

Прислоняться разрешается.

Как мы подружили MSSQL, PostgreSQL, написали свою «репликацию» и переводим на PostgreSQL одну из самых больших МИС в России



Тарас Чикин

Главный Архитектор ПО

Единая цифровая платформа в здравоохранении сегодня

Более **17 лет**
разработки

Мобильные приложения
для **врача и пациента**

34 региона
России и несколько
областей Казахстана

35 млн охват населения

1500 медицинских организаций

Обеспечивает реализацию задач
национального проекта
«Здравоохранение»

ПРИМЕРЫ РЕАЛИЗАЦИИ

Пермский край

2.623.100

ЧЕЛОВЕК

- 40 390 пользователей
- 15 515 онлайн
- 100% пациентов с ЭМК
- 55% электронных записей на приём через мобильное приложение
- 17.2 млн. электронных документов ежемесячно и более 200 млн ежегодно
- БД 5.3 Тб

ПРИМЕРЫ РЕАЛИЗАЦИИ

Республика Башкортостан

4.051.000

ЧЕЛОВЕК

- 86 523 пользователей
- 18 515 онлайн
- 100% пациентов с ЭМК
- 235 025 605 записей на приём в электронном виде
- 963 475 626 электронных документов в базе данных
- БД 7.7 Т6

Технические особенности



PHP 7, Ext.JS



Java 8, Mybatis, CXF



Apache ActiveMQ

Не используем в разработке фреймворки,
генерирующие SQL-запросы

Правило программиста:

**Читать из представлений, добавлять, обновлять,
удалять записи с помощью хранимых процедур**

Трудности выбора

- 1 Много данных
- 2 Большое количество пользователей
- 3 Нагрузка на БД
- 4 Информация, требующая защиты
- 5 ???
- 6 ???
- 7 **PostgreSQL!**

Статистика по полям в БД

Тип MSSQL Количество полей

Тип PostgreSQL

| | | |
|-------------------------|-------|---|
| bigint | 31750 | bigint |
| datetime | 12942 | timestamp |
| varchar | 11310 | varchar |
| int | 7443 | int |
| nvarchar | 1603 | varchar |
| float | 1358 | double precision (decimal) |
| rowversion | 1002 | bytea + эмуляция через последовательности |
| numeric | 818 | numeric |
| money | 203 | money |
| uniqueidentifier | 168 | uuid |
| decimal | 50 | decimal |
| varbinary | 37 | bytea |
| time | 30 | time |
| xml | 8 | xml |

По объектам БД

Объекты БД для переноса

| Описание | Количество |
|-----------------------|------------|
| Таблицы | 4811 |
| Представления (вьюхи) | 4137 |
| Хранимые процедуры | 14408 |
| Внешние ключи | 11292 |
| Скалярные функции | 246 |
| Табличные функции | 6226 |

Хранимые процедуры

| | |
|----------------------------|-------|
| Всего | 14408 |
| Из них не автогенерируемые | 3764 |

Представления

| | |
|----------------------------|------|
| Всего | 4137 |
| Из них не автогенерируемые | 679 |

Предпосылки перехода

Необходимость перехода



Возрастающая стоимость лицензий MSSQL



Курс на импортозамещение в ПО, работающем в госсекторе



Лицензионная политика MSSQL, ограничивающая использование аппаратных ресурсов в приемлемых по стоимости редакциях



Надежды что станет еще лучше

Мы хотели попробовать что-то новое 😊

Предпосылки перехода

Нас останавливало



Большой объем
SQL-кода для
перевода



Непрерывная
разработка
и доработка



Много
разнообразных
сервисов



Непрерывная
работа
системы



Накопленные
в процессе работы
проблемы



**Нельзя так просто взять
и перейти на PostgreSQL**

Различия синтаксиса

- Хинты
- Конкатенация
- TOP
- OUTER/CROSS APPLY
- Функции даты-времени
- Строковые функции
- ISNULL
- CAST и CONVERT
- Вызов хранимых процедур
- Xml-операторы
- Объявление переменных
- Объявление переменных в запросах
- Присвоение значений переменным
- Значение по умолчанию в параметрах хранимых процедур и функций
- Регистрозависимые объекты
- Регистрозависимое сравнение строк и оператор LIKE

Что можно сделать для замены?



Обращались в организации, занимающиеся переводом. Но предлагаемая стоимость работ была слишком высокой



Пробовали конвертировать с помощью регулярных выражений, но результат был далек от идеала



Хотели написать конвертер с помощью ANTLR, но, прочитав документацию, решили пока отказаться от этой идеи



Для некоторых запросов хорошо работали online-сервисы конвертации, мы использовали <http://www.sqlines.com/online>.

Но в целом, большую часть работы пришлось выполнять вручную.

Перенос данных

1

SQLines Data

<http://www.sqlines.com/sqldata>

Работает достаточно быстро, если возникают ошибки - продолжает работать дальше
Проблемы с импортом данных в кириллице, случается непредсказуемо падает

2

SQLWorkbench

<http://www.sql-workbench.eu>
в режиме Savepoint

Все то же самое что в предыдущем пункте, но еще медленнее
Если возникают ошибки - продолжает работать

3

SQLWorkbench

<http://www.sql-workbench.eu>
без режима Savepoint

Работает медленнее, если возникают ошибки - останавливается
Стабильна, много настроек, особых проблем за время тестирования не обнаружено

4

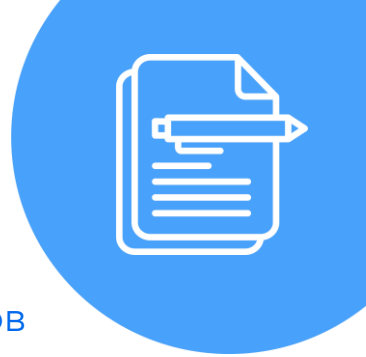
Еще можно использовать

FDW

требует настройки FDW и структуры таблиц

COPY

требуется много места для хранения



Выводы по переносу



SQLines Data по скорости один из самых быстрых вариантов
Но из-за проблем с передачей кириллицы и внезапных падений можно использовать только для таблиц, не содержащих строковые данные и на небольшом количестве таблиц



SQLWorkbench можно использовать для всего и стабильно
Для повышения скорости можно запускать в несколько процессов по разным таблицам и схемам



Если надо еще быстрее – **FDW** и **COPY**

Всегда стоит вопрос поддержки актуальности данных в БД для перехода. Т.е. все перенесли, затем прошло время и надо снова актуализировать данные.

Перенос структуры

1

Intelligent Converter, MS SQL to PostgreSQL

<https://www.convert-in.com/mss2pgs.htm>

В пробной версии нет конвертации представлений. Не удалось перенести другие схемы кроме dbo.

49 \$

2

Sqlines, SQLines SQL Converter, Open Source

<http://www.sqlines.com/download>

Предназначен для конвертации скриптов, но не для переноса структуры

free

3

DbConvert, Microsoft SQL Server to PostgreSQL Migration

<https://dbconvert.com/mssql/postgresql>

Работает только для небольших БД. Для наших объёмов необходимо запастись терпением

149 \$

4

SQLMaestro, PostgreSQL Database Converter

<https://www.sqlmaestro.com/products/postgresql/converter>

Работает, удобен, много настроек. Очень низкая скорость работы

139 \$

5

Liquibase.Org, Liquibase

<http://www.liquibase.org>




Хорошая скорость работы. Не поддерживается специфика последних версий Postgres

free



Выводы по переносу

Как оказалось, ни одна из этих утилит не конвертирует структуру «из коробки», везде есть свои проблемы:

-  Не рассчитаны на такое количество таблиц
-  Работают очень медленно
-  Переносят структуру с ошибками

Взвесив результаты и возможность доработки мы, остановились на варианте с Liquibase, но с некоторыми доработками.

Дорабатываем Liquibase:

xpath + regex + драйвер БД = МАГИЯ!!!

- 1 Все объекты сохраняются в нужном регистре
- 2 Можно делать замену типов для столбцов таблиц
- 3 Замену SQL-выражений и функций в представлениях
- 4 Добавить дополнительные шаги, например для запуска скрипта после создания объекта
- 5 Фильтровать объекты по различным критериям: типу, наименованию и т.д.

Что не переносится по структуре

1

Хранимые процедуры
и функции

2

Представления, при большом
количестве платформу-зависимого кода

Проснуться и везде PostgreSQL – это фантастика



Нельзя остановить систему на долгое время для перехода или конвертации



Нельзя остановить текущую разработку



Всегда есть модули, которые не протестированы в полном объеме и не будут корректно работать



Всегда есть сервисы, о которых просто «забыли»



Если что-то пошло не так, нужно иметь возможность быстро откатиться.

**наша задача —
поменять колеса
быстро движущегося
поезда.**

1 Upminster
2 Circle Line via L
3 Upminster

Symmetric DS



Действительно работает и реплицирует данные



Гибкие и удобные настройки



Все изменения собираются с помощью триггеров



Очень долгий процесс «роутинга»



Сбор данных с помощью триггеров создает нагрузку на БД



Складирование данных в большую таблицу в той же БД

В результате - отставание данных на несколько часов.

Проблема: некорректный порядок записей в случае их последовательного обновления в одной транзакции.

Альтернативная репликация

Основные принципы

- Для MSSQL собираем данные, анализируя поле типа rowversion
- Для PostgreSQL собираем данные просматривая даты изменения. В будущих версиях – через эмуляцию rowversion или wal2json
- Формат обмена – JSON. Каждой записи в таблице соответствует одно JSON-сообщение
- На сервере хранится только журнал репликации. Данные после выгрузки попадают сразу в Брокер
- Используем все возможные для брокера режимы подписки

Альтернативная репликация

Возможности

- Кроссплатформенная
- Поддерживает MSSQL и PostgreSQL
- Настройки через yaml-файл
- Разнообразная статистика
- Многопоточность
- Отправка данных по таблице или с помощью произвольного запроса
- Списки на включение/исключение полей
- Если идет несколько подряд изменений записи, то в репликацию пойдет только последнее состояние
- Различные модели подписки
- Интеграция с другими сервисами и частями программного комплекса
- Настраиваемая через JavaScript обработка ошибок
- Обработка данных с помощью JavaScript

Альтернативная репликация

Решаемые задачи

- Передача данных между MSSQL – MSSQL
- Передача данных между MSSQL – PostgreSQL
- Асинхронный пересчет различных показателей
- Асинхронная обработка данных от различных сервисов
- Уведомление сервисов о появлении новых данных
- Дополнительная обработка данных при передаче между различным БД

Объектов:

1051

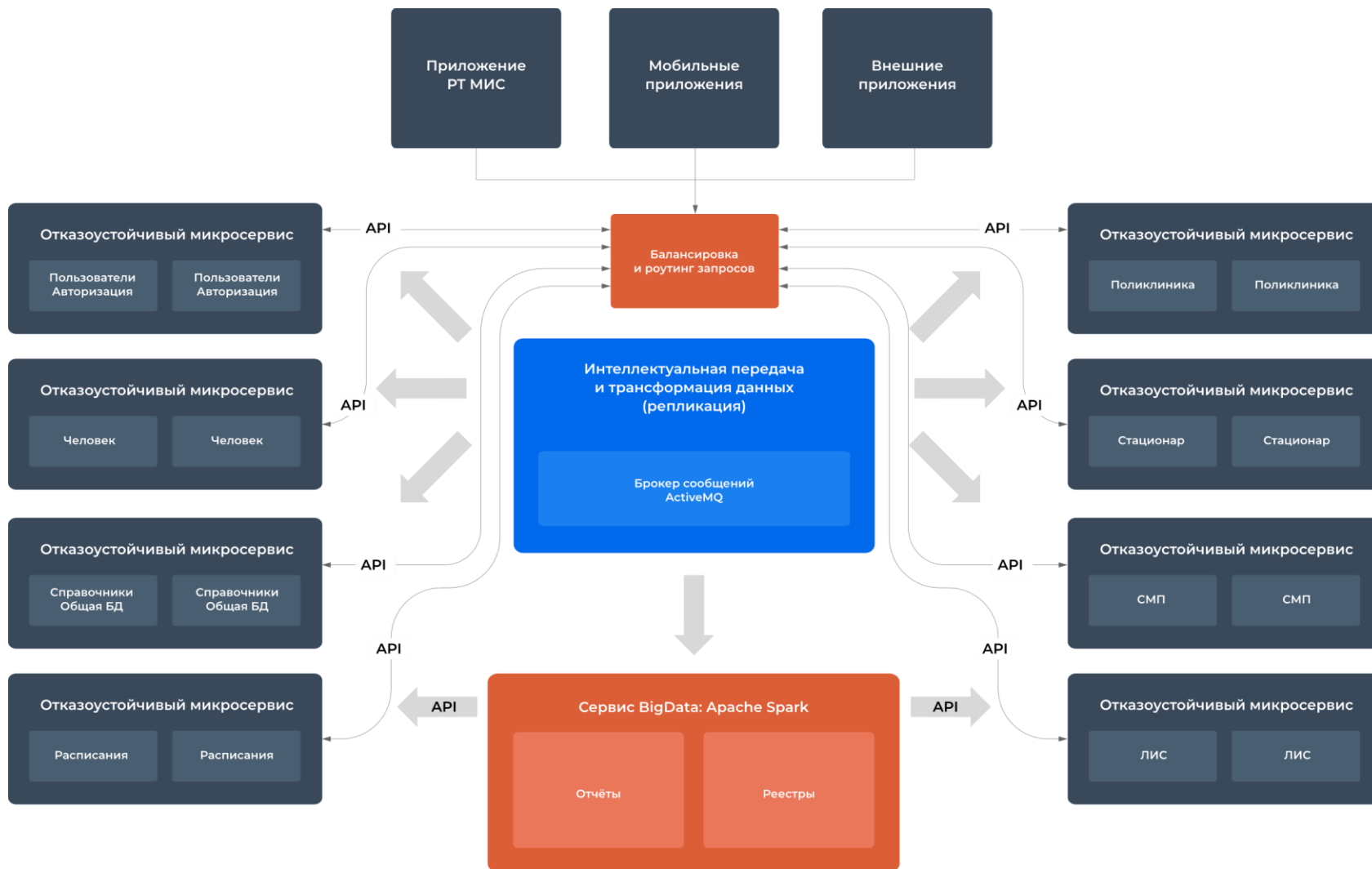
Сообщений:

1.1 млн./час

Скорость:

12 тыс. с/сек

Схема работы





Выводы

1. Перейти на другую платформу, с MSSQL на PostgreSQL – возможно
2. Переход будет долгим
3. На PostgreSQL есть своя специфика
4. Необходимы четкие регламенты по разработке
5. Сама по себе конвертация структуры БД занимает немного времени
6. Наибольшее количество трудозатрат потребует переписывание SQL-запросов в представлениях, хранимых процедурах и прикладном коде
7. Еще большее количество трудозатрат потребуется на дальнейшую отладку и оптимизацию после переписывания



СЧАСТЬЕ НЕ ЗА ГОРАМИ