PostgreSQL Cloud Solutions

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Тема вебинара

Способы горизонтального масштабирования PostgreSQL

Алексей Железной

Data Engineer

- 4 года опыта Инженером данных/аналитиком (Python, Airflow, Clickhouse, Greenplum)
- 2 года опыта преподавания в OTUS (курсы DWH Analyst/DE/PostgreSQL Cloud)

LinkedIn

Правила вебинара

Активно
участвуем

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Маршрут вебинара

- Параллельные кластеры БД
- Способы горизонтального масштабирования PostgreSQL
- Практика
- Рефлексия

Цели вебинара

1. Иметь представление горизонтальном масштабировании PostgreSQL
2. Понимать когда стоит использовать горизонтальное масштабирование, а когда нет
3. Уметь настраивать параллельный кластер

Смысл

Зачем вам это уметь

1. Если придется столкнуться с параллельными кластерами, знать что и как можно сделать реально
2. А главное понимать, “нужно ли?”

Параллельные кластеры БД

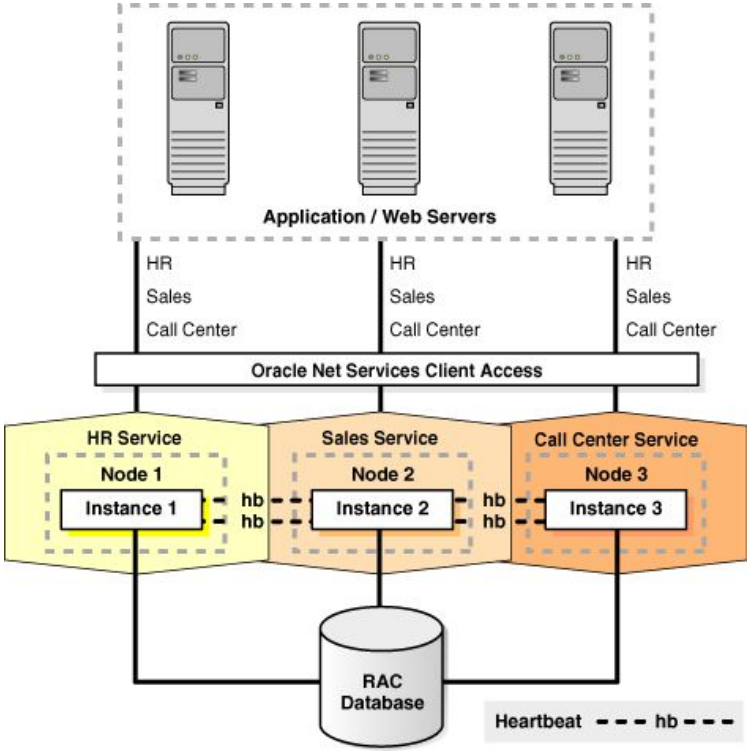
Параллельные кластеры сегодня

Только про ACID и SQL

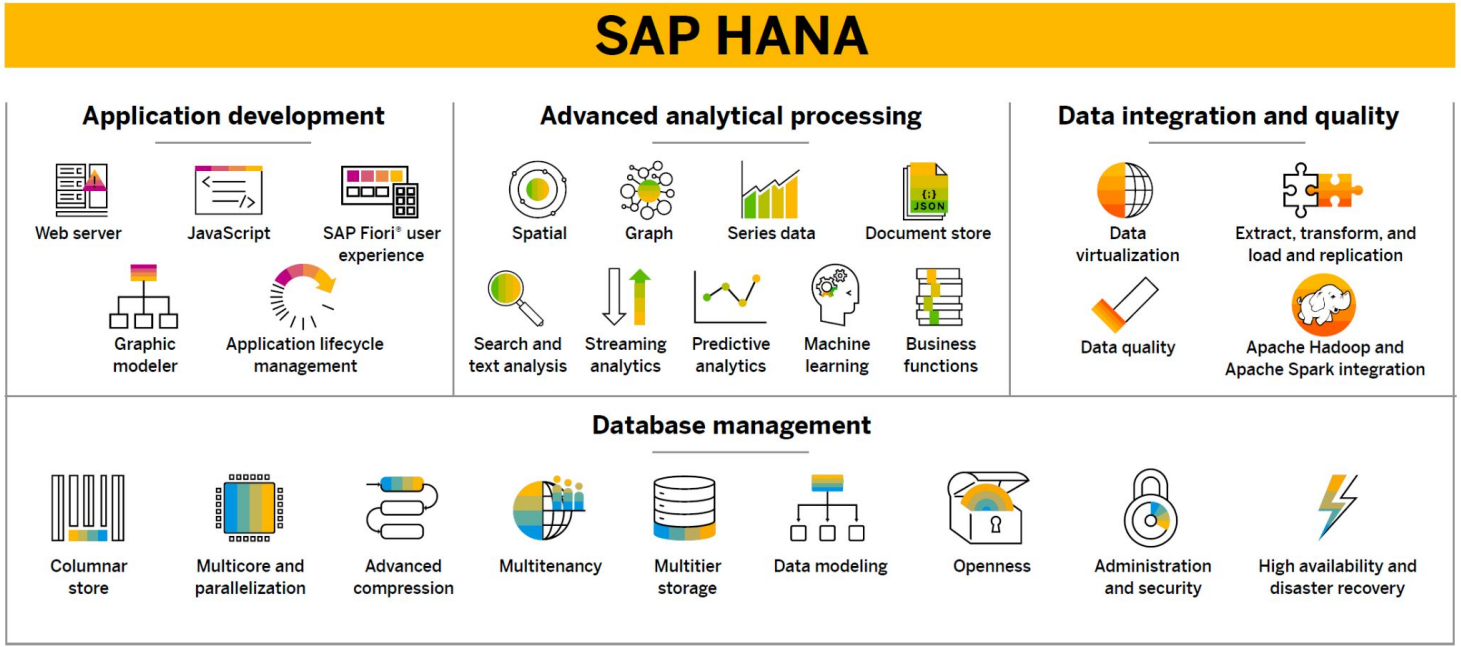
- Oracle RAC / Exadata
- SAP HANA
- Google Cloud Spanner
- MySQL Galera
- Teradata

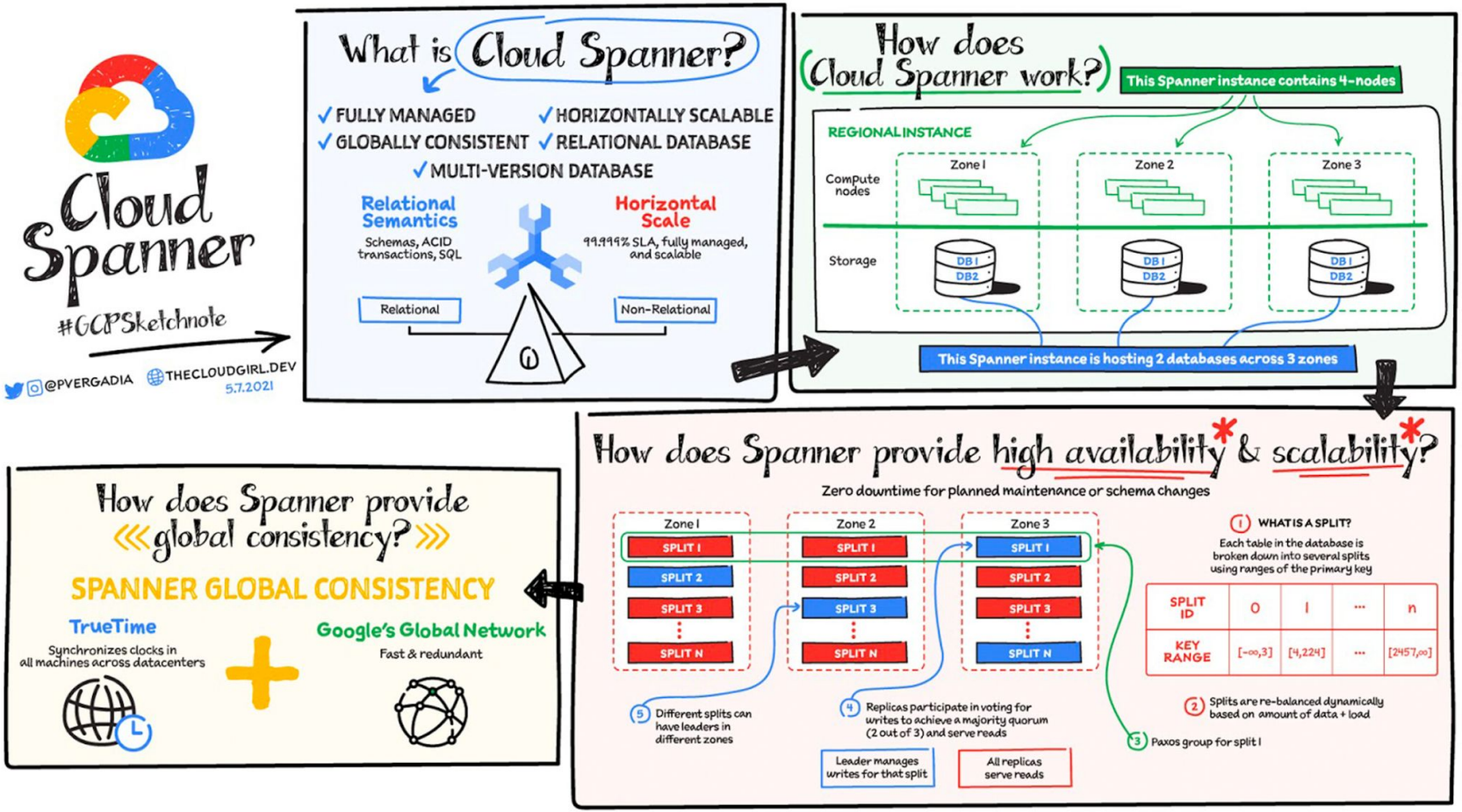
Oracle RAC / Exadata

- Real Application Clusters
- Стратегическое аппаратно-программное решение Oracle для
  - OLTP
  - Хранилищ данных
  - Смешанных нагрузок
  - Консолидации приложений на базе Oracle Database
- Построено на основе:
  - Oracle Database, т.е. все приложения, работающие на Oracle, могут работать на Exadata
  - SUN Hardware



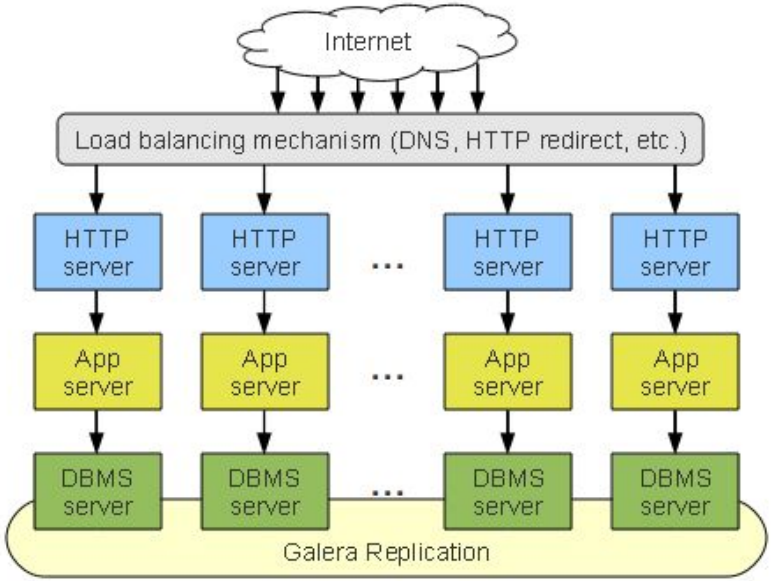
SAP HANA


SAP HANA
Application development
Web server
JavaScript
SAP Fiori® user experience
Graphic modeler
Application lifecycle management
Advanced analytical processing
Spatial
Graph
Series data
Document store
JSON
Search and text analysis
Streaming analytics
Predictive analytics
Machine learning
Business functions
Data integration and quality
Data virtualization
Extract, transform, and load and replication
Data quality
Apache Hadoop and Apache Spark integration
Database management
Columnar store
Multicore and parallelization
Advanced compression
Multitenancy
Multitier storage
Data modeling
Openness
Administration and security
High availability and disaster recovery


Cloud Spanner
#GCPSketchnote
@PVERGADIA
THECLOUDGIRL.DEV
5.7.2021
What is Cloud Spanner?
✓FULLY MANAGED
✓HORIZONTALLY SCALABLE
✓GLOBALLY CONSISTENT
✓RELATIONAL DATABASE
✓MULTI-VERSION DATABASE
Relational Semantics
Schemas, ACID transactions, SQL
Horizontal Scale
99.999% SLA, fully managed, and scalable
Relational
Non-Relational
How does Cloud Spanner work?
This Spanner instance contains 4-nodes
REGIONAL INSTANCE
Zone 1
Zone 2
Zone 3
Compute nodes
Storage
DB 1
DB 2
This Spanner instance is hosting 2 databases across 3 zones
How does Spanner provide high availability* & scalability*?
Zero downtime for planned maintenance or schema changes
SPLIT 1
SPLIT 2
SPLIT 3
SPLIT N
1 WHAT IS A SPLIT?
Each table in the database is broken down into several splits using ranges of the primary key
SPLIT ID
KEY RANGE
[-∞,3]
[4,224]
[2457,∞]
2 Splits are re-balanced dynamically based on amount of data + load
3 Paxos group for split 1
4 Replicas participate in voting for writes to achieve a majority quorum (2 out of 3) and serve reads
Leader manages writes for that split
All replicas serve reads
5 Different splits can have leaders in different zones
How does Spanner provide «global consistency?»
SPANNER GLOBAL CONSISTENCY
TrueTime
Synchronizes clocks in all machines across datacenters
Google's Global Network
Fast & redundant

MySQL Galera

- Очень давно на рынке
- Настоящий мультимастер
- Активно развивается и продвигается коммьюнити
- Куча документации
- Много вариантов развертывания, мониторинга, миграции и т.д.



https://galeracluster.com/

Зачем?

- acid и атомарные транзакции
- распределенные транзакции и блокировки
- ограниченное вертикальное масштабирование
- дорогое вертикальное масштабирование
- 5-ть 9-к
- гео-резервирование
- гео-доступность
- окно на обслуживание

Как? bidirectional replication (полная копия данных), распределение информации (shards)

А как же PostgreSQL?

- если коротко - все плохо ;)
- community не выработало единый подход
- драйв только силами отдельных компаний
- кто то еще занимается upstream
- а кто то уже нет

Вопросы?

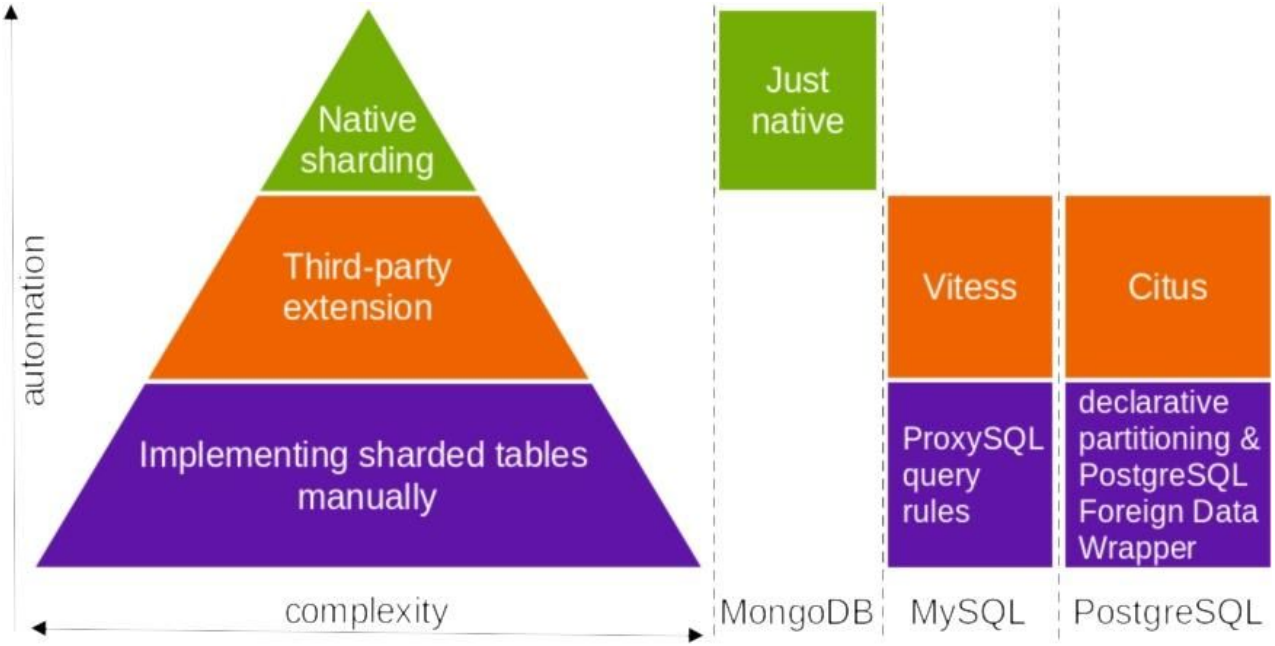
Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Способы горизонтального масштабирования PostgreSQL

Ходили в горы и упали

- pg_dtm
- postgres-x2
- postgres-xl



Кто сейчас заметен на рынке?

На логическую репликацию смысла смотреть нет:

- ее много
- вся она достаточно разная
- требует очень аккуратной работы
- вряд ли кто то будет ее использовать

Решения PostgreSQL:

- https://www.2ndquadrant.com/en/ - BDR, Postgres-XL ?
- https://www.citusdata.com/
- https://bucardo.org/

Кто сейчас заметен на рынке?

PostgreSQL like

- https://www.cockroachlabs.com/ - бывшие разработчики спаннера
- https://www.yugabyte.com/
- https://docs.yugabyte.com/latest/comparisons/ - сравнение кластеров
- https://greenplum.org/
- https://arenadata.tech/products/arenadata-db/

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Вернемся к обсуждению

Когда не нужно думать о масштабировании?

- Все данные помещаются в RAM
- Проект находится на стадии MVP

Когда приходят мысли о масштабировании?

Недостаточно текущих ресурсов:

- Вычислительные ресурсы (CPU, RAM)
- Ресурсы хранения (Disk Space)
- Отказоустойчивость на уровне ДЦ

Когда масштабирование нужно?

- Исчерпали ресурс оптимизации запросов
- Исчерпали ресурс схемы данных
- Исчерпали инфраструктурные ресурсы (hardware, software)

Streaming Replication

Самый простой и частый способ масштабирования. Потоковая репликация – это когда у нас есть мастер. Мастер "стримит" журналы транзакций на реплики. Все просто.

- Всегда есть в качестве hot standby
- Доступен для read-only трафика
- Часто недооценен

Поэтому, когда мы говорим о масштабировании, нужно начинать именно с реплик

Streaming Replication. Проблемы

- Лаг репликации - чтобы решать эту проблему, мы должны переключиться на уровень приложения
- Балансировка трафика - один поток на запись, который нужно отправить на мастер и большой поток чтения, который нужно отправить на реплику.
  - infrastructure-based
    - Если в инфраструктуре используется какие-то DCS системы - Consul, Etcd, DNS
    - Использование отдельных балансировщиков - Haproxy, Confd, Consul-Template
    - Keepalived или VRRP
  - application-based

Streaming Replication. Где и как используем?

- Почти всегда. У всех есть hot standby - переносим на него часть нагрузки
- Реже отельный read-трафик, аналитические запросы
- Резервное копирование

Streaming Replication. Недостатки и особенности

- Лаг репликации (вакуумы, bulk-операции, остановка wal replay)
- Recovery conflicts - отмена долгих запросов на реплике
- Bloat на мастере, см. hot_standby_feedback

Итог:

- *Мастер на запись - запись ограничена мощностью одного узла*
- *Реплики на чтение - масштабирование*

Declarative partitioning - много данных

- Оперативные данные
- Холодный архив

С 10-ой версии PostgreSQL, которая вышла в 2017 году, появилось декларативное партиционирование. Раньше партиционирование делалось на триггерах.

Не относится напрямую к масштабированию

Declarative partitioning. Когда?

- Хорошо для неизменяемых данных
- Легко управлять местом
- Архив можно унести
- FDW - foreign data wrappers

Если мы можем разбить данные на секции, то мы можем эти данные как-то отдельно заархивировать, отдельно сгрузить, положить в другое хранилище.

Declarative partitioning. Где и как?

- Исторические данные
- Оперативные данные
- Данные из S3
- Архивы

Когда партиционирования недостаточно

- FDW
- Declarative partitioning
- **Scale-up**

Путь для сильных духом

- Несвязанные репликацией узлы
- Мастер узел содержит partitioned таблицу
- Leaf узлы доступны на запись
- Каждая таблица - foreign table

Проблемы:

- *Пока нет инструментов для управления этим всем*
- *По большей части все руками*
- *Издержки для HA*
- *Не лучшая производительность*

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Практика

Практика

- Архитектура 21.1.6
- Настроим через кластер кубера
  - Orchestrate CockroachDB in a Single Kubernetes Cluster (Insecure)
- IMPORT INTO
- Ручное раскатывание в GCE
- Ключи для старта
- Обработка параллельных транзакций
- Github

ДЗ

Домашнее задание

1 вариант:

- Развернуть CockroachDB в GKE или GCE
- Потесировать dataset с чикагскими такси
- Или залить 10 Гб данных и протестировать скорость запросов в сравнении с 1 инстансом PostgreSQL
- Описать что и как делали, с какими проблемами столкнулись

```
cockroach sql --insecure --host=localhost:26257 --database="taxi" --execute="IMPORT INTO taxi_trips CSV DATA ('gs://hw18/taxi_trips.csv.000000000000') WITH delimiter = ',', nullif = '';", skip = '1'
```
- для пропуска заголовка

Домашнее задание

2 вариант:

- Переносим тестовую БД 10 Гб в географически распределенный PostgeSQL like сервис
- Описать что и как делали и с какими проблемами столкнулись

Выполнение ДЗ:

- 1 из 2 частей - 10 баллов
- 2 из 2 – 20 баллов
- + 2 балла за красивое решение
- - 2 балла за рабочее решение, и недостатки указанные преподавателем не устранены

Список материалов для изучения

1. PostgreSQL Scaling Usecases. Алексей Лесовский
2. Автоматическое масштабирование БД в Kubernetes для MongoDB, MySQL и PostgreSQL

Цели вебинара

1. Иметь представление горизонтальном масштабировании PostgreSQL
2. Понимать когда стоит использовать горизонтальное масштабирование, а когда нет
3. Уметь настраивать параллельный кластер

Рефлексия

Рефлексия

С какими впечатлениями уходите с вебинара?

Как будете применять на практике то,
что узнали на вебинаре?

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Алексей Железной

Data Engineer

- 4 года опыта Инженером данных/аналитиком (Python, Airflow, Clickhouse, Greenplum)
- 2 года опыта преподавания в OTUS (курсы DWH Analyst/DE/PostgreSQL Cloud)

LinkedIn