ydb.tech

github.com/ydb-platform/ydb

YDB

Катастрофоустойчивая и высокопроизводительная распределенная СУБД для операционных нагрузок

- \+ Горизонтальное масштабирование
- \+ Транзакции с гарантиями ACID в нескольких AZ
- \+ Работоспособность и автоматическое восстановление при отказах
- \+ Масштабирование на миллионы транзакций в секунду и петабайты данных

КРАТКАЯ ИСТОРИЯ

2014

Инфраструктурный проект

- SQL
- Распределённые транзакции
- Бесконечная масштабируемость
- Катастрофоустойчивость

2017

Yandex Cloud: основа

- База для Control Plane облачных сервисов
- Слой хранения для сетевых дисков

2020

Yandex Cloud: сервис

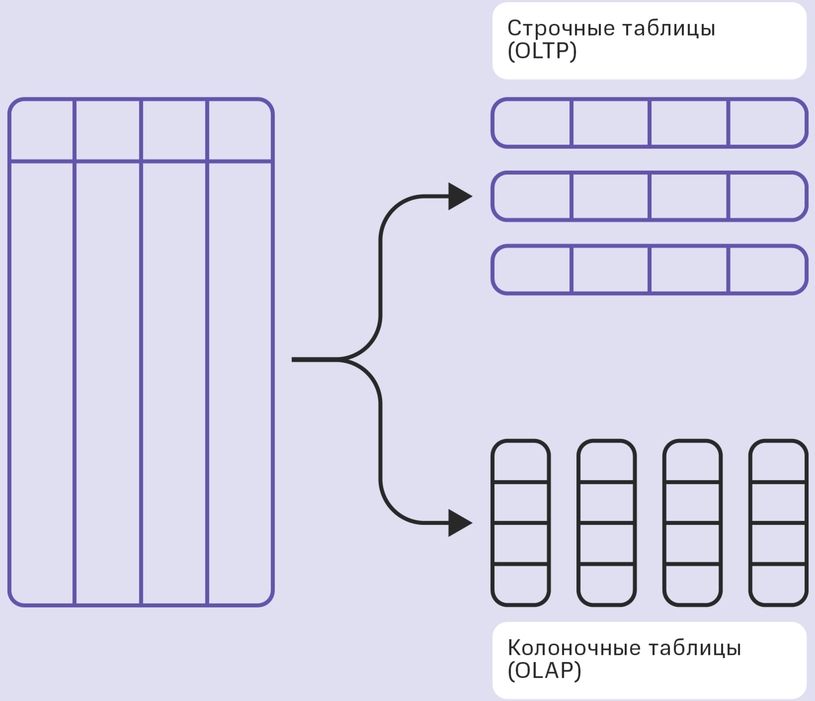
- Serverless или Dedicated
- Совместим с Amazon DynamoDB

2022

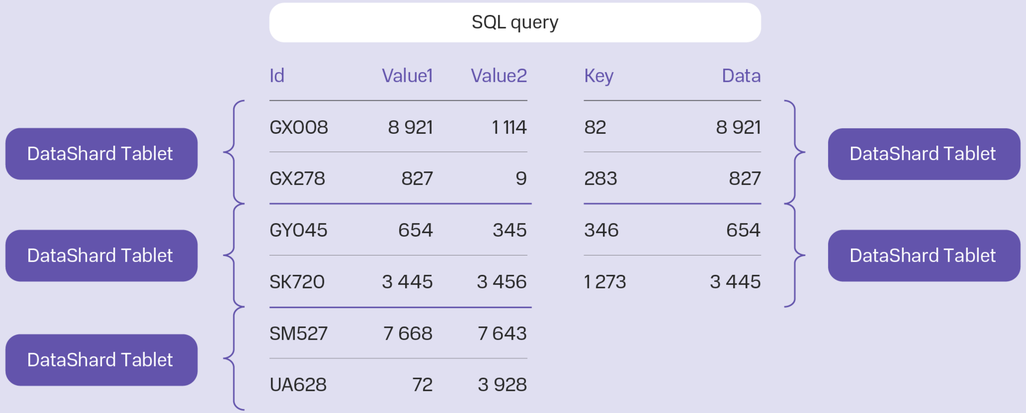
Open Source

- Код ядра полностью открыт (лицензия Apache 2.0)

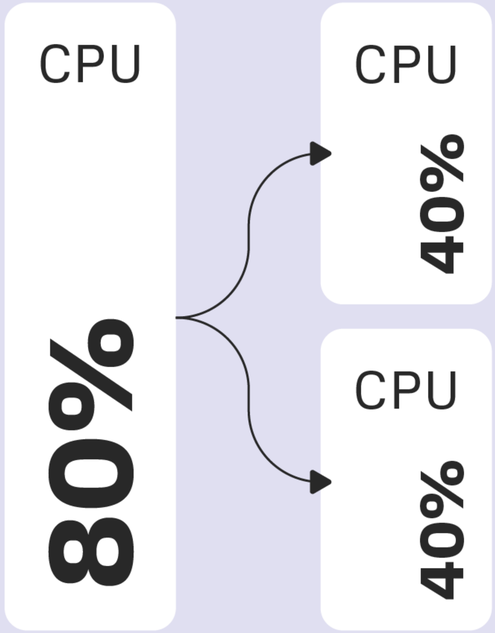
ОСНОВНЫЕ ОСОБЕННОСТИ



ГОРИЗОНТАЛЬНОЕ ПАРТИЦИРОВАНИЕ ТАБЛИЦ

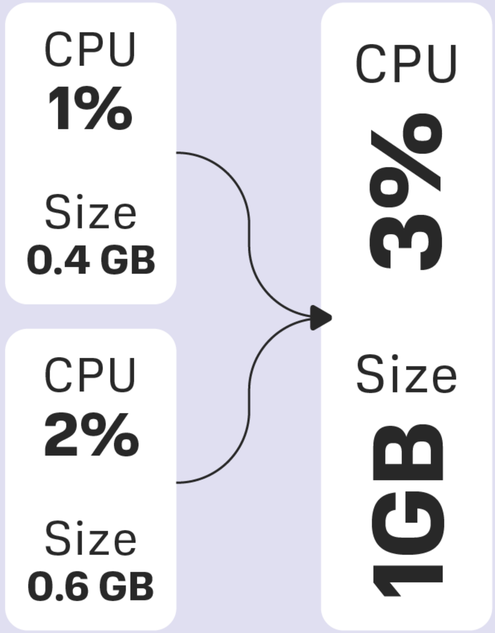


ОСНОВНЫЕ ОСОБЕННОСТИ

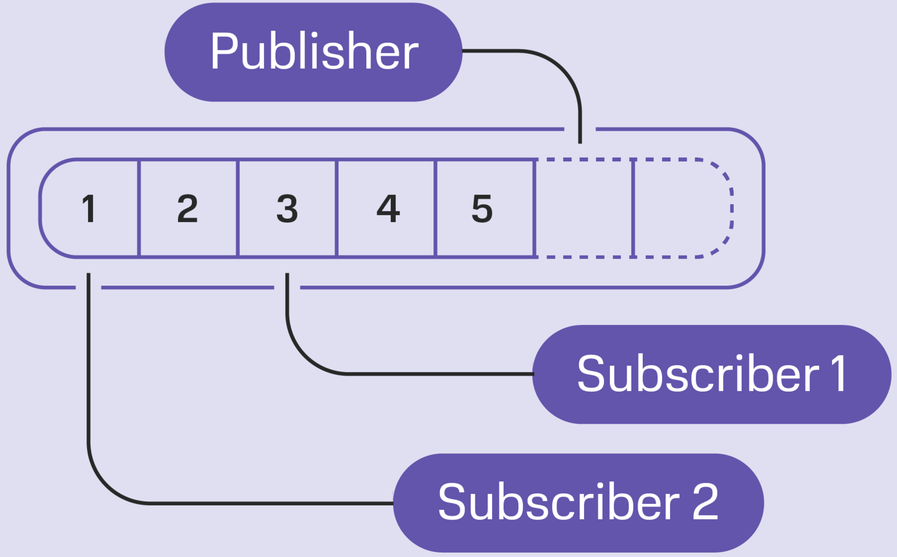


Горизонтальное автоматическое разделение и слияние партиций таблиц (перепартицирование)

Основные критерии: обьем и нагрузка



ОСНОВНЫЕ ОСОБЕННОСТИ

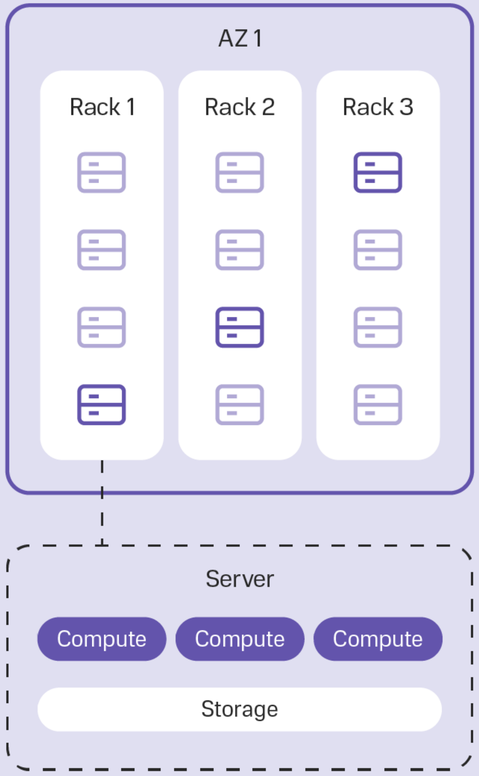


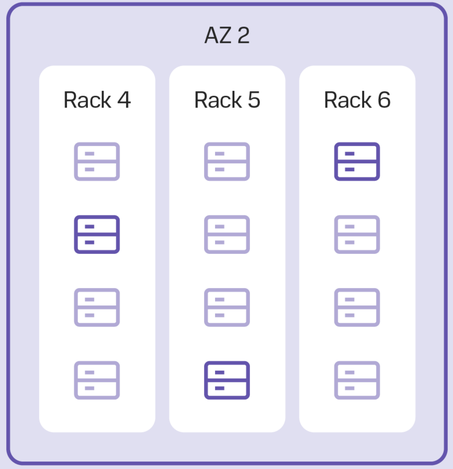
Горизонтально масштабируемые топики для хранения и доставки неструктурированных сообщений множеству подписчиков

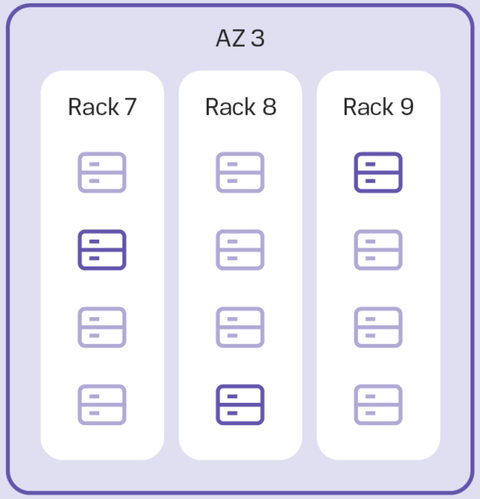
Поддерживаются семантики доставки at-least-once или exactly-once

КАТАСТРОФОУСТОЙЧИВАЯ КОНФИГУРАЦИЯ

Катастрофоустойчивая конфигурация кластеров с синхронной репликацией – переживает потерю зоны доступности и серверной стойки в другой зоне

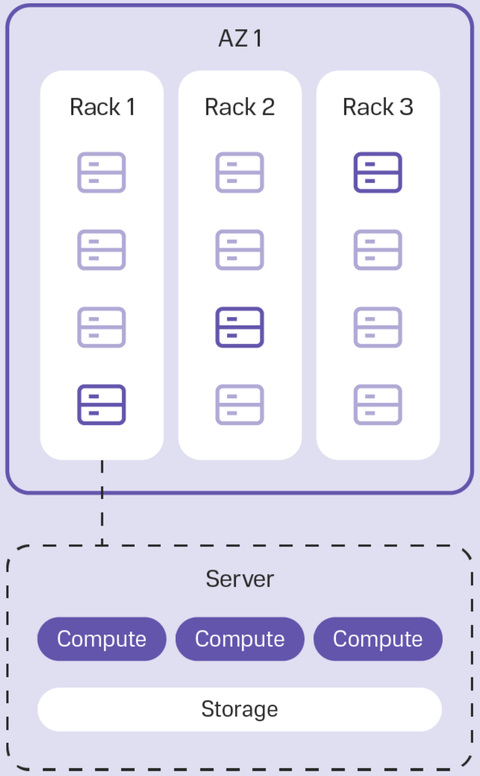


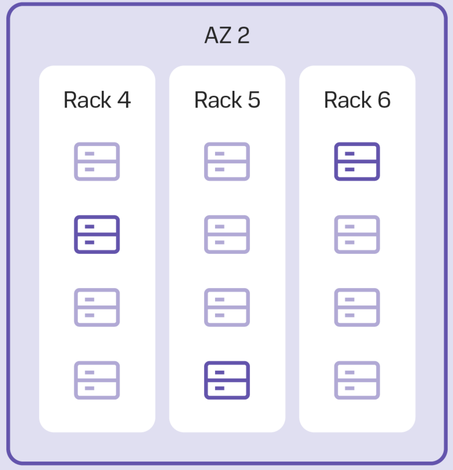


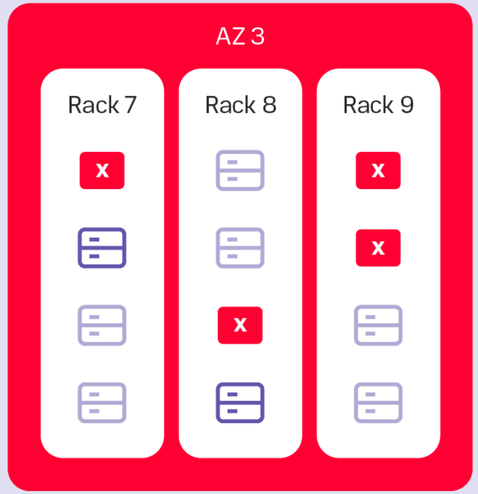


КАТАСТРОФОУСТОЙЧИВАЯ КОНФИГУРАЦИЯ

Катастрофоустойчивая конфигурация кластеров с синхронной репликацией – переживает потерю зоны доступности и серверной стойки в другой зоне



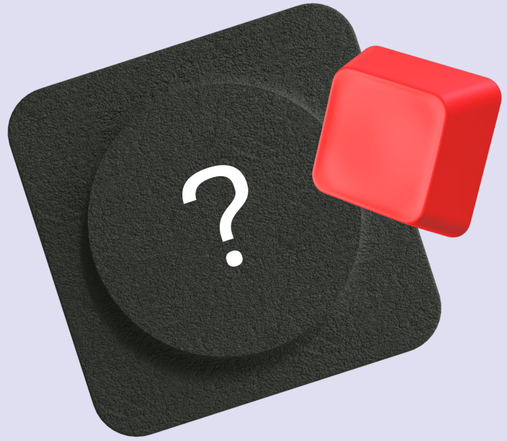




Задержки между зонами доступности в этой конфигурации должны быть достаточно низкими – и это накладывает определенные ограничения

АСИНХРОННАЯ РЕПЛИКАЦИЯ В YDB КАК РАСПРЕДЕЛЕННОЙ СИСТЕМЕ – НЕТРИВИАЛЬНА

В каких случаях она могла бы понадобиться?



Асинхронная реплика
для катастрофоустойчивости

- Случайные инциденты
- Намеренные атаки
- Техногенные катастрофы

АСИНХРОННАЯ РЕПЛИКАЦИЯ – НАДЕЖНЫЙ И ГИБКИЙ ИНСТРУМЕНТ ПОД ЗАДАЧИ ЛЮБОГО МАСШТАБА

Асинхронная реплика для катастрофоустойчивости

- Случайные инциденты
- Намеренные атаки
- Техногенные катастрофы

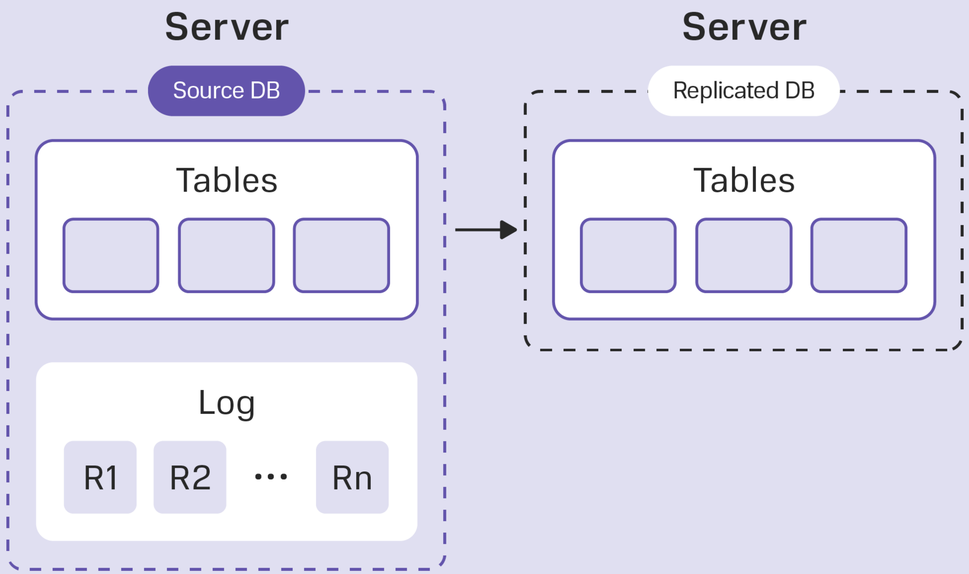
АСИНХРОННАЯ РЕПЛИКАЦИЯ – НАДЕЖНЫЙ И ГИБКИЙ ИНСТРУМЕНТ ПОД ЗАДАЧИ ЛЮБОГО МАСШТАБА

Выполнение разнообразных рабочих сценариев

- Региональные кластеры
- Комплаенс
- Отдельные кластеры для OLAP-нагрузки

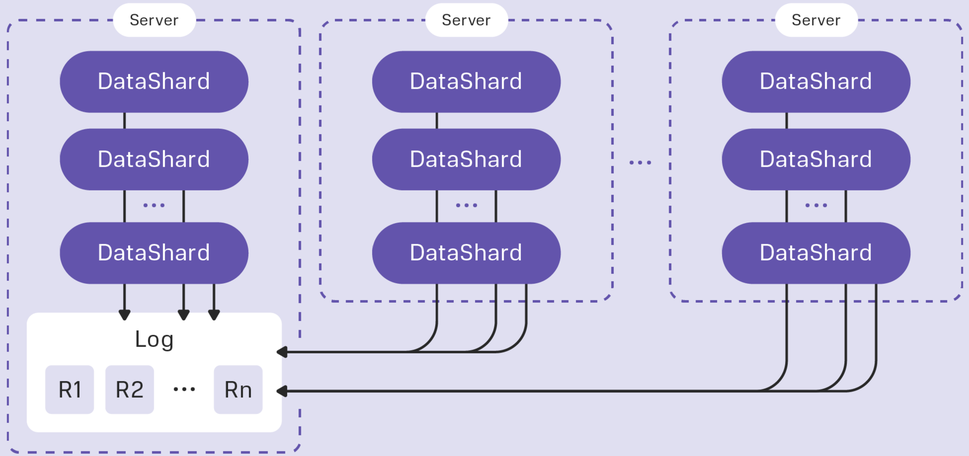
РАЗРАБОТКА АСИНХРОННОЙ РЕПЛИКАЦИИ В YDB ШАГ ЗА ШАГОМ

КАК ВЫГЛЯДИТ ТРАДИЦИОННЫЙ ВАРИАНТ?



База данных размещена
на одном сервере,
в ней много таблиц,
но только один лог

ПОЧЕМУ ТРАДИЦИОННЫЙ ВАРИАНТ НЕ ПОДХОДИТ ДЛЯ РАСПРЕДЕЛЕННОЙ БД?

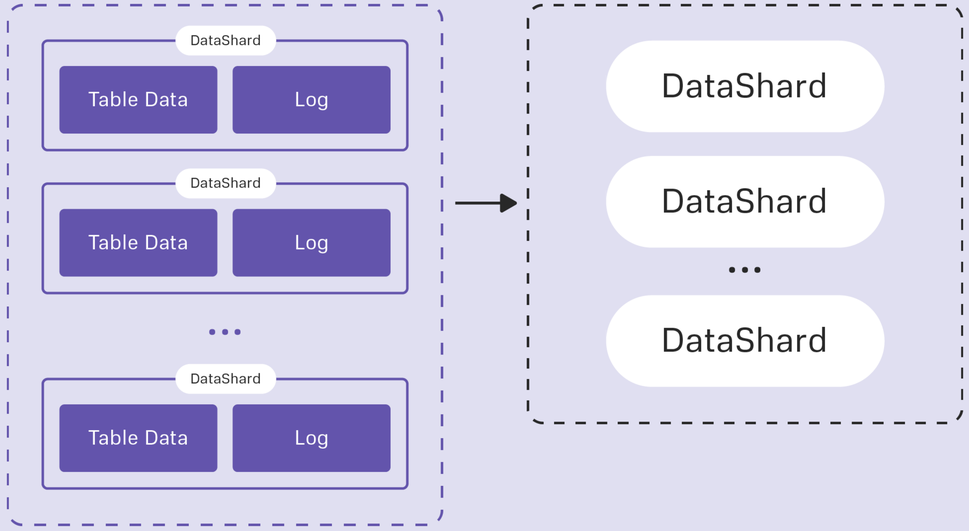


У лога структура (FIFO)

Пропускная способность одного лога ограничена

Нам нужно много логов

ИСПОЛЬЗУЕМ ЛОГИ ПАРТИЦИЙ

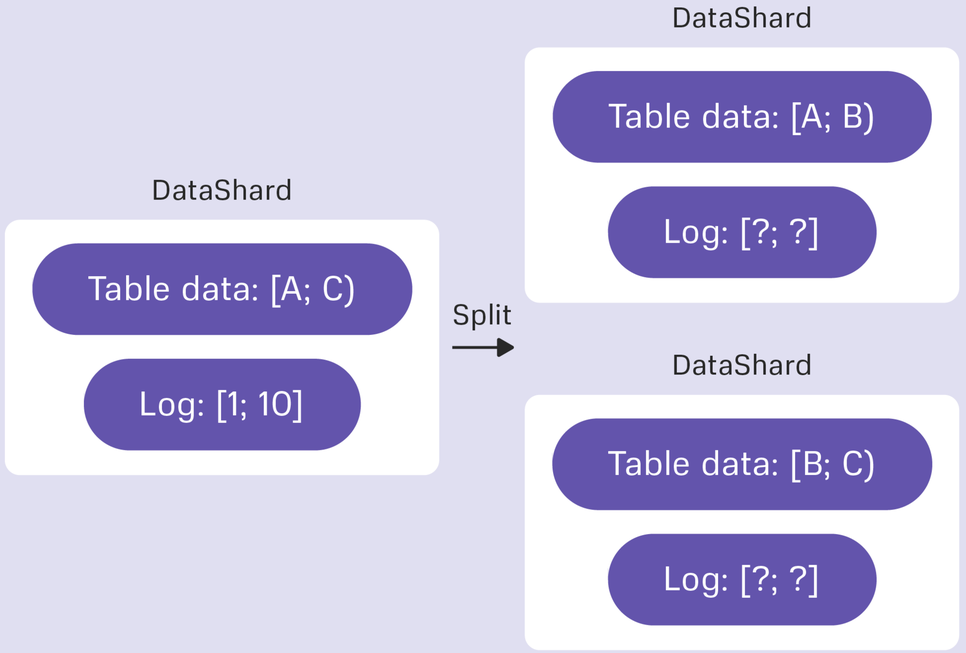


У каждой партиции таблицы есть свой лог

Обычно он маленький

Но может расти при нарушениях сетевой связности

РОСТ
ЛОГА ПАРТИЦИЙ
ВЫЗЫВАЕТ
ПРОБЛЕМЫ

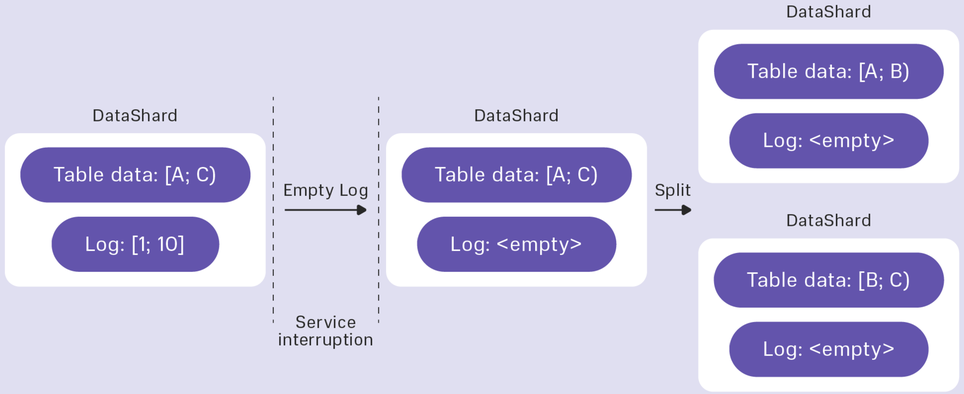


Таблицы отсортированы по PK

Лог отсортирован в порядке поступления изменений

Разделение лога при разделении партиции может быть не тривиальным

РОСТ
ЛОГА ПАРТИЦИЙ
ВЫЗЫВАЕТ
ПРОБЛЕМЫ



Вместо разделения лога можно его очистить перед разделением

Это вызовет перерыв в обслуживании, зависящий от размера лога

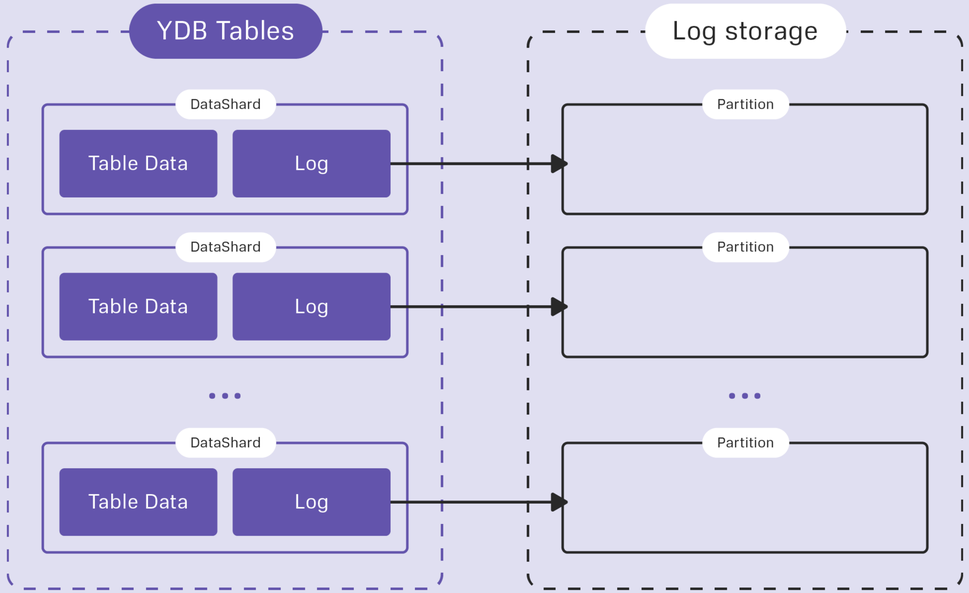
ИСПОЛЬЗОВАТЬ ЛОГИ ПАРТИЦИЙ — СЛОЖНО

Партиций может быть очень много и у каждой свой лог

Из-за проблем связности эти логи могут расти

Большой лог может вызвать длительный перерыв в обслуживании при разделении партиции таблицы

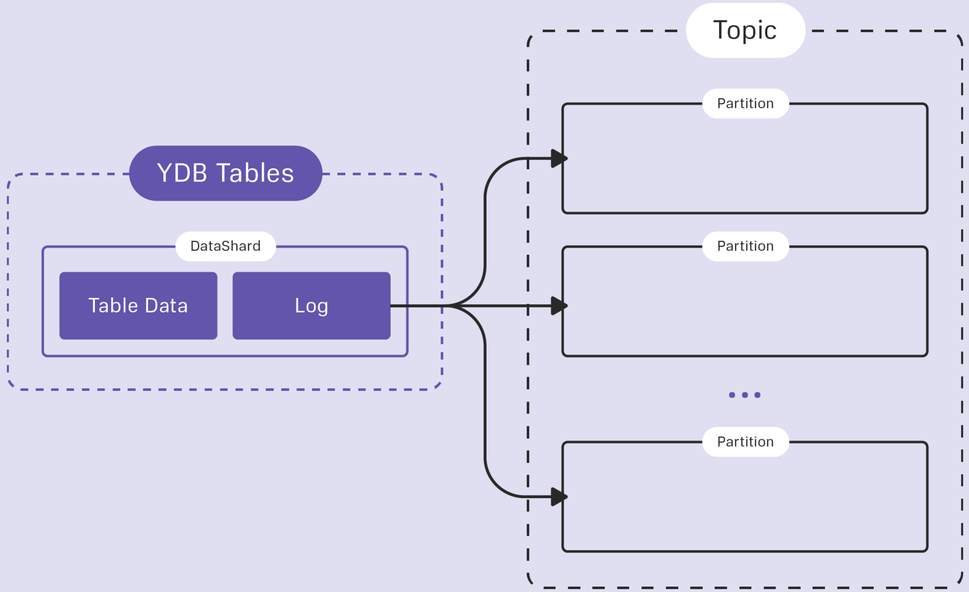
КАК МЫ МОЖЕМ ПЕРЕИСПОЛЬЗОВАТЬ ПРОВЕРЕННЫЕ ТЕХНОЛОГИИ – ТОПИКИ?



Логи партиций таблиц могут остаться маленькими

Партиции топика предназначены для долговременного хранения лога

СООТНОШЕНИЕ КОЛИЧЕСТВА ПАРТИЦИЙ ТАБЛИЦЫ И ТОПИКА МОЖЕТ БЫТЬ РАЗНЫМ...

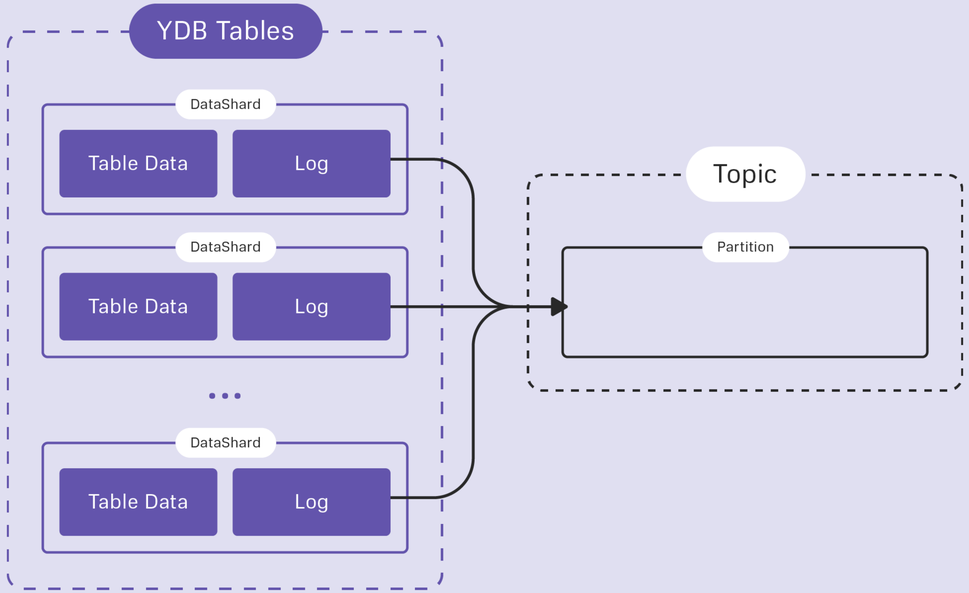


Партиции таблицы могут генерировать большой лог

Например при частом обновлении ключей

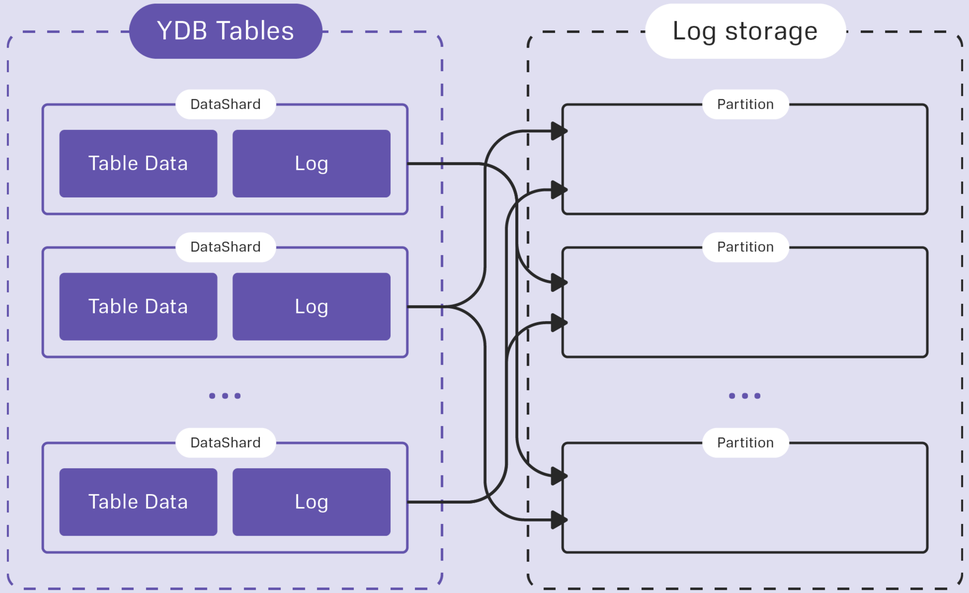
Для хранения такого лога используется много партиций топика

СООТНОШЕНИЕ КОЛИЧЕСТВА ПАРТИЦИЙ ТАБЛИЦЫ И ТОПИКА МОЖЕТ БЫТЬ РАЗНЫМ...



Если генерируется маленький лог, то требуется одна партиция топика на несколько партиций таблиц

ОТНОШЕНИЕ ДАТАШАРДОВ И ПАРТИЦИЙ

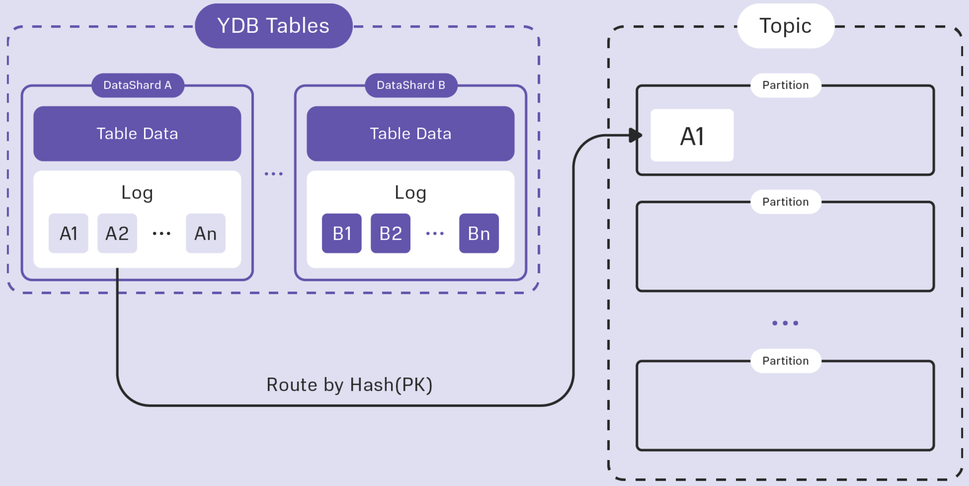


Random

N:M (1:1, 1:M, N:1)

Consistent hashing

ЗАПИСЬ В ТОПИКИ



ПРОБЛЕМЫ ДОСТАВКИ

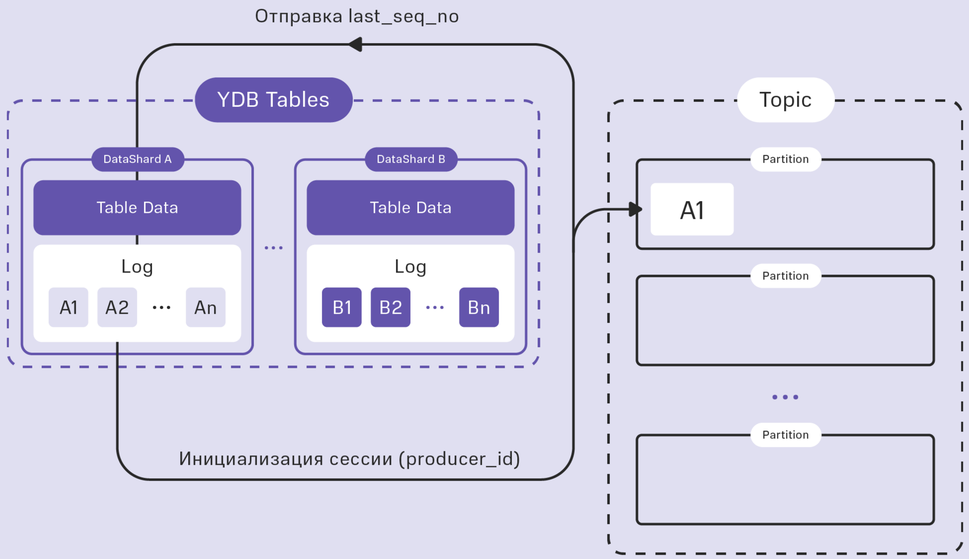
ТАБЛЕТКИ МОГУТ РЕСТАРТОВАТЬ ИЗ-ЗА:

- Обновление кластера
- Отказы оборудования
- Балансировка ресурсов
- Проблемы сетевой связности

ПОСЛЕДСТВИЯ

- Дубликаты
- Рост лога партиций таблиц

КАК УСТРАНИТЬ ПРОБЛЕМЫ ДОСТАВКИ?

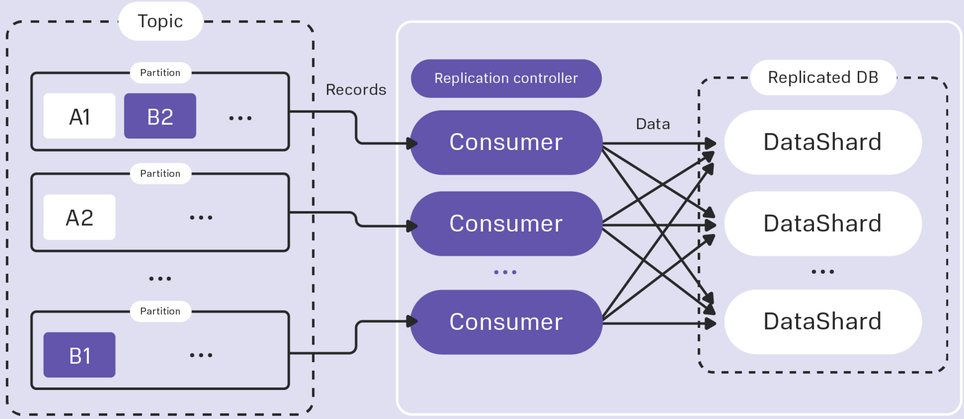


Каждая партиция таблицы (producer) идентифицируется своим producer_id

Каждая запись лога каждой партиции таблицы идентифицируется монотонно возрастающим порядковым номером seq_no

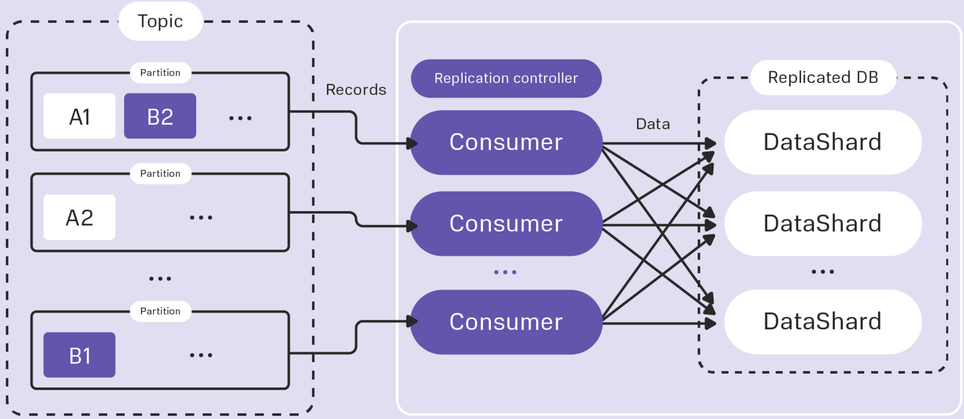
Пара (producer_id, seq_no) позволяет дедуплицировать записи и обеспечить семантику exactly_once

КАК ОБЕСПЕЧИТЬ ЗАПИСЬ ДАННЫХ В ЦЕЛЕВОЙ КЛАСТЕР?



Компонент Consumer вычитывает данные и записывает их в даташарды

КАК ОБЕСПЕЧИТЬ ЗАПИСЬ ДАННЫХ В ЦЕЛЕВОЙ КЛАСТЕР?



Компонент
Replication controller

Создает consumer для каждой партиции топика

Оркестрирует процесс репликации, в частности:

- нагрузку и ее равномерность
- пропускную способность топиков

КАК РЕАЛИЗОВАТЬ РАСПРЕДЕЛЕННЫЕ ТРАНЗАКЦИИ?

2PC

Наиболее стандартный вариант

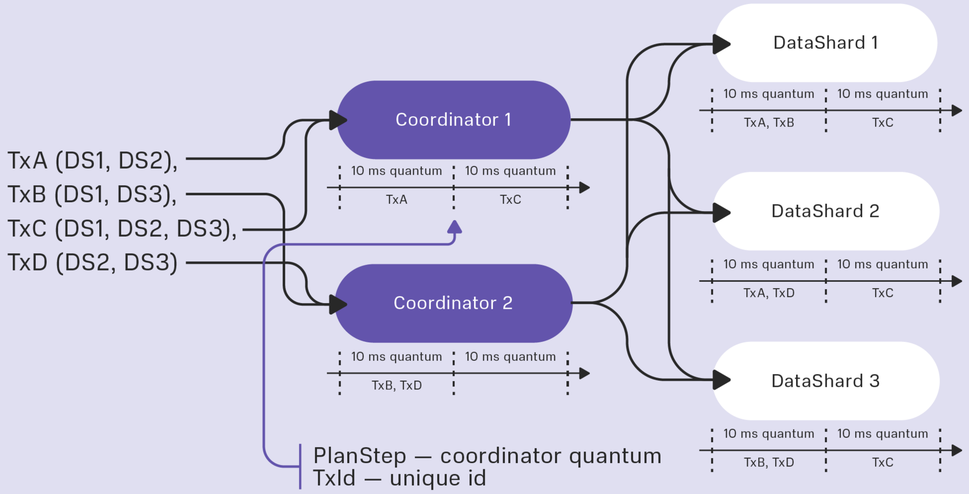
Низкая пропускная способность при высокой конкуренции

Calvin

Позволяет выполнять детерминированные транзакции без пессимистических блокировок

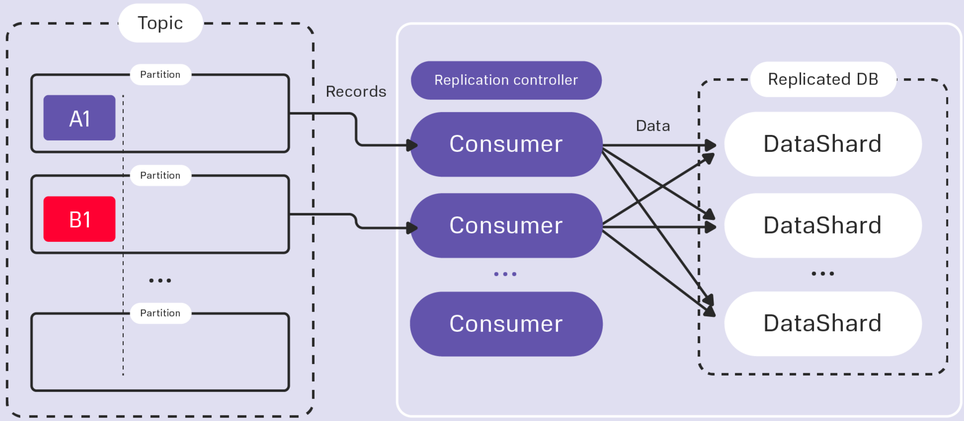
Транзакции YDB — это расширение Calvin

КАК РЕАЛИЗОВАТЬ РАСПРЕДЕЛЕННЫЕ ТРАНЗАКЦИИ?



Метаданные транзакции — (TxId и PlanStep)

КАК ОБЕСПЕЧИТЬ ГЛОБАЛЬНУЮ УПОРЯДОЧЕННОСТЬ И КОНСИСТЕНТНОСТЬ?

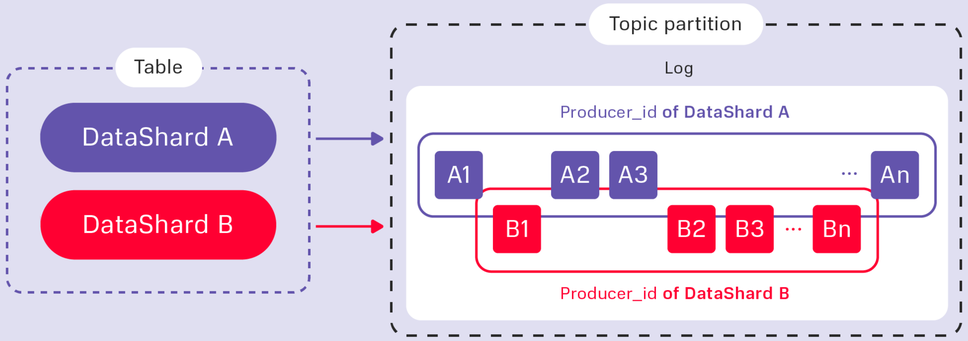


Временные метки позволяют на стороне реплицированной базы восстановить ход событий

Но для консистентности реплицированной базе еще нужно удостовериться, что получены точно все данные

Для этого необходимо знать список даташардов (producer_id) на стороне исходной базы

КАК ОБЕСПЕЧИТЬ ГЛОБАЛЬНУЮ УПОРЯДОЧЕННОСТЬ И КОНСИСТЕНТНОСТЬ?

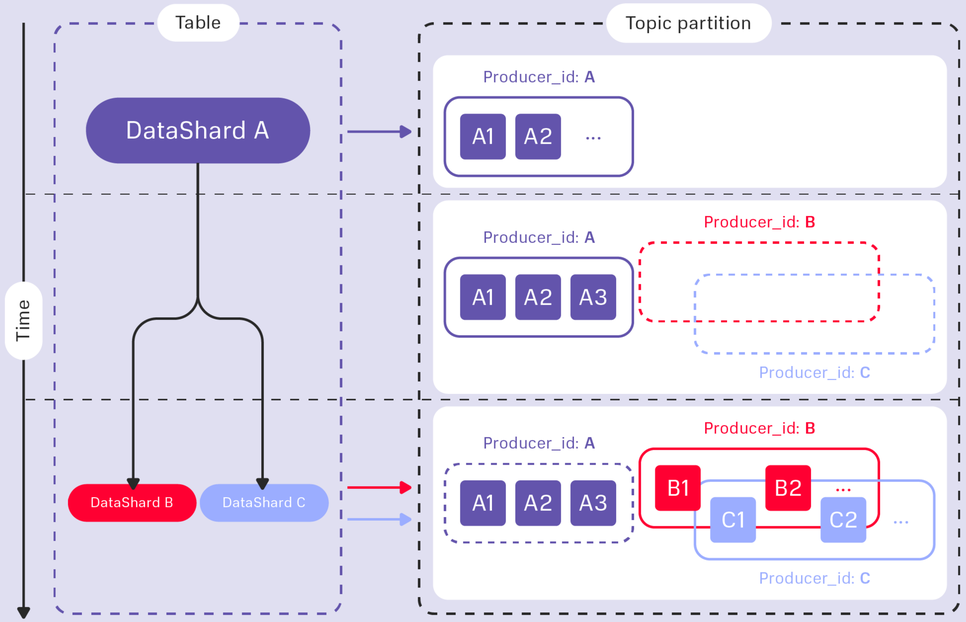


Партиция топика содержит записи всех партиций таблиц (продюсеров)

Каждая партиция таблицы идентифицируется producer_id

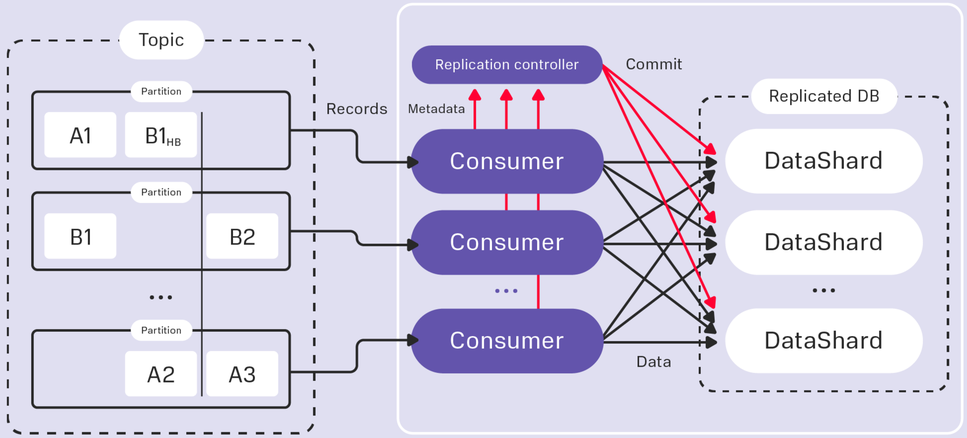
Партиция топика знает список всех продюсеров в любой момент времени

КАК ОБЕСПЕЧИТЬ ГЛОБАЛЬНУЮ УПОРЯДОЧЕННОСТЬ И КОНСИСТЕНТНОСТЬ?



Данные остаются актуальными в любой момент времени, в частности, после операций split/merge с даташардами

КАК ДОБИТЬСЯ ГЛОБАЛЬНОЙ УПОРЯДОЧЕННОСТИ И КОНСИСТЕНТНОСТИ?

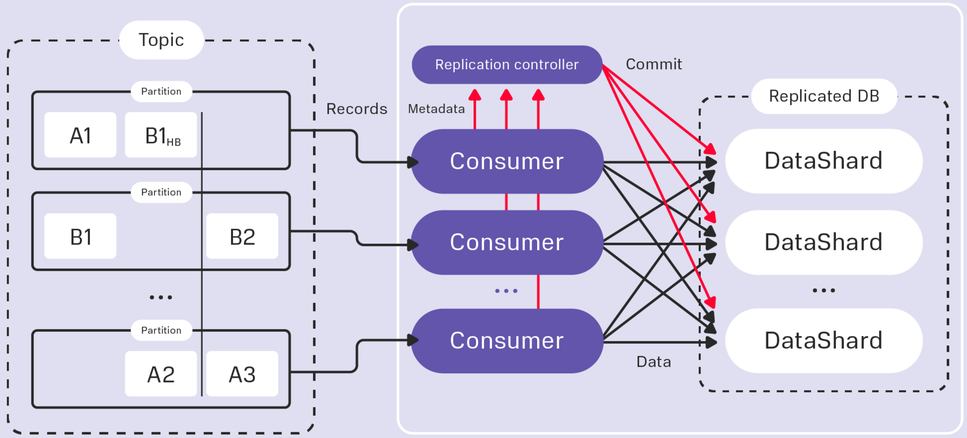


Данные от даташарда могут не поступить

В этом случае даташардом отправляется специальная «заглушка» – heartbeat

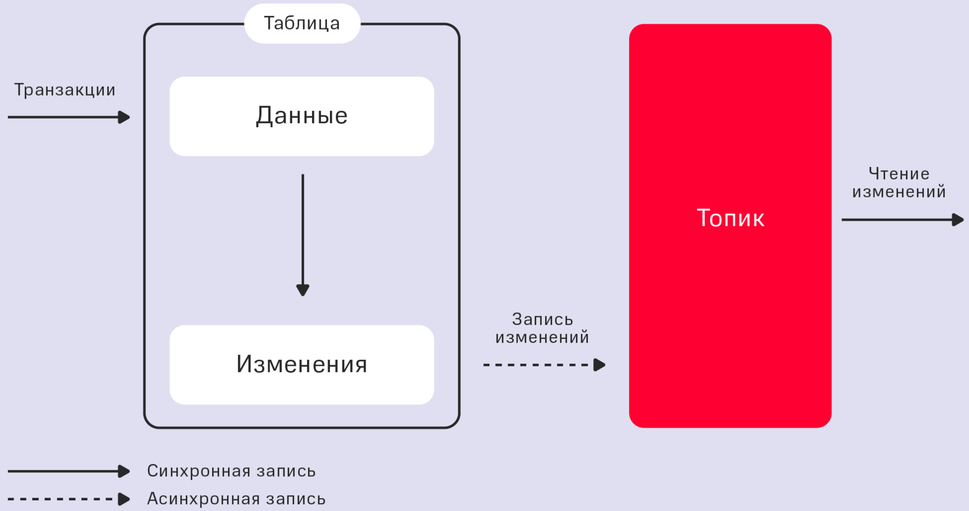
Это дает возможность без задержек продвигаться от одной отсечки (PlanStep) к другой и коммитить данные в реплики

КАК ДОБИТЬСЯ ГЛОБАЛЬНОЙ УПОРЯДОЧЕННОСТИ И КОНСИСТЕНТНОСТИ?



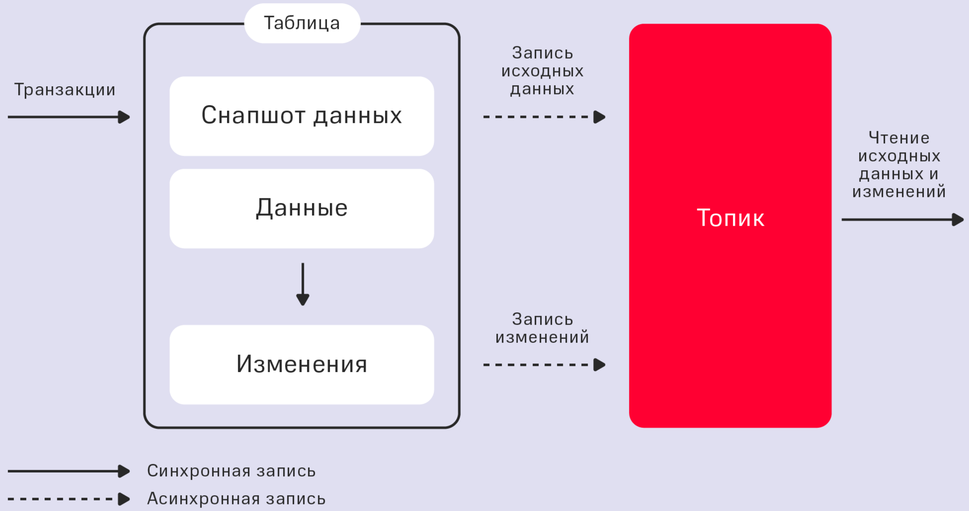
Соответственно,
в рамках каждого «среза» (PlanStep) данные для реплицированной базы остаются актуальными – что обеспечивает глобальную консистентность

АСИНХРОННАЯ РЕПЛИКАЦИЯ БАЗИРУЕТСЯ НА МЕХАНИЗМЕ CDC



В CDC мы получаем только набор изменений без первоначального состояния

НУЖНО ИМЕТЬ ВОЗМОЖНОСТЬ В ЛЮБОЙ МОМЕНТ ИЗМЕНИТЬ ИЛИ ЗАПУСТИТЬ РЕПЛИКАЦИЮ



Initial Scan позволяет отгрузить первоначальное состояние строк

- Первоначальное состояние строк отгружается в топик вместе с изменениями
- Гарантия порядка: сначала исходное значение строки, потом изменения

ВЫВОДЫ

УПРАВЛЯЕМЫЙ СЕРВИС YDB В YANDEX CLOUD

- Полностью управляемый сервис Yandex Cloud
- Нет необходимости в собственной инфраструктуре
- Не требуется CAPEX
- Поддержка режима бессерверных вычислений
- Легкий деплой для тестирования

YDB + Yandex Cloud

Эффективная гибридная платформа для эксплуатации и непрерывного развития систем

- **Production** — on-premise
- **Test&Dev** — публичное облако

Серверы

Виртуальные машины

В РАМКАХ СОБСТВЕННОЙ ИНФРАСТРУКТУРЫ

Kubernetes®

Программно-аппаратный комплекс

Open Source

Поддержка от вендора

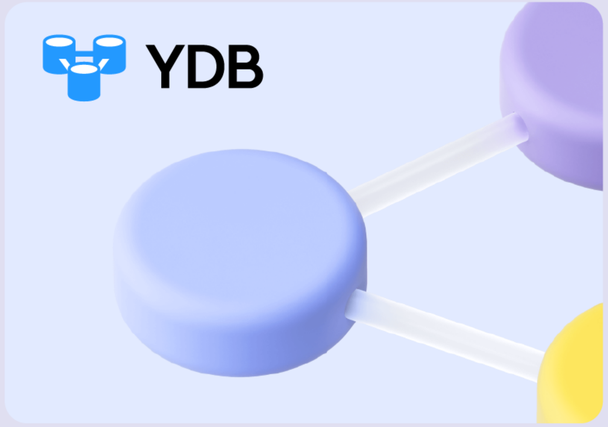
ПОДДЕРЖКА YDB ON-PREMISE

Базовая поддержка

Режим 8/5

Расширенная поддержка

Режим 24/7



ydb.tech

СПАСИБО!