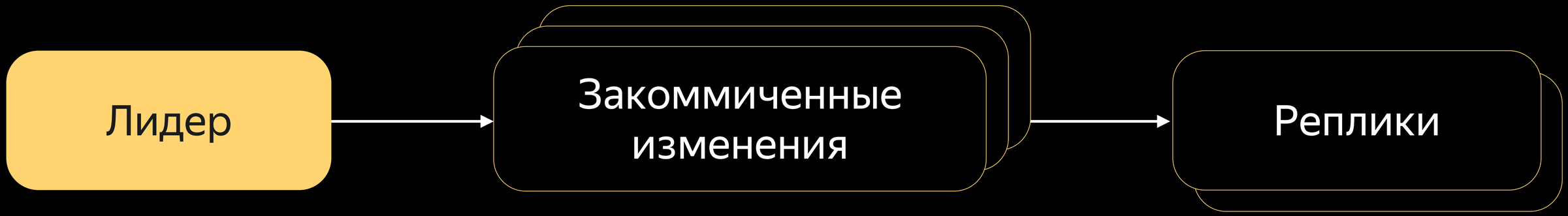
Технически

01 Лидер фиксирует только закоммиченные изменения в лог

02 Реплика читает лог

03 Применяются только коммиты (без «грязных» данных)

04 С реплики читается свежее подтверждённое



Как сжать лог, не сломав восстановление?

Простыми словами

- Я пишу миллионы событий в секунду
- Через неделю — терабайт лога

- Я не хочу, чтобы через месяц мой кластер упёрся в диск
- Я не хочу, чтобы новая реплика запускалась часами

Как чистить лишнее, но не потерять данные для репликации?

Как сжать лог, не сломав восстановление?

Технически

Регулярно создаём снапшоты состояния

После этого удаляем старый лог

Управление — через сборку мусора

Как решить, откуда читать данные?

Простыми словами

Свежесть важна
(баланс, статус заказа)

С лидера: согласовано

Можно немного
устаревшее (лайки,
история, рекомендации)

С ближайшей реплики: быстрее

Как решить, откуда читать данные?

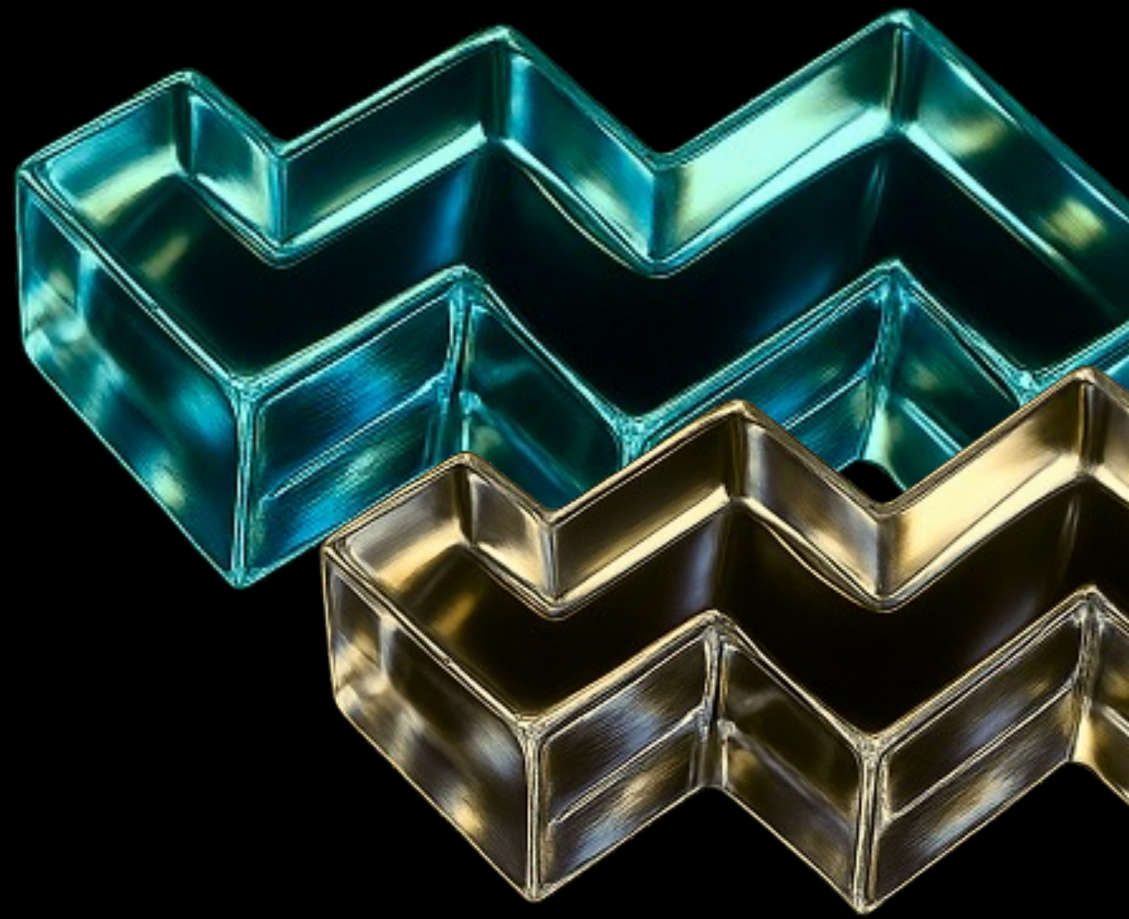
Технически

Режим транзакций Serializable

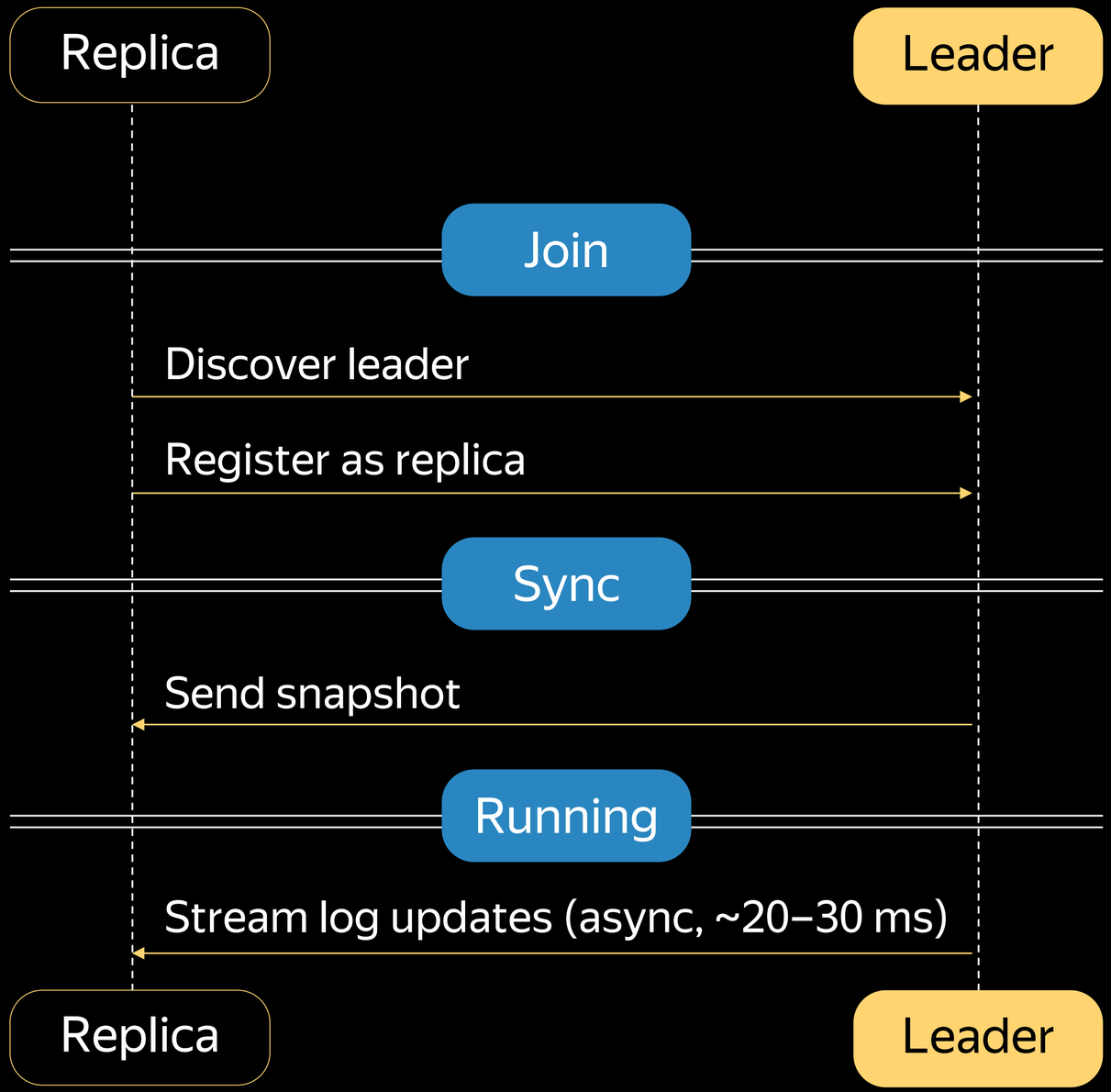
- только с лидера, самый высокий уровень изоляции

Режим транзакций StaleReadOnly

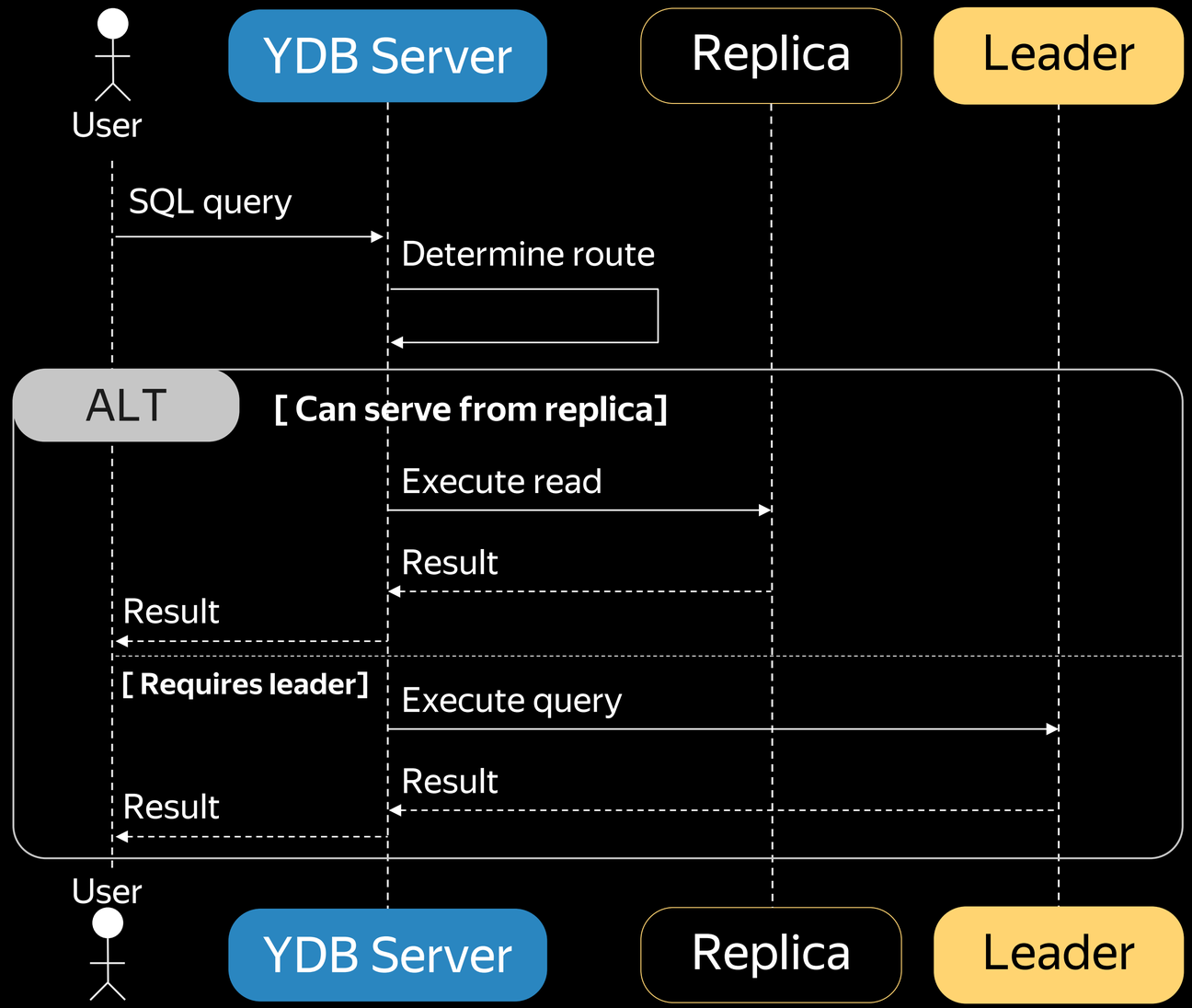
- разрешает чтение с реплики
- автоматический выбор реплики: ближайшей или наименее загруженной



Синхронизация реплики и лидера: схема



Обработка запросов на репликах: схема



03

Поиск и Рекламные технологии

YDB

Включение реплик

Механика работы и настройки

Команда включения:

```
ALTER TABLE table SET READ_REPLICAS_SETTINGS 'PER_AZ:3'
```

PER_AZ:3 — по одной реплике
в каждой зоне доступности / ДЦ

В каждой партиции — минимум
один лидер и указанные реплики

Два вида согласованности

01

Лидер — реплика

В рамках одной партиции

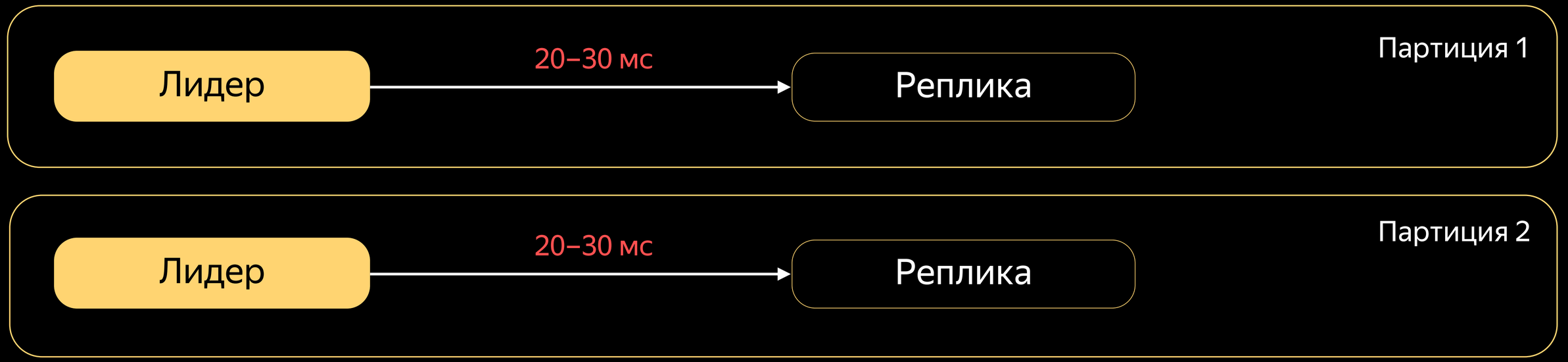
02

Несколько реплик

В рамках разных партиций

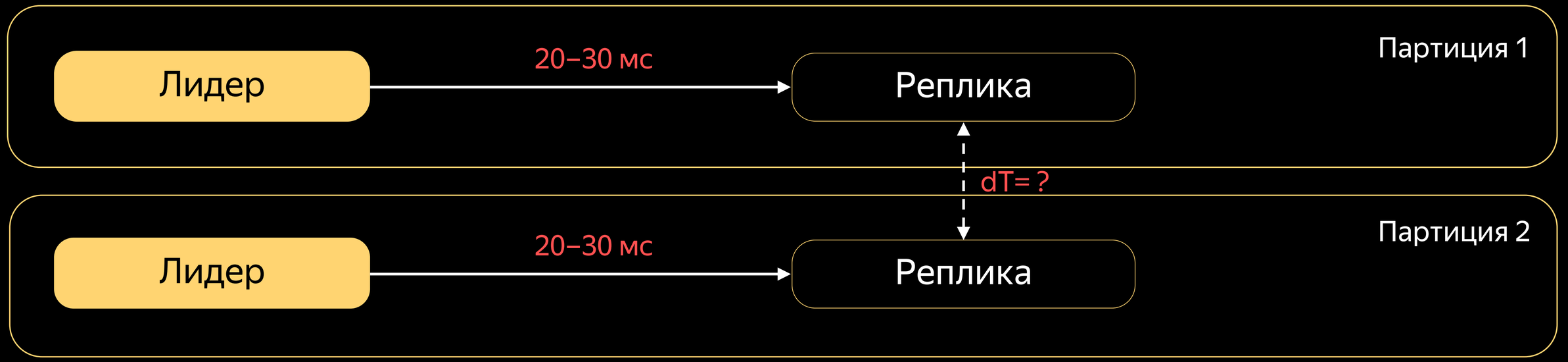
Eventual consistency (согласованность в конечном счёте)

- Чтения могут возвращать слегка устаревшие данные
- Если много реплик — их отставание разное



Согласованность между партициями

Для чтения из нескольких партиций гарантий согласованности нет



Классический пример: перевод денег со счёта на счёт

Какие типы запросов поддерживаются

Чтения из одной
партиции одной таблицы

```
SELECT name FROM users WHERE user_id = 42;
```

Когда чтения пойдут с лидера?

Строгий уровень изоляции (Serializable)

только с лидера

JOIN, аналитика

только с лидера

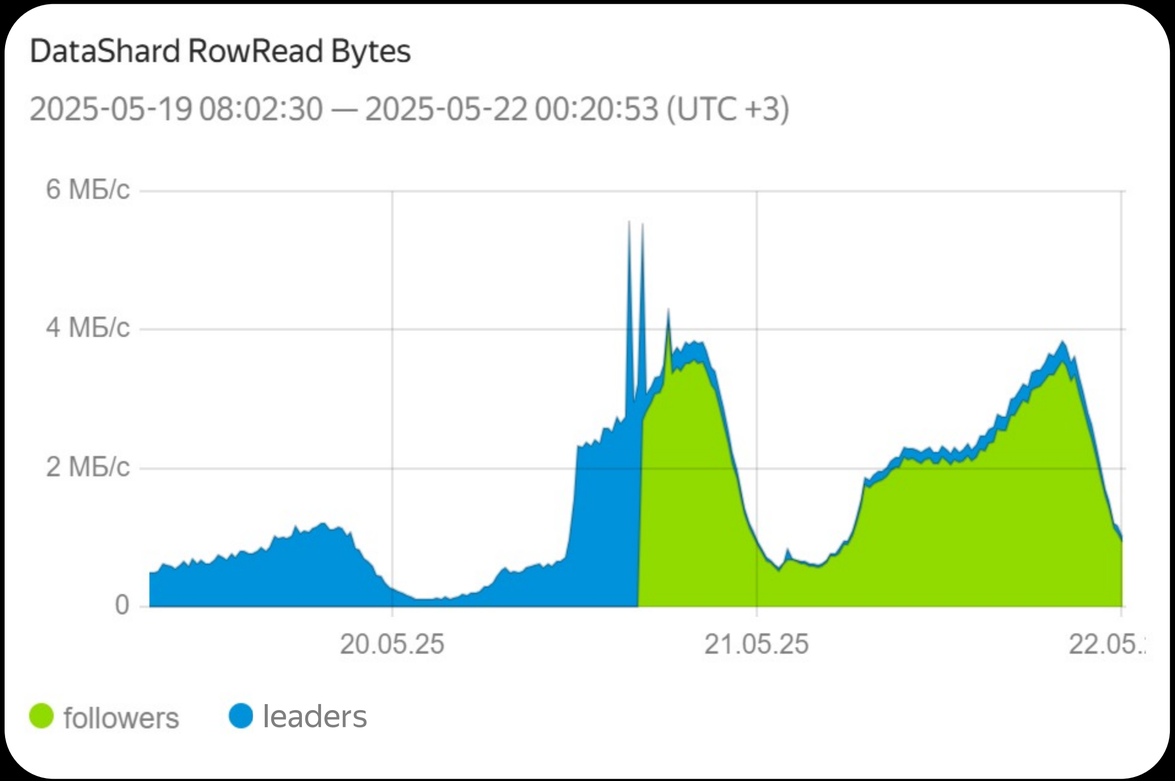
Согласованное чтение нескольких партиций

только с лидера

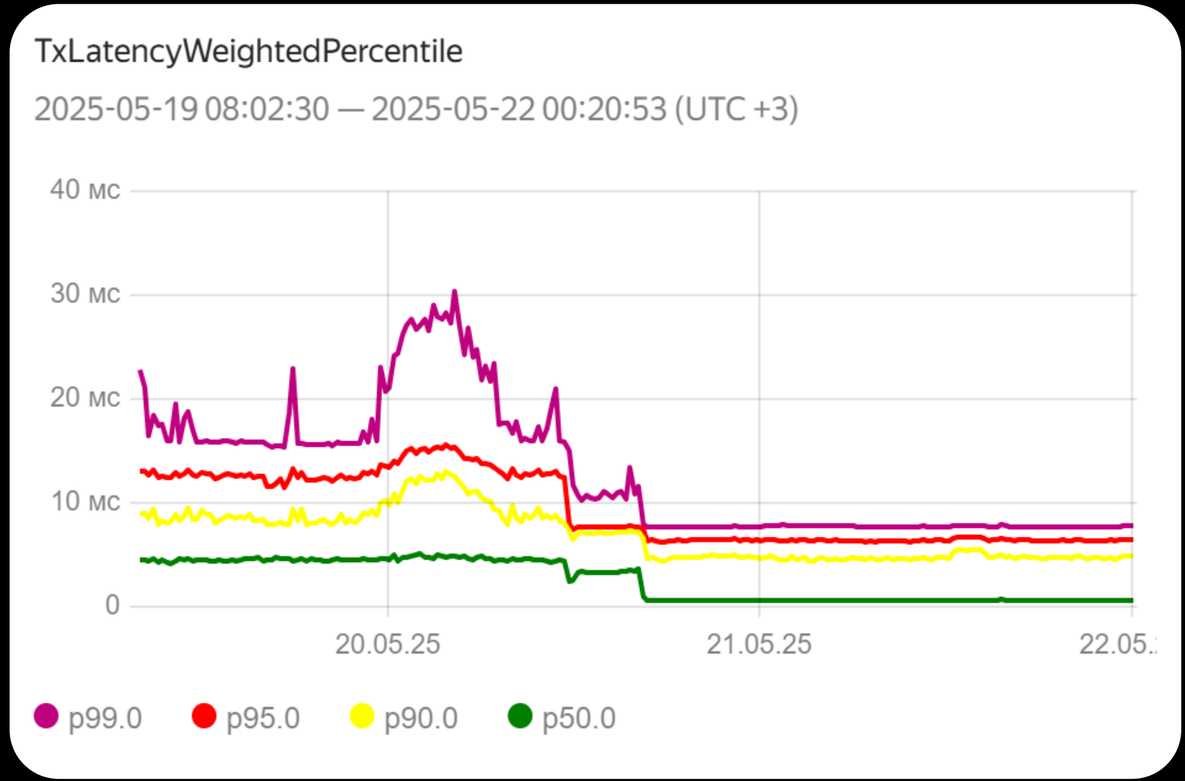
есть планы, как это сделать с реплик

Типичный случай включения реплик

Поток чтения



Время чтения



Рекомендации по использованию реплик

01

Реплики в каждой AZ

02

Мониторьте задержки

03

Не усложняйте запросы

04

Настраивайте алерты

Итого: реплики в YDB

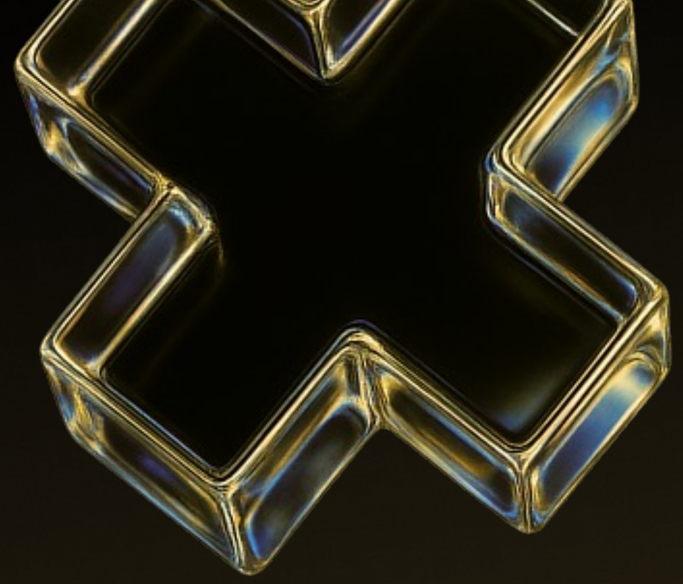
Горизонтальное масштабирование по чтению

Время чтения с реплик: <1мс

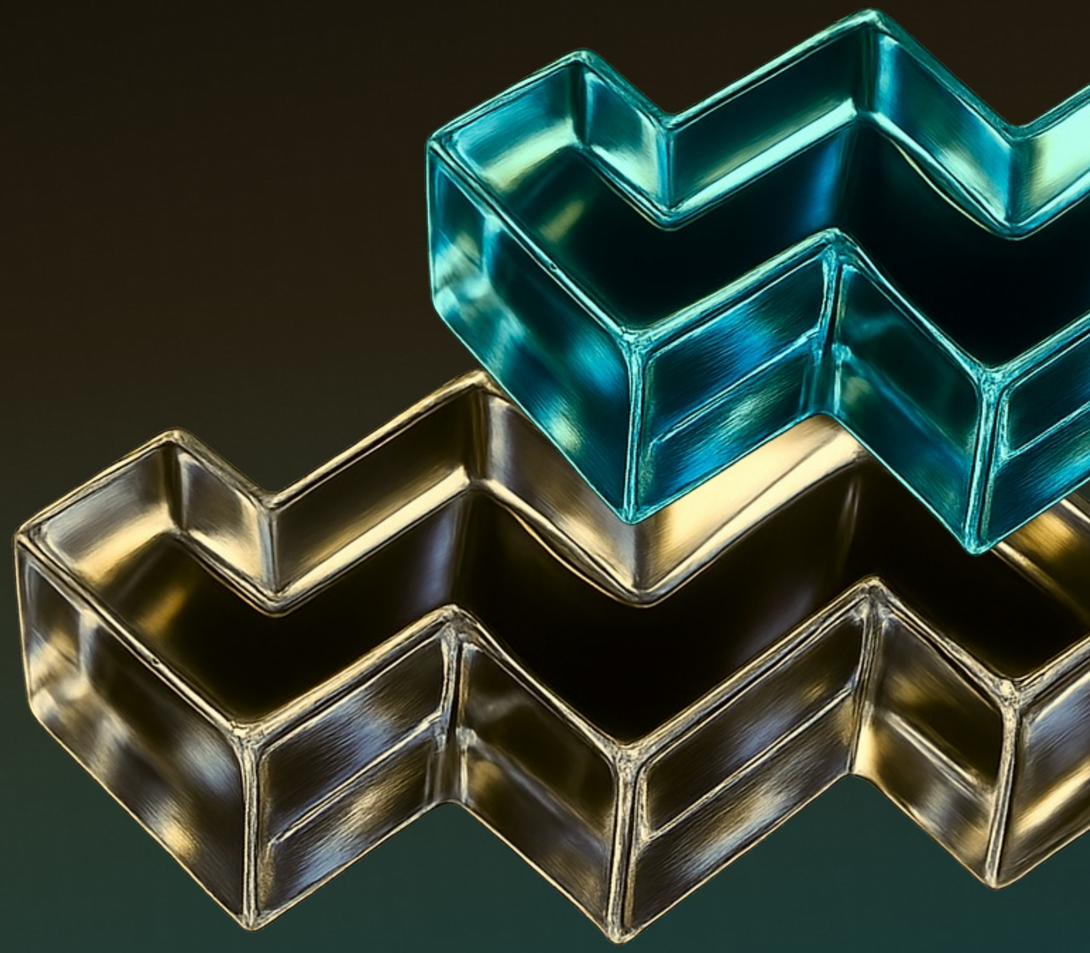
Отставание от лидера: 20–30 мс

Работа в нескольких AZ

Поиск и Рекламные технологии



Спасибо!
Вопросы?



Александр Зевайкин

Руководитель группы разработки, YDB
Кандидат технических наук, доцент