А Э Р О

Как мы тестировали 5 способов загрузки данных в Greenplum

И что из этого вышло

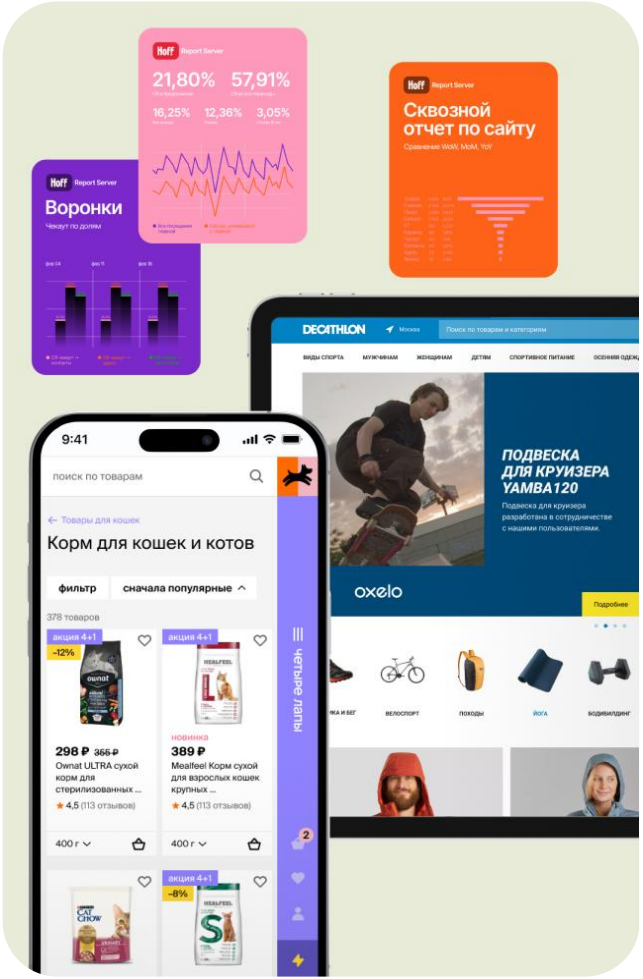
Татьяна Дидова

Tech Lead Data Engineer в АЭРО

5 лет в IT. Занимаюсь разработкой гибких архитектурных решений DWH для различного объема данных. Основной вектор работы направлен на технологическое развитие компаний.

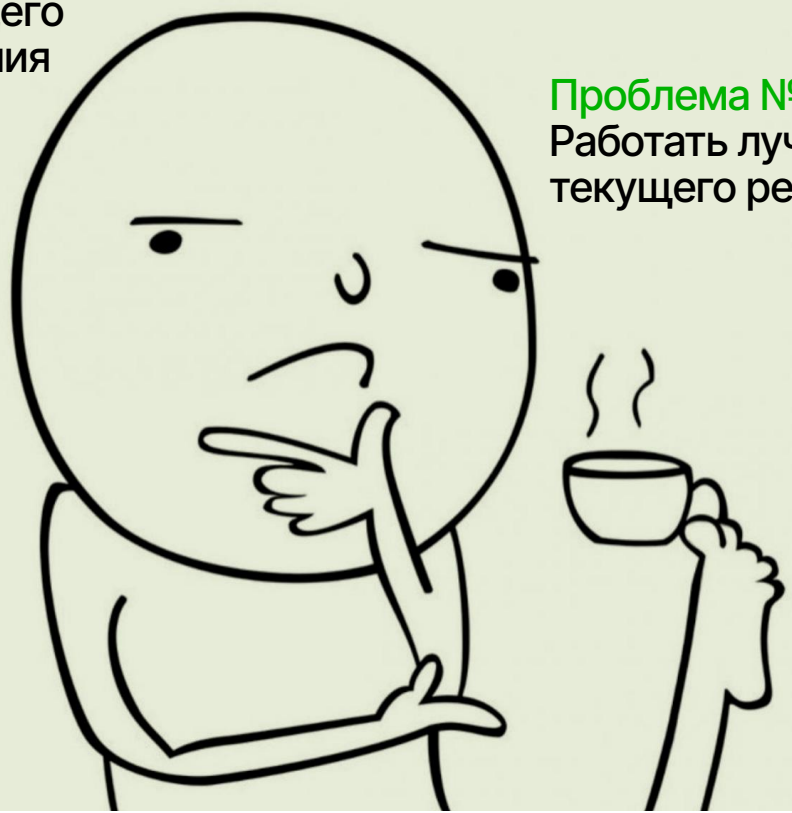
АЭРО. Провайдер e-commerce & data решений

- Создаем интернет-магазины и приложения больше 17 лет
- eCommerce-агентство №1 по данным «Рейтинга Рунета 2023»
- Призеры конкурсов «МИКС Россия 2022» и «Большой оборот 2018»
- Крупнейшие российские eCom-проекты разработаны при нашем участии: М.Видео, ЦУМ, Стокманн, Gloria Jeans, Утконос, Burger King, Лента, 36,6
- Адаптированы под крупный бизнес с выручкой 50+ млрд рублей
- >150 млн посетителей приходят на наши площадки каждый месяц



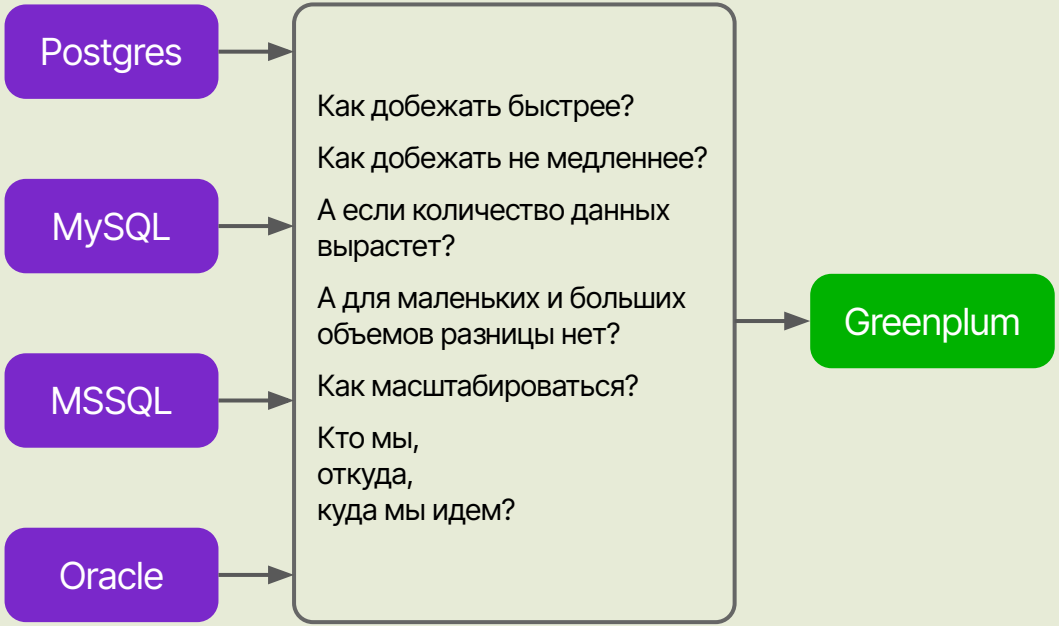
Поступила задача

Проблема №1:
Работать не хуже текущего решения

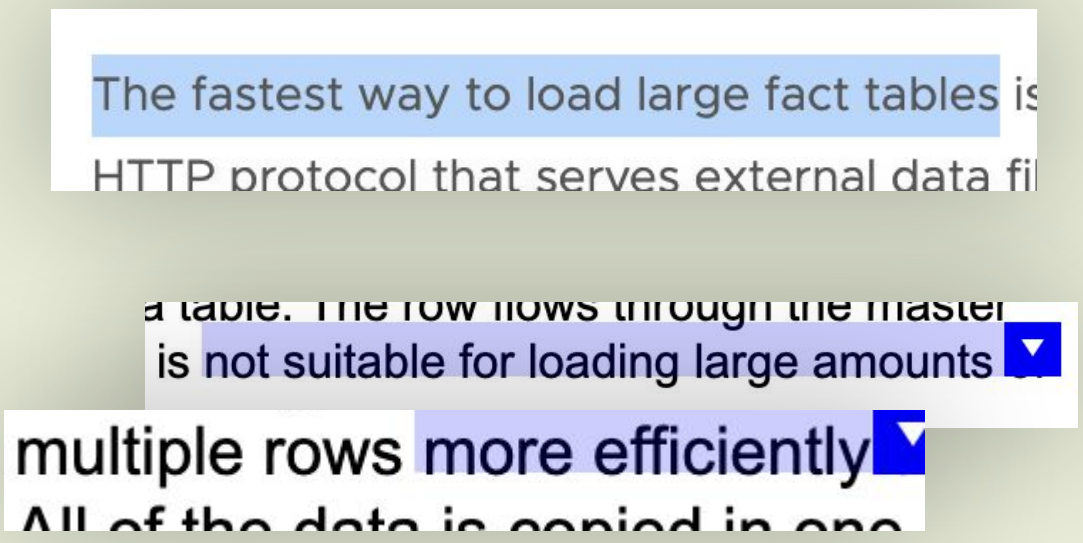


Проблема №2:
Работать лучше текущего решения

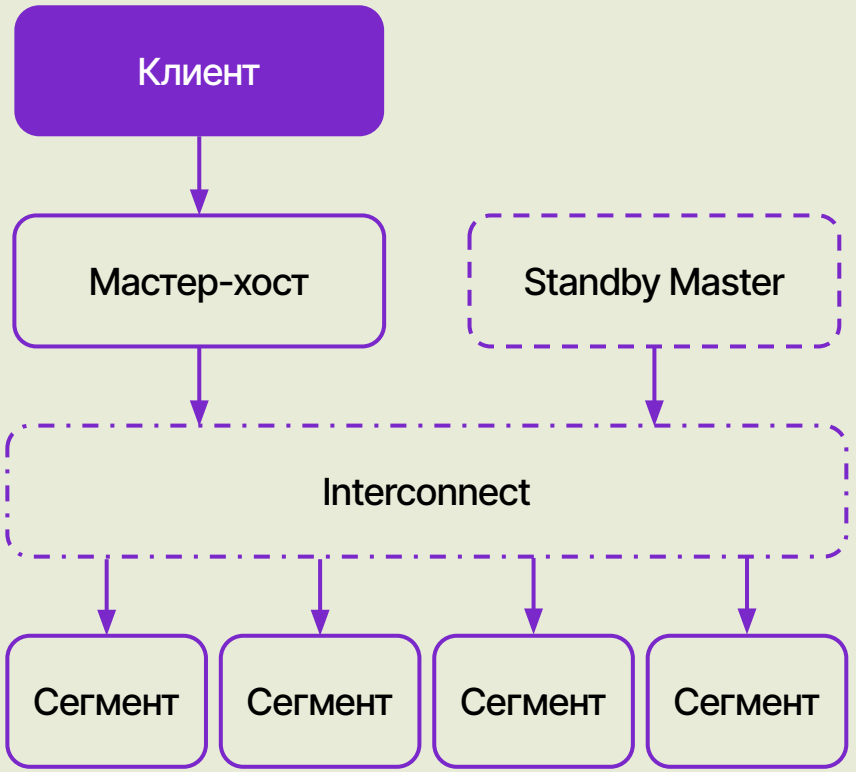
Как это всё выглядит



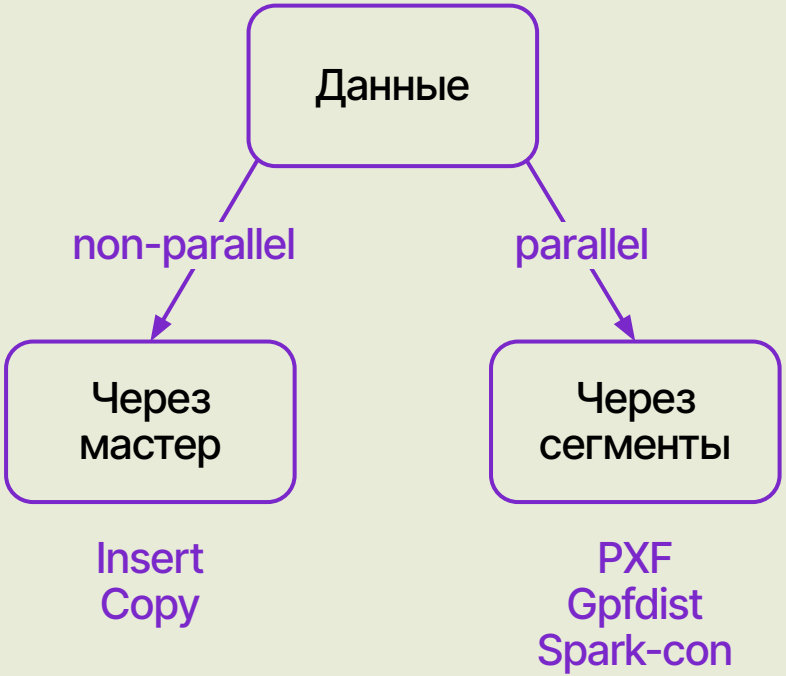
Amazing The Best Fantastic



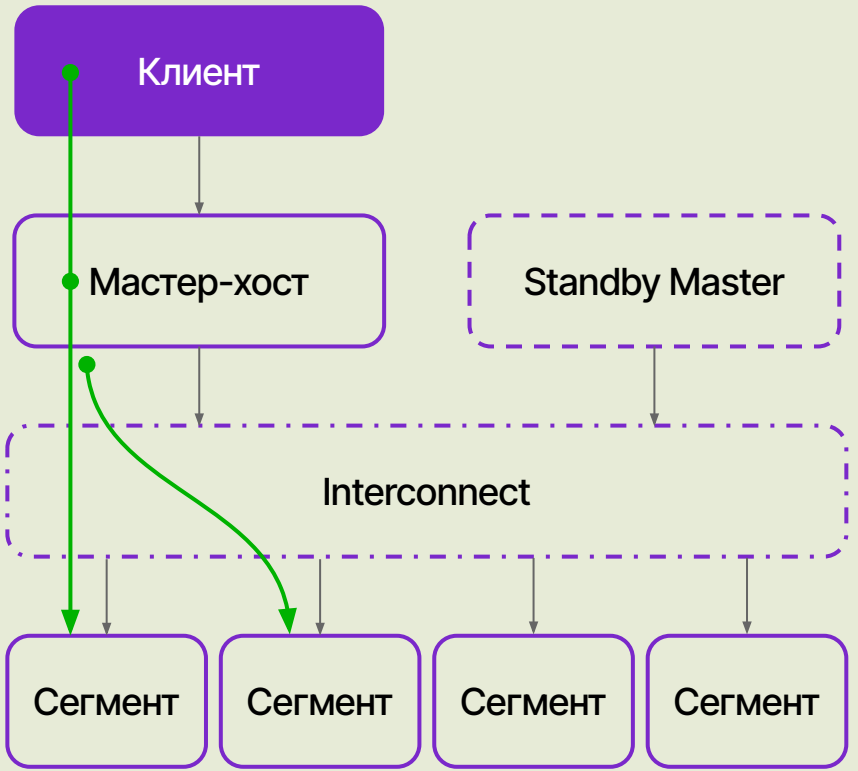
Минутка теории
(не минутка)



В чём разница способов?



Insert/Copy


Клиент
Мастер-хост
Standby Master
Interconnect
Сегмент
Сегмент
Сегмент
Сегмент

Insert

Плюсы

- **Не нужно сложных настроек:** нужен только один сервер для развертки кода загрузчика:)
- **Удобство отладки:** так как позволяет вставлять отдельные строки или небольшие наборы данных и сразу же видеть результаты

Минусы

- **Высокая нагрузка на мастер-узел:** большое количество запросов может создать нагрузку на мастер-узел
- **Как следствие - низкая эффективность при большом объеме данных:** плохо масштабируется при необходимости вставки большого объема данных

Copy

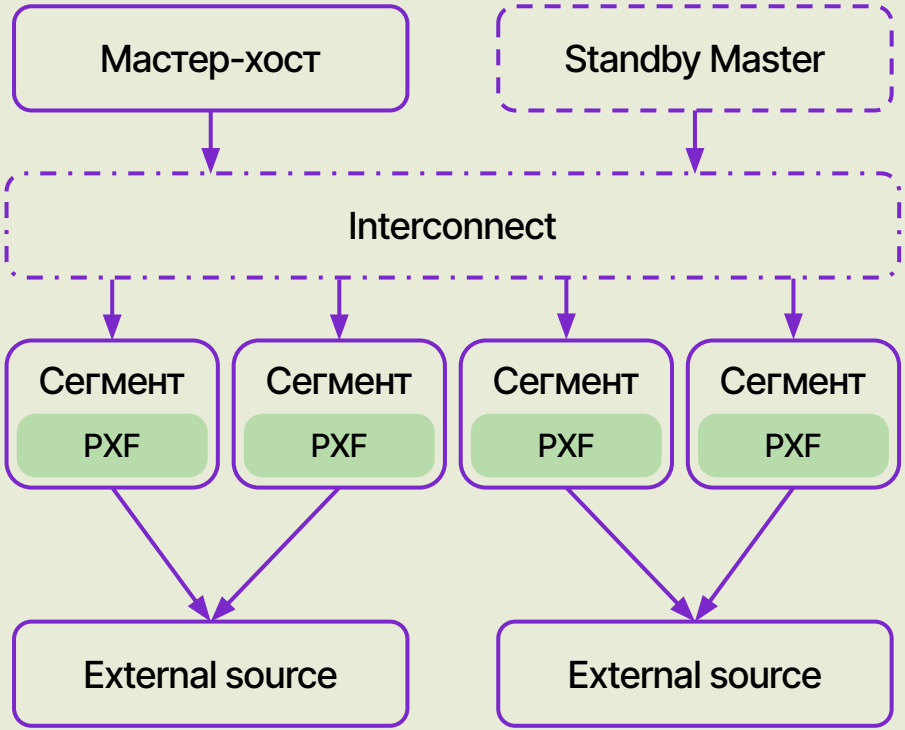
Плюсы

- **Не нужно сложных настроек:** нужен только один сервер для развертки кода загрузчика:)
- **Высокая производительность:** самый быстрый не параллельный способ загрузки данных в Greenplum.

Минусы

- **Высокая нагрузка на мастер-узел:** большое количество запросов может создать нагрузку на мастер-узел
- **Проблемы с ошибками данных:** если в данных есть ошибки или несоответствия, COPY может завершиться неудачей.

PXF


Мастер-хост
Standby Master
Interconnect
Сегмент
PXF
Сегмент
PXF
Сегмент
PXF
Сегмент
PXF
External source
External source

PXF

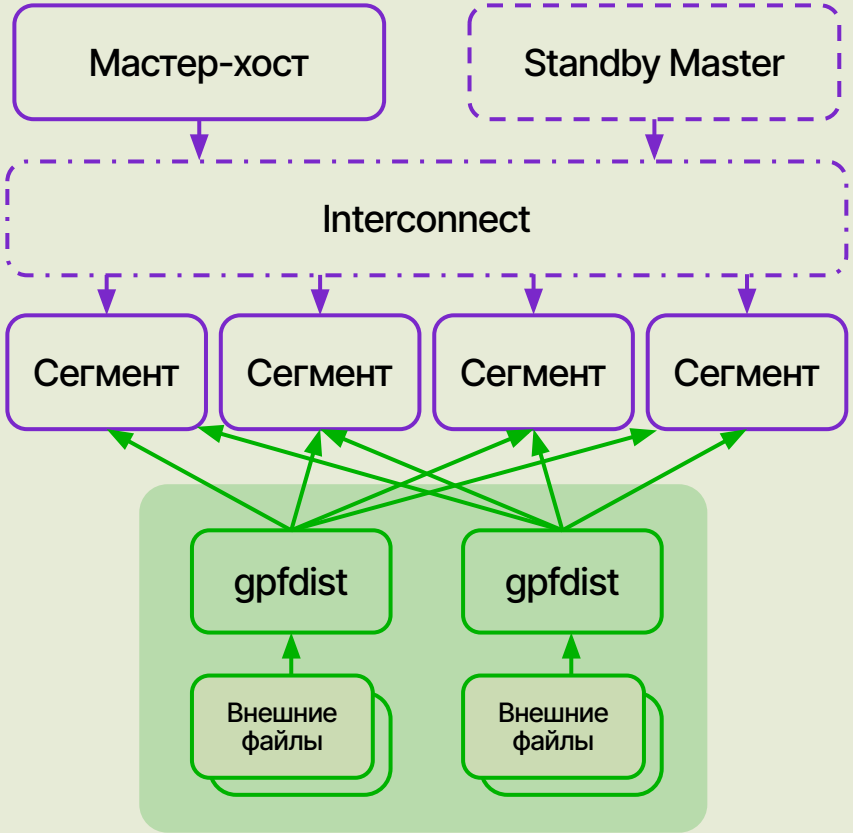
Плюсы

- **Прямой коннект к источнику:** PXF позволяет подключаться ко множеству различных источников данных, включая HDFS, Amazon S3, и другим распределенным файловым системам.
- **Нужно знать только SQL и немного credentials для подключения к источнику**

Минусы

- **Сложность настройки и администрирования:** требует установки и настройки отдельных служб, которые работают на каждом сегментном узле Greenplum.
- **Отъедание ресурсов сегмента:** потому что pxf устанавливается непосредственно на мощностях сегмента
- **Зависимость от источника:** нужно правильно класть данные не только к себе в базу, но и оптимизировать источник
- **Те самые credentials в запросе подключения**

Gpfdist


Мастер-хост
Standby Master
Interconnect
Сегмент
Сегмент
Сегмент
Сегмент
gpfdist
gpfdist
Внешние файлы
Внешние файлы

Gpfdist

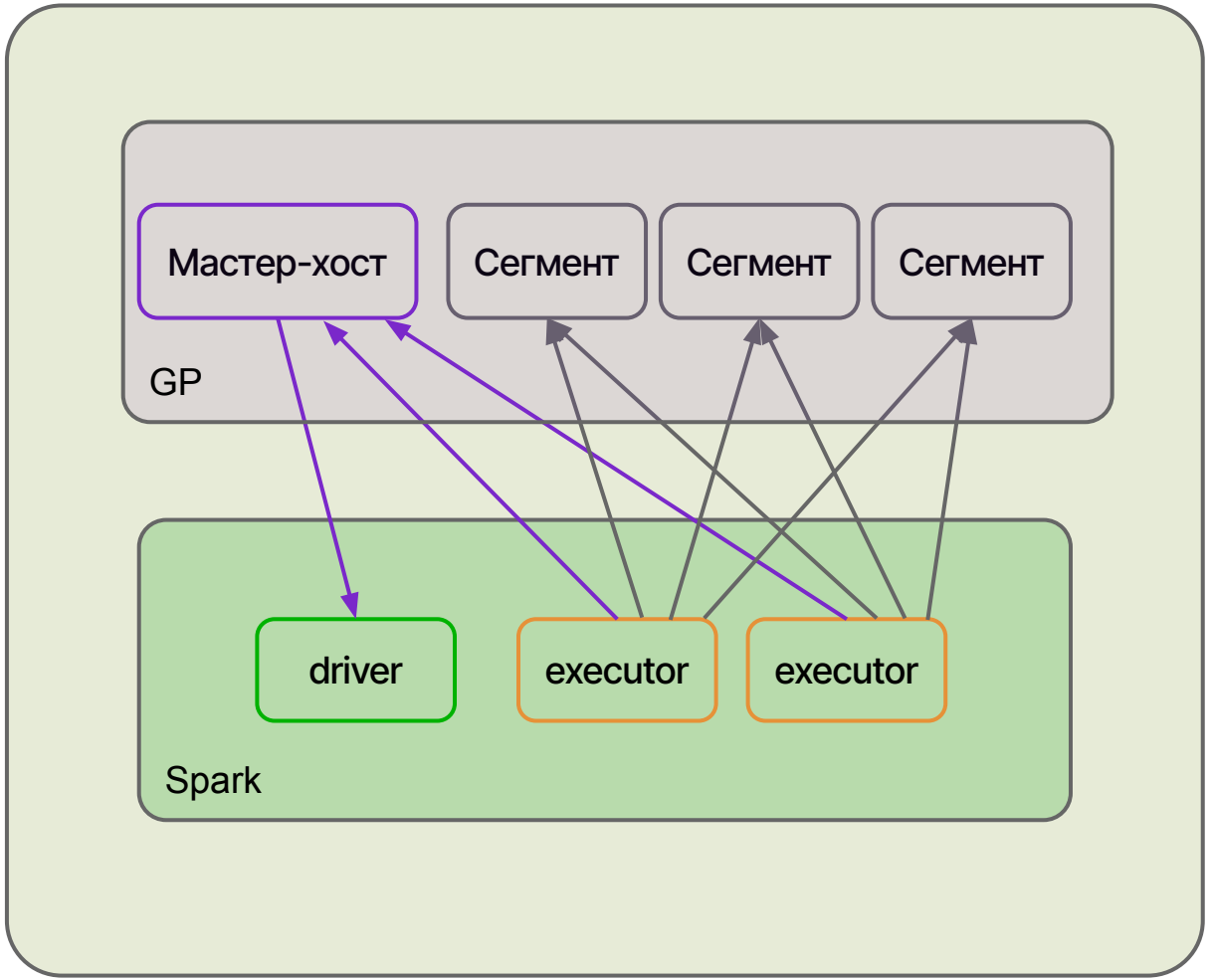
Плюсы

- **Оптимизированное чтение:** Gpfdist оптимизирован для работы с Greenplum, что обеспечивает эффективное использование ресурсов
- **Настраиваемые параметры:** Вы можете настроить параметры gpfdist, такие как количество потоков и размер блоков, что позволяет адаптировать его под конкретные задачи и характеристики данных

Минусы

- **Зависимость от производительности сети:** Gpfdist работает через сеть, и производительность загрузки данных напрямую зависит от пропускной способности сети.
- **Необходимость управления множеством экземпляров:** может потребоваться запуск нескольких экземпляров gpfdist на различных узлах, чтобы справляться с большими объемами данных.
- **Ограниченная масштабируемость:** числом сегментов и мощностью узлов, на которых он работает.
- **Порты:** могут возникнуть сложности с тем, чтобы зафиксировать порты gpfdist

Spark connector



Spark connector

Плюсы

- **Поддержка сложных трансформаций**: что упрощает создание сложных пайплайнов обработки данных.
- **Расширение спектра источников по сравнению с gpfdist**

Минусы

- **Сложность первичной настройки:** требуется знание специфических параметров и конфигураций для обеспечения оптимальной производительности
- **Интеграция и совместимость:** возможные проблемы с интеграцией и совместимостью версий Spark и Greenplum могут усложнить настройку и использование коннектора
- **Отъедает время на развертку и преобразования по ходу процесса**

К эксперименту

Забирали данные из PG (Не тестировали сам забор данных, поэтому просто выбрали удобную БД)

Конфигурация кластера:
2 мастера 8 CPU, 16 RAM
4 сегмента 16 CPU, 64 RAM

TPC-H:
100 тыс, 10 млн,
20 млн записей

Insert

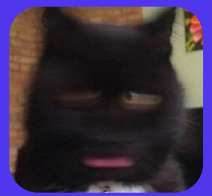
- **SINGLE INSERT:**
  Вставка одной строки за один запрос.
  1600 сек. / 100 тыс. строк
- **EXECUTE MANY:**
  Вставка нескольких строк за один запрос
  103 сек. / 100 тыс. строк
- **EXECUTE BATCH (extra):**
  Объединение нескольких запросов в один большой запрос
  127 сек. / 100 тыс. строк
- **EXECUTE VALUE (extra):**
  Объединение всех данных в один запрос
  4 сек. / 100 тыс. строк

Copy

Быстрый

1,4 сек. /
100 тыс. строк

PXF



Быстрый

35 сек. /
10 млн строк

64 сек. /
20 млн строк

Gpfdist

Быстрый

37 сек. /
10 млн строк

78 сек. /
20 млн строк

Spark connector

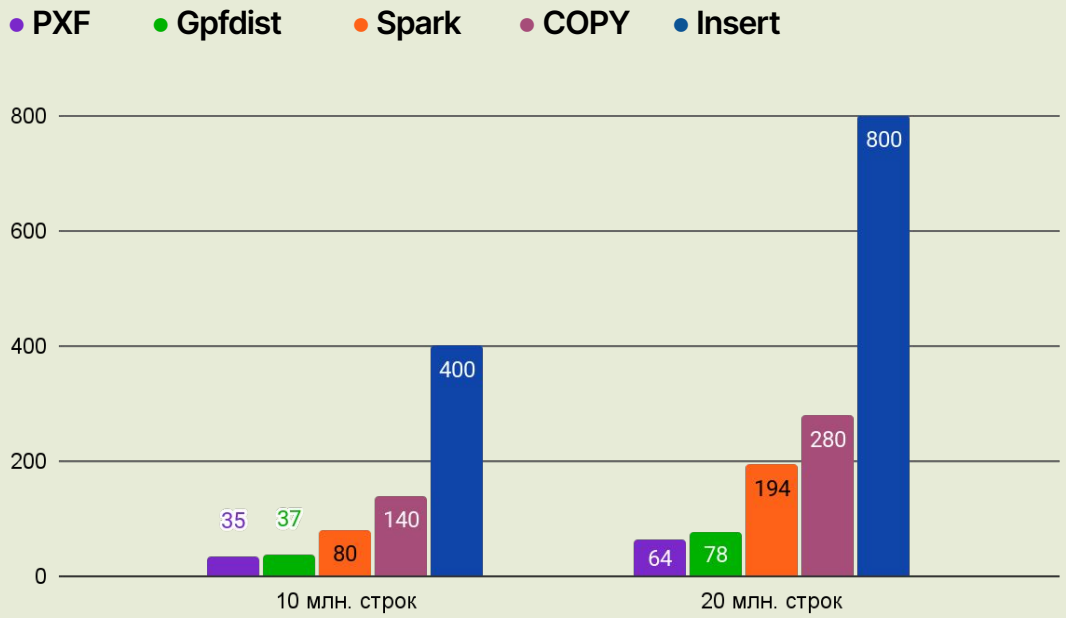
Средне-быстро

80 сек. /
10 млн строк

216 сек. /
20 млн строк

Сравнение замеров

Скорость загрузки, сек (меньше — лучше)



Сравним

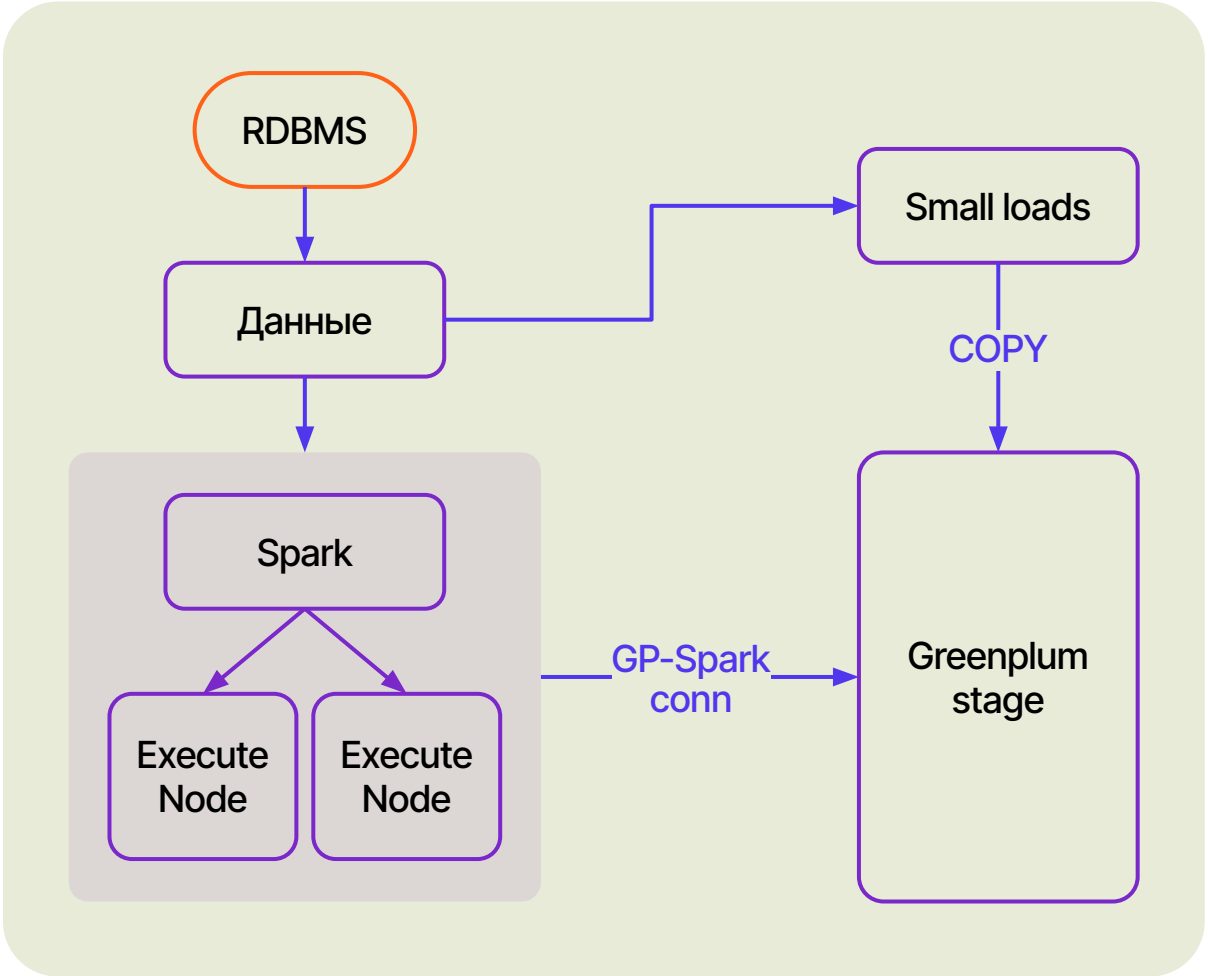
	Insert	Copy	PXF	Gpfdist	Spark
Файлы	CSV	CSV, TSV	CSV, TSV, Avro, JSON, Parquet and ORC	CSV, TSV и сжатые файлы gzip и bzip2	CSV, TSV, Avro, JSON, Parquet and ORC
Прямой коннект	-	-	HDFS, Amazon S3, JDBC-connect	-	HDFS, S3, Kafka, JDBC-connect
Плюсы	Не нужно сложных настроек Удобство отладки	Не нужно сложных настроек Лучшая производительность относительно не параллельных методов	Прямой коннект к источнику Достаточно знать SQL	Оптимизированное чтение Настраиваемые параметры	Поддержка сложных трансформаций Шире спектр источников по сравнению с gpfdist
Минусы	Высокая нагрузка на мастер-узел Низкая эффективность при большом объеме данных	Высокая нагрузка на мастер-узел Сложнее дебажить проблемы в массиве данных	Сложность настройки и администрирования Отъедание ресурсов сегмента Зависимость от источника Не подходит для adhoc	Зависимость от производительности сети Необходимость управления множеством экземпляров Ограниченная масштабируемость	Сложность первичной настройки Интеграция и совместимость Дополнительное время на развертку

Тюнинг

	Insert	Copy	PXF	Gpfdist	Spark
Общий	Удалить все индексы из существующей таблицы перед загрузкой данных. Создание нового индекса выполняется быстрее, чем постепенное обновление индекса по мере загрузки каждой строки Отключить автоматический сбор статистики во время загрузки, установив для параметра конфигурации gp_auto start_mode значение NONE Запустить VACUUM, чтобы освободить место, если при загрузке были ошибки				
Частный	Используйте COPY:)	Запустить несколько одновременных команд COPY.	Оптимизировать данные на стороне источника Изменить размер *pxf.jdbc.fetch-size* и *pxf.max-fragment-size*	Использовать PIPE Добавить сегментов в кластер GP:) Не сжимать передаваемые данные	Расширить используемые ресурсы под execute node Изменить размер *batch size*, fetchSize

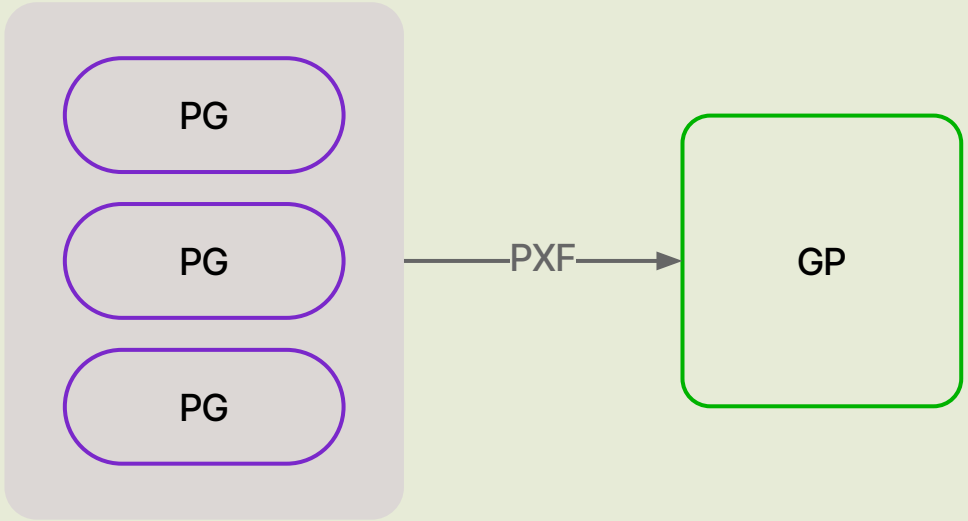
Куда применили

Архитектура 1

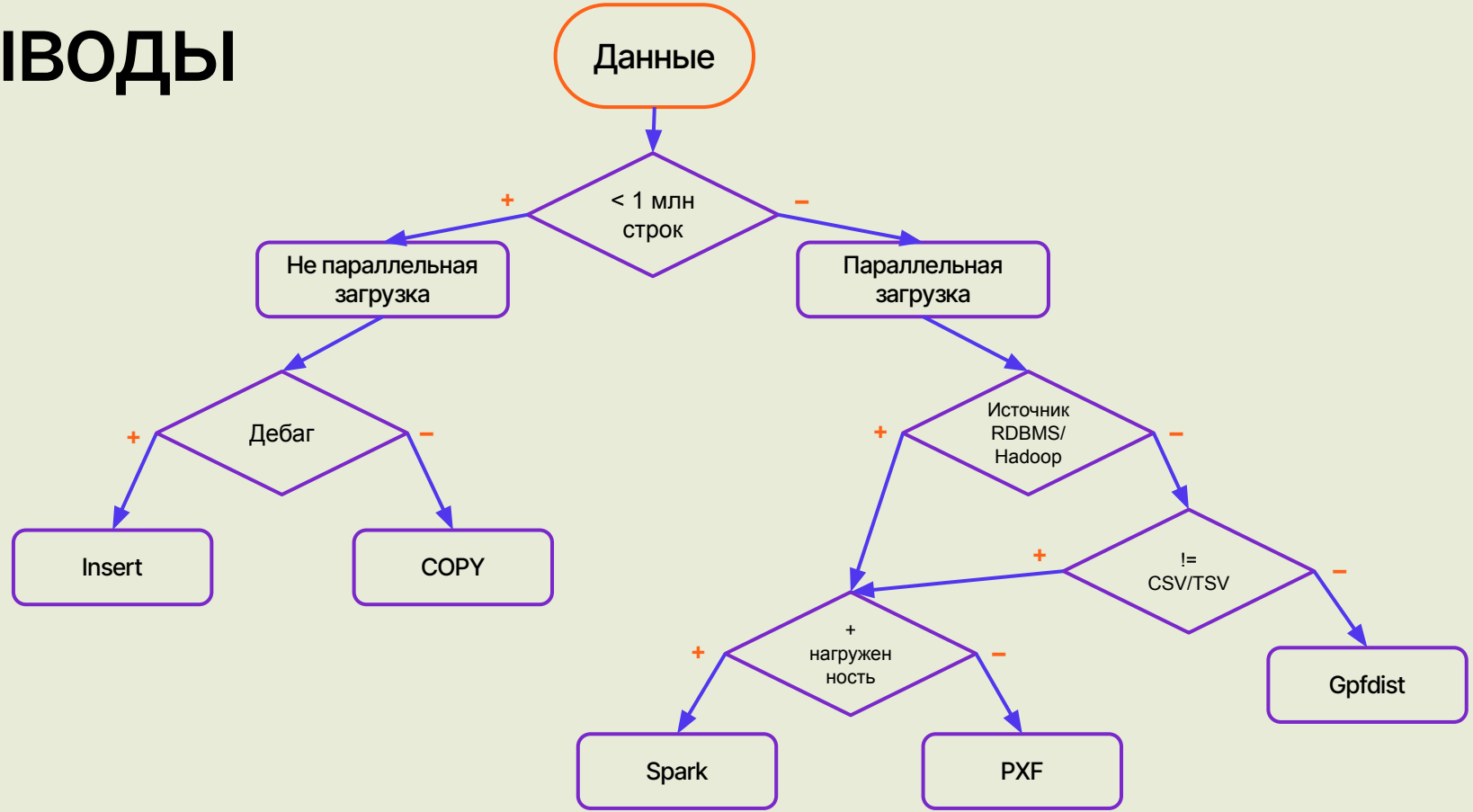


Куда применили

Архитектура 2



Выводы


Данные
< 1 млн строк
+
–
Не параллельная загрузка
Параллельная загрузка
Дебаг
+
–
Insert
COPY
Источник RDBMS/ Hadoop
+
–
!= CSV/TSV
+
–
+ нагружен ность
+
–
Spark
PXF
Gpfdist

Спасибо!