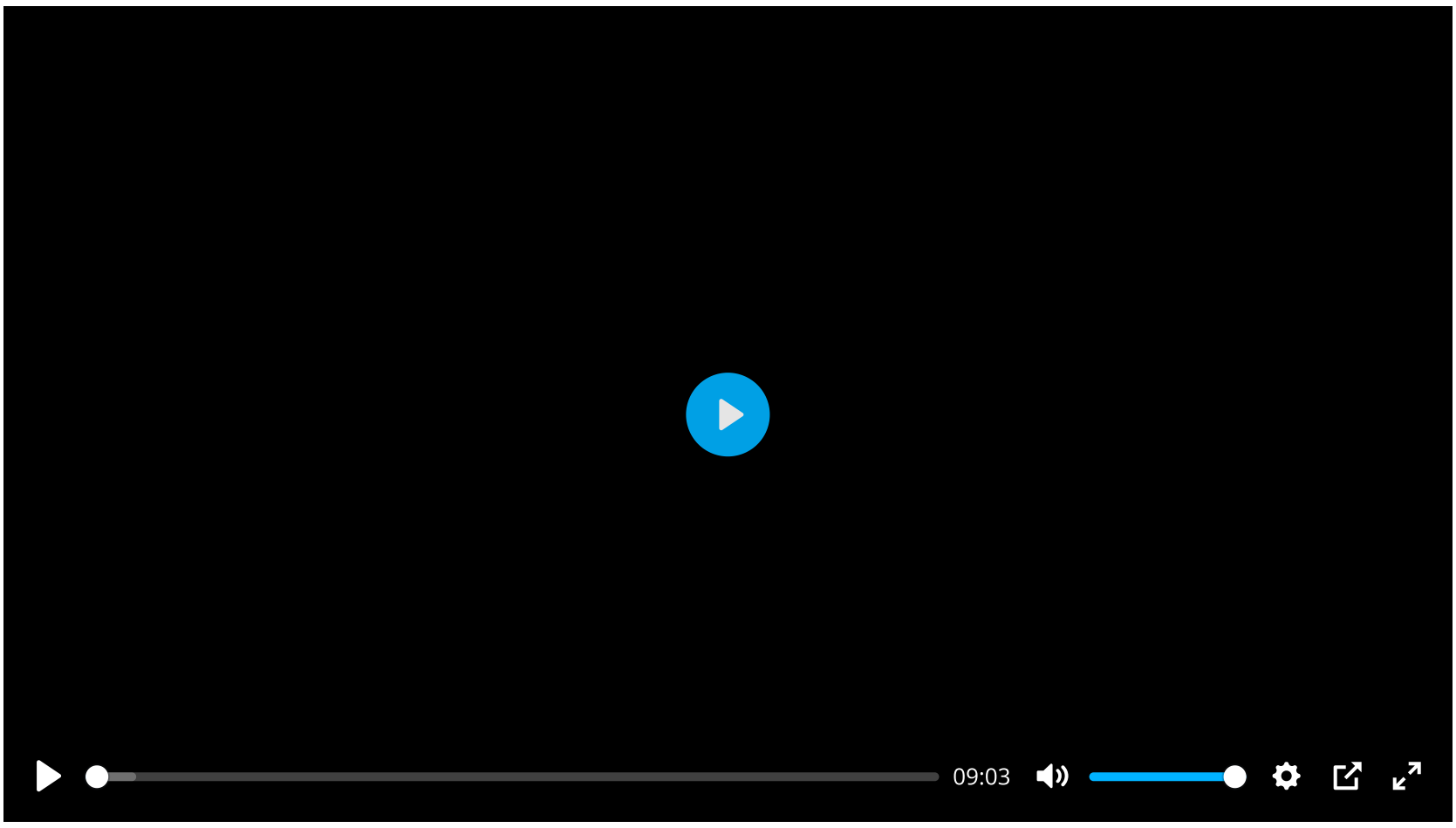
5.5 ORM, события и история



Текстовая расшифровка видео:

ЛОГИ И ИЗМЕНЕНИЯ

- Кейс: переложить данные из OLTP в OLAP;
- Мы ждем перемен;
- Change Data Capture;
- Примеры систем.

Кейс: переложить данные из OLTP в OLAP

Представим, что у нас есть продуктовая OLTP БД, и мы хотим переложить данные в OLAP.

Как это сделать:

- **Запросом** доставать **всю базу целиком** или явно развешивать на все записи **таймстемпы** и выбирать по ним.
- Это будет либо **пачка сложных SQL-запросов**, либо какая-то сложная, возможно, **проприетарная система**.
- Нам придется **регулярно нагружать** продуктовую базу **сложными и тяжелыми** запросами.

На сегодняшний день это не очень рекомендуется.

Мы ждем перемен

Мы можем воспользоваться классным свойством реляционных БД. В современных базах существует такое понятие, как «Транзакционный лог». Мы можем сделать систему, которая будет состоять трех компонентов:

- **Change Detection** – отслеживание изменений;
- **Change Capture** – захват изменений;
- **Change Propagation** – прокидывание изменений.

Change Data Capture

Рассмотрим, каким образом это может работать.

Если мы хотим отслеживать изменения, то можем:

- Отслеживать UPDATED_DATE;
- Выбирать по timestamp’ам;
- Развешивать триггеры на все row-level операции;
- Читать лог транзакций базы в реальном времени.

Транзакционный лог на сегодняшний день наиболее популярен.

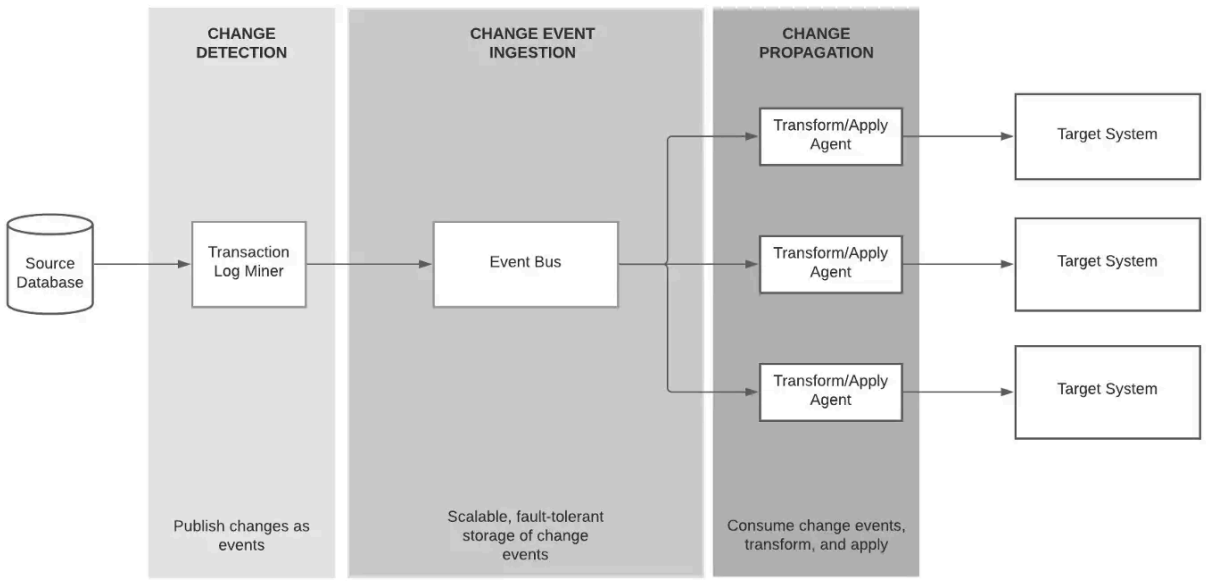
Когда мы читаем лог транзакций, мы не оказываем никакого дополнительного давления на базу. Если мы развешиваем триггеры, то это ощутимо влияет на скорость запросов и эффективность распределения загрузки. Чтение же транзакционного лога не создает дополнительной нагрузки.

Ряд требований к подобной системе:

- Строгий порядок доставки;
- Работа в асинхронном режиме;
- Семантика at least once; нельзя терять сообщения;

Это достаточно серьезный набор требований, реализовать на практике его весьма нетривиально.

Рассмотрим схему подобного проекта:



Подобные проекты работают похожим образом. Разные современные коммерческие решения БД (Mssql, Oracle) обладают встроенными решениями для организации Change Data Capture, которые позволяют пересылать изменения куда-либо.

На примере видим три стадии, которые мы описывали:

- **Change Detection:** видим изменения в транзакционном логе.
- **Change Event:** сериализуем его;
- **Change Propagation:** пересылаем его в шину, в целевую систему.

Примеры систем

Debezium – это open-source’ное решение. Его можно использовать в паре с MySQL, PostgreSQL и т.д. Оно представляет изменения в виде JSON’ов, которые можно кидать дальше в шину, Kafka и т.д.

Пример пайплайна MySQL -> PostgreSQL

Maxwell – это решение для Change Data Capture с забавным логотипом:

https://maxwells-daemon.io/

Совет:

Если вам действительно важно иметь минимальную нагрузку на продуктовую базу, то мы рекомендуем поговорить с DBL, которые администрируют вашу продуктовую базу, и настроить механизм Change Data Capture, чтобы не мешать друг другу при получении выгрузок данных в аналитический контур.

Существуют нюансы синхронизации изменений в структуре таблиц, так как изменения не всегда нормально отлавливаются подобными системами.

Как вам урок?

Изучил, далее