ClickHouse для инженеров и архитекторов БД

Область применения и первое представление

otus.ru

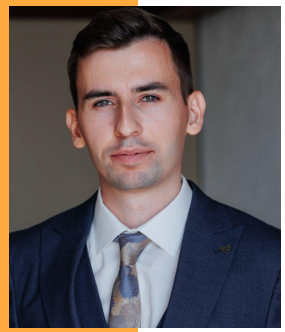
Проверить, идет ли запись

Меня хорошо видно && слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Область применения и первое представление



Алексей Железной

Senior Data Engineer/Architect
Магистратура - ФКН ВШЭ

*Руководитель курсов **DWH Analyst, ClickHouse для инженеров и архитекторов БД, Greenplum для разработчиков и архитекторов баз данных** в OTUS*

*Преподаватель курсов **Data Engineer, DWH Analyst, PostgreSQL** и пр. в OTUS*

LinkedIn

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе
#**OTUS ClickHouse**

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

Что такое ClickHouse

Возможности

Терминология

Сообщество

Рефлексия

Цели вебинара

К концу занятия вы сможете

1. познакомиться с ClickHouse
2. рассмотреть возможности ClickHouse
3. научиться искать ответы на свои вопросы

Смысл

Зачем вам это уметь

1. выявлять потребность
2. избегать не оптимальные сценарии использования
3. получать поддержку в решении проблем

Что такое ClickHouse

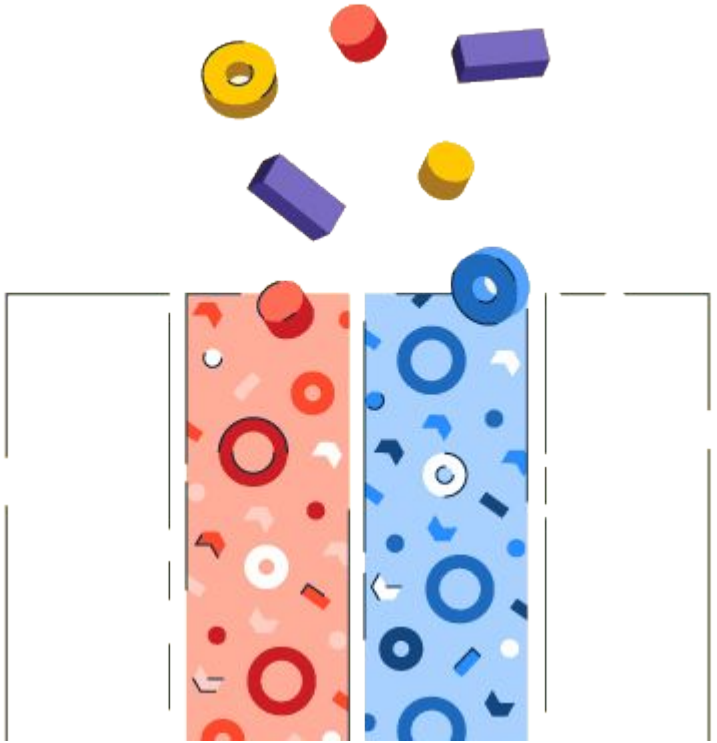
ClickHouse - аналитическая колоночная система управления базами данных реального времени (column-based OLAP DBMS)

Аналитическая
(analytic)

Область применения:

- годовые/квартальные/месячные отчеты
- выявление закономерностей
- изучение аудитории и таргетинг
- обучение ML-моделей
- хранение метрик и логов

Колоночная
(column-based)



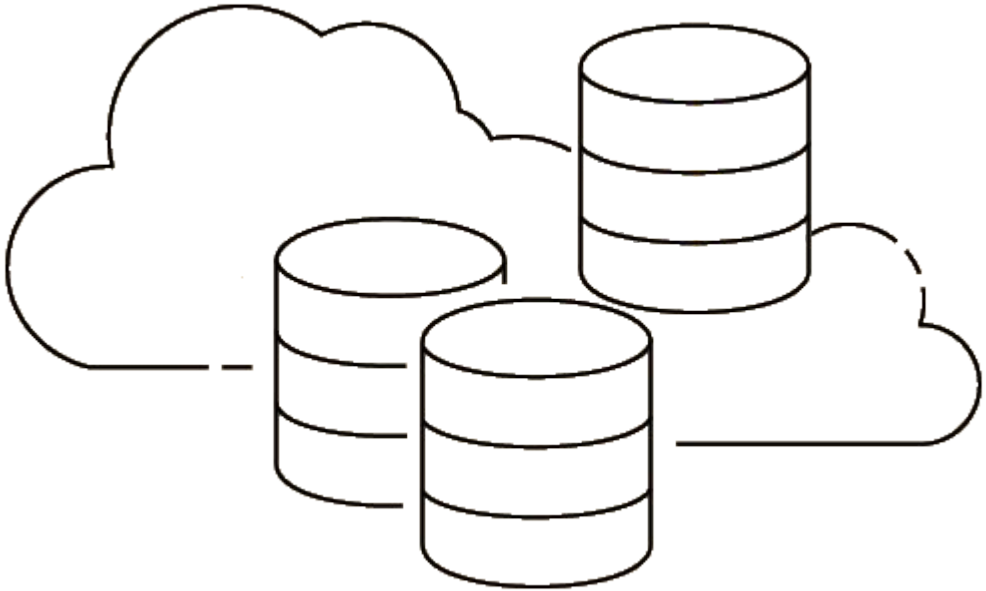
Особенности:

- данные хранятся столбцами, не строками
- эффективное хранение и выборка отдельных столбцов
- широкие таблицы
- нормализация - зло
- удаление/изменение данных - дорого

Система управления базами данных
(СУБД/DBMS)

Понятие:

- создавать и управлять несколькими БД
- можно просто говорить «база данных», вас поймут
- практически все БД в настоящее время являются СУБД

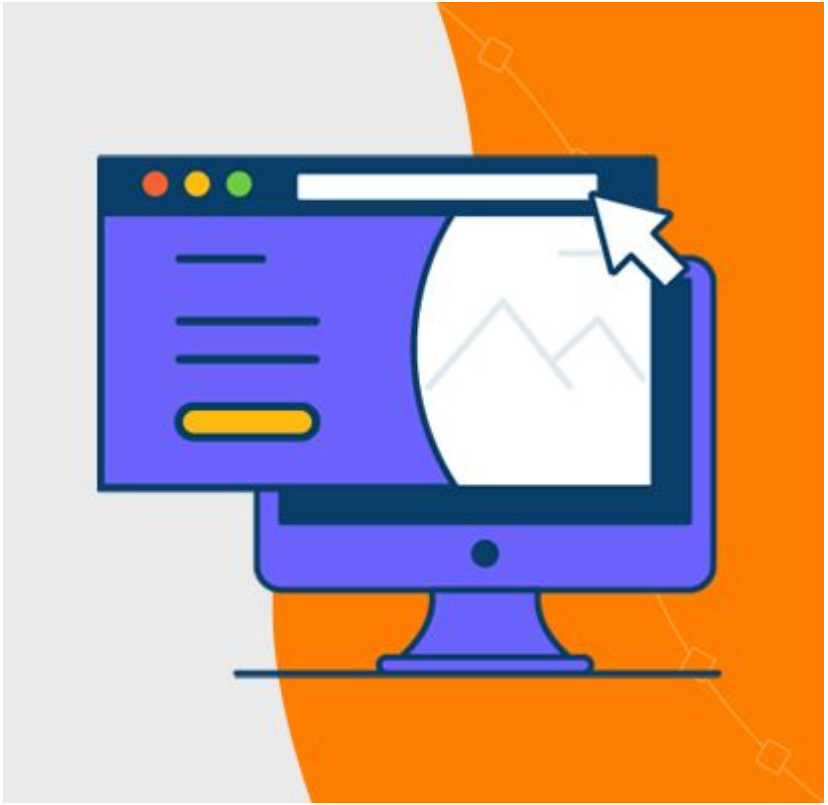


Базы данных *отличаются от* ***СУБД*** *тем, что сами по себе представляют лишь файл на компьютере. Базы данных не умеют ничего делать с этими данными — только хранить. А вот СУБД уже предоставляют возможности по манипуляции ими.*

Реального времени
(online)

Определение:

- лучше говорить online, в русском языке есть более популярное понятие, называемое так же
- данные запрашиваются, готовятся и выводятся за один запрос
- достаточное время ответа для интеграции на сайт



Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Возможности

горизонтальное масштабирование
многоуровневое хранение
высокая пропускная способность
приближенные вычисления
интеграция с другими системами
преобразование данных

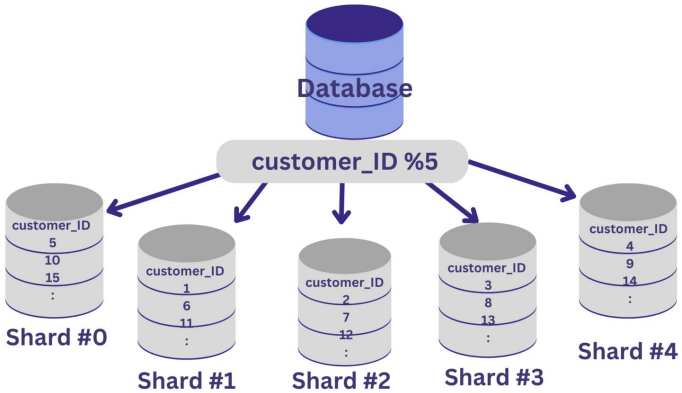
Горизонтальное масштабирование

поддерживается:

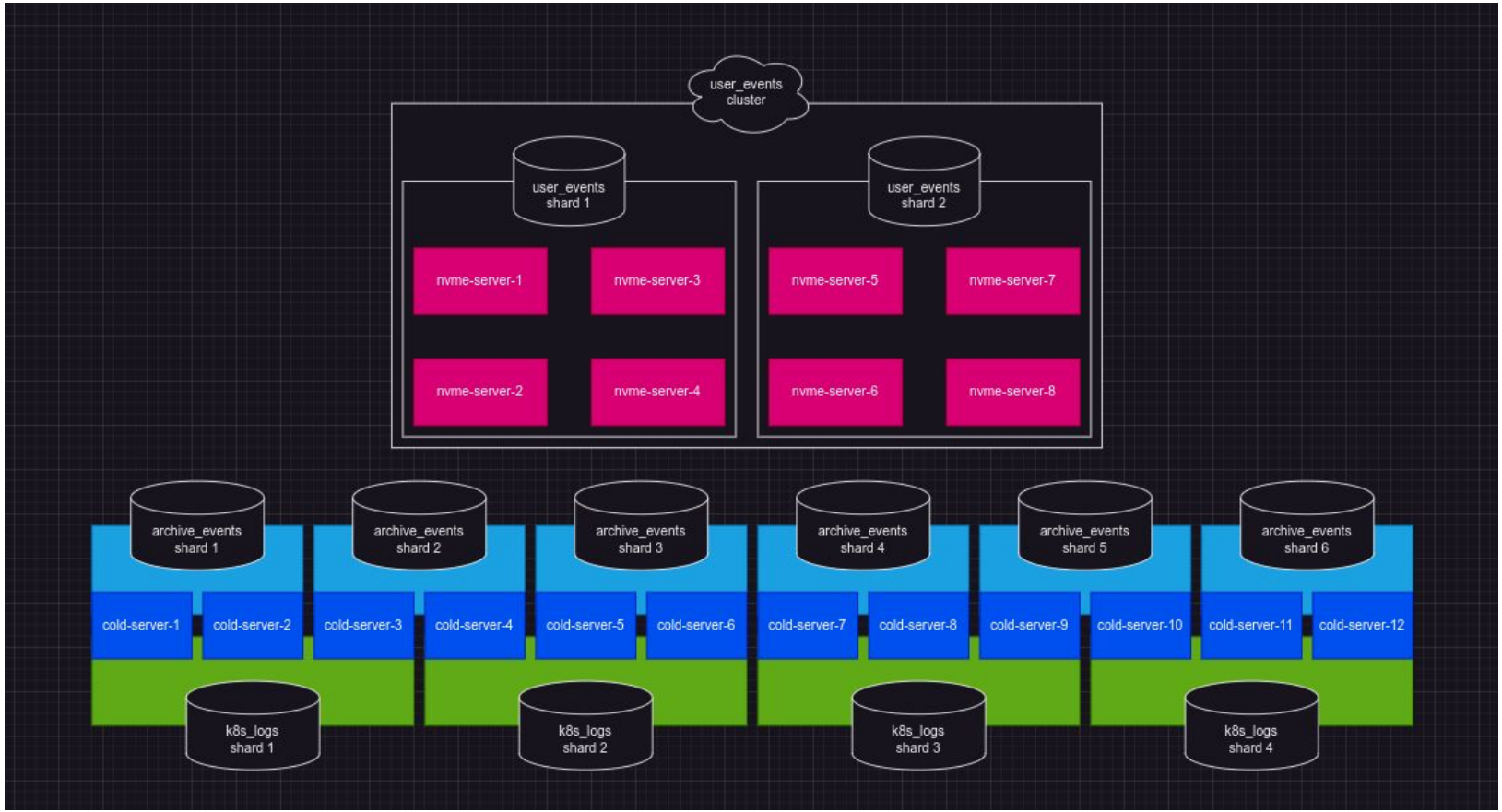
- добавление/удаление шардов
- добавление/удаление реплик

особенность:

- кластер это топология, описанная в конфигурации
- можно описать несколько топологий
- сервер может быть частью нескольких кластеров



Сервер может быть частью нескольких кластеров



remote_servers.xml

```xml
<remote_servers>

<user_events>
  <shard>
    <replica>  <host>nvme-server-1</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-2</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-3</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-4</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>nvme-server-5</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-6</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-7</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-8</host>  <port>9000</port>  </replica>
  </shard>
</user_events>

```

```
<archive_events>
  <shard>
    <replica>  <host>cold-server-1</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-2</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-3</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-4</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-5</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-6</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-7</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-8</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-9</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-10</host> <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-11</host> <port>9000</port>  </replica>
    <replica>  <host>cold-server-12</host> <port>9000</port>  </replica>
  </shard>
</archive_events>
```

```
<k8s_logs>
  <shard>
    <replica>  <host>cold-server-1</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-2</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-3</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-4</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-5</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-6</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-7</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-8</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-9</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-10</host> <port>9000</port>  </replica>
    <replica>  <host>cold-server-11</host> <port>9000</port>  </replica>
    <replica>  <host>cold-server-12</host> <port>9000</port>  </replica>
  </shard>
</k8s_logs>
```

```
<all_events>
  <shard>
    <replica>  <host>nvme-server-1</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-2</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-3</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-4</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>nvme-server-5</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-6</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-7</host>  <port>9000</port>  </replica>
    <replica>  <host>nvme-server-8</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-1</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-2</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-3</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-4</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-5</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-6</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-7</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-8</host>  <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-9</host>  <port>9000</port>  </replica>
    <replica>  <host>cold-server-10</host> <port>9000</port>  </replica>
  </shard>
  <shard>
    <replica>  <host>cold-server-11</host> <port>9000</port>  </replica>
    <replica>  <host>cold-server-12</host> <port>9000</port>  </replica>
  </shard>
</all_events>

</remote_servers>
```

Многоуровневое хранение storage_policy

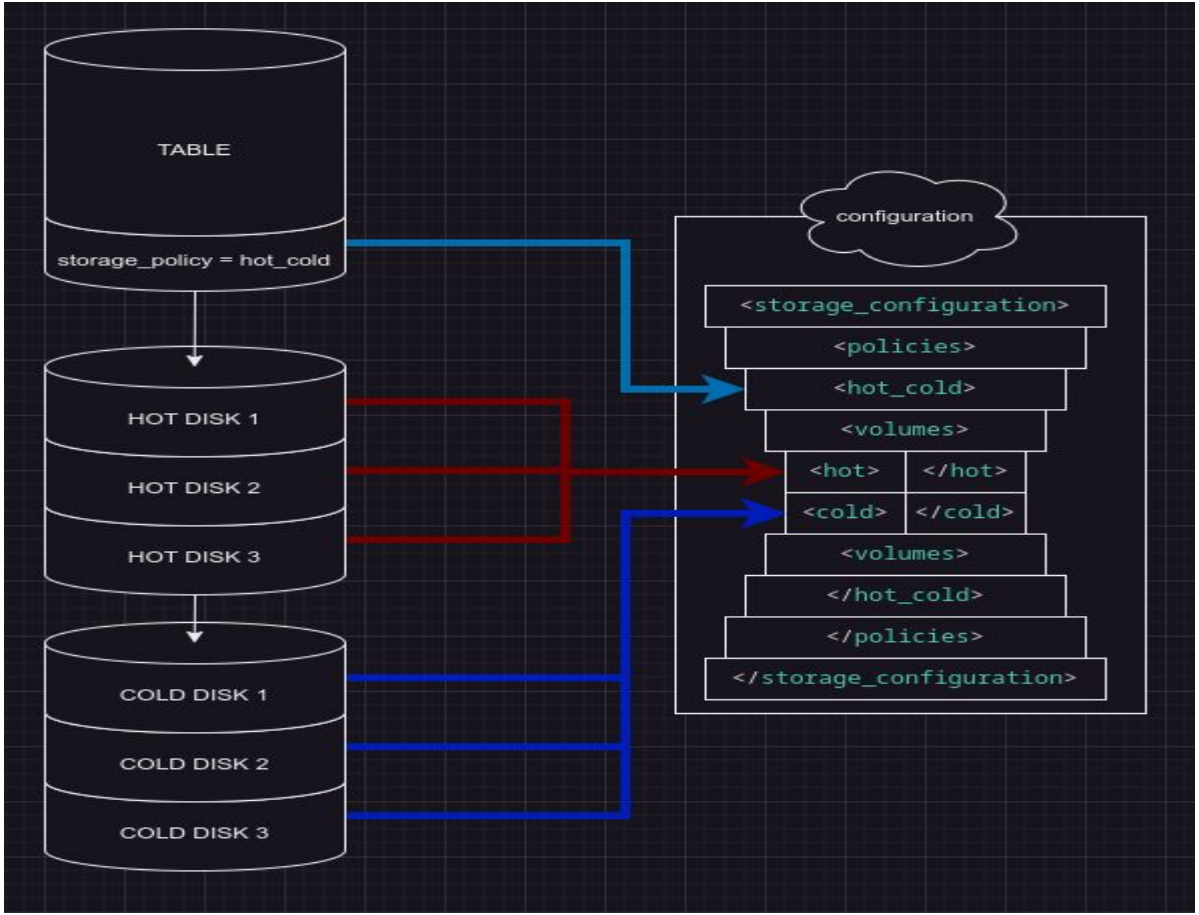
- можно описать несколько «storage_policy»
- storage_policy состоит из набора сущностей «volume»
- volume состоит из набора сущностей «disk»
- можно назначить storage_policy на таблицу
- volume имеют приоритет, данные перемещаются на нижестоящие по мере заполнения вышестоящих

поддерживаются облачные диски:

- s3 / gcs / azure / hdfs / readonly web

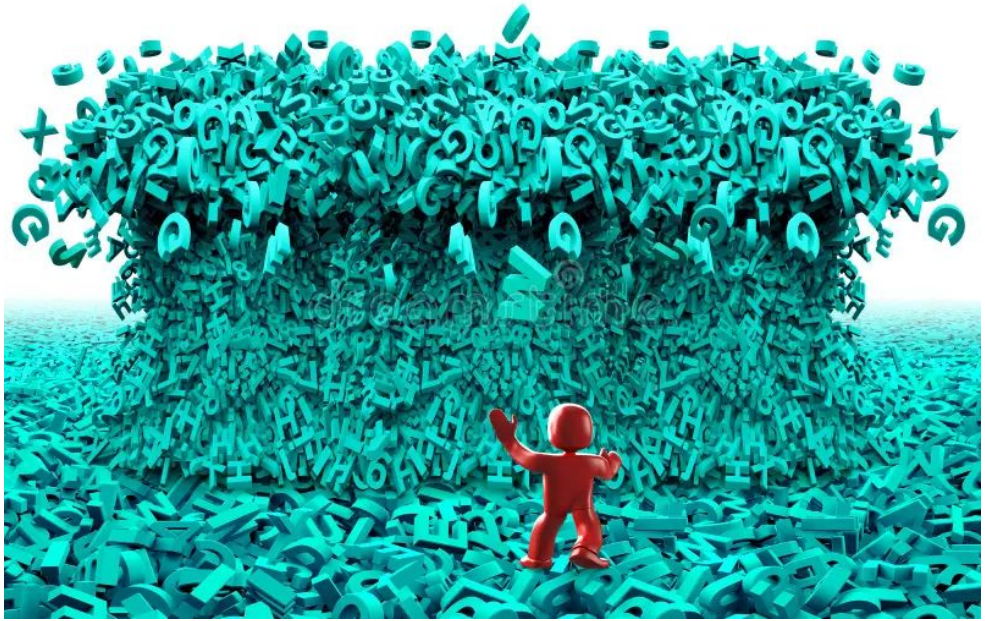
Пример использования storage_policy:

- для таблицы задана storage_policy hot_cold
- в конфигурации hot_cold описана как 2 volume, hot и cold
- порядок указания volume имеет значение как приоритет
- данные будут записываться на HOT DISK (1,2,3), ClickHouse будет вытеснять старые данные на COLD DISK (1,2,3)



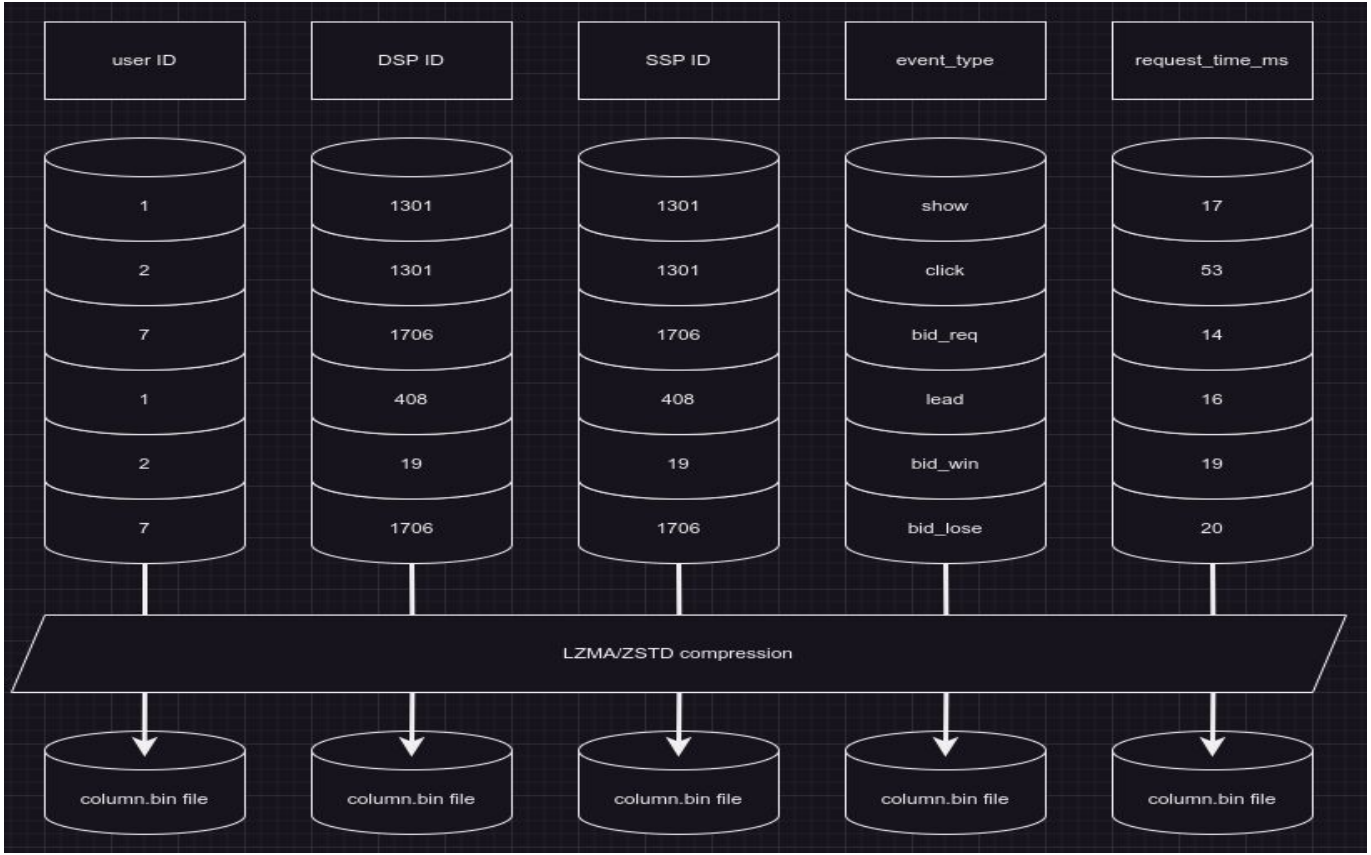
Высокая пропускная способность

- колоночное хранение обеспечивает высокую степень сжатия любыми кодеками
- на простых запросах достигается выдача данных на скорости дисковой системы умноженной на уровень сжатия
- низкая производительность на множестве точечных запросов



Колоночное хранение обеспечивает высокую степень сжатия любыми кодеками

- однородные данные хранятся рядом
- кардинальность чаще низкая, чем высокая
- даже на быстрых алгоритмах достигается сжатие в десятки раз



На простых запросах достигается выдача данных на скорости дисковой системы умноженной на уровень сжатия

- допустим у нас есть дисковая подсистема, способная выдавать на чтение 200 MB/s
- и мы храним метрики, которые сжимаются до x40
- представим что кто-то пришел с запросом «покажи мне ВСЕ метрики»
- он получит 200x40MB/s=8GB/s поток данных, или хотя бы сколько позволит сеть

Низкая производительность на множестве точечных запросов

- важно не количество запросов в секунду, а их конкурентность
- по умолчанию количество тредов на запрос равно количеству ядер
- 10К тредов - предел
- на сервере с 64 ядрами, предел конкурентных запросов 10000/64 = 156.25 одновременных запросов

Приближенные вычисления

функции:

- uniq
- квантили
- медианы
- линейная/стохастическая регрессия

особенности:

- семплинг
- группировка по первым N ключам

Интеграция с другими системами

С кем работает ClickHouse:

- облачные файловые системы: s3 / gcs / azure / hdfs
- базы данных: postgresql / mysql / mongo / rocksdb / redis
- брокеры сообщений: rabbitmq / kafka / NATS
- hive / iceberg / hudi / deltalake / sqlite

Кто работает с ClickHouse:

- redash / grafana / tableau / superset / vector
- есть официальные библиотеки для основных языков: python / golang / java / C++
- а так же неофициальные для многих других

Преобразования данных

материализованные представления (MV):

- механизм для преобразования данных

- реализован как преобразование данных, поступающих в INSERT, со складываемым в другую таблицу результатом

проекции:

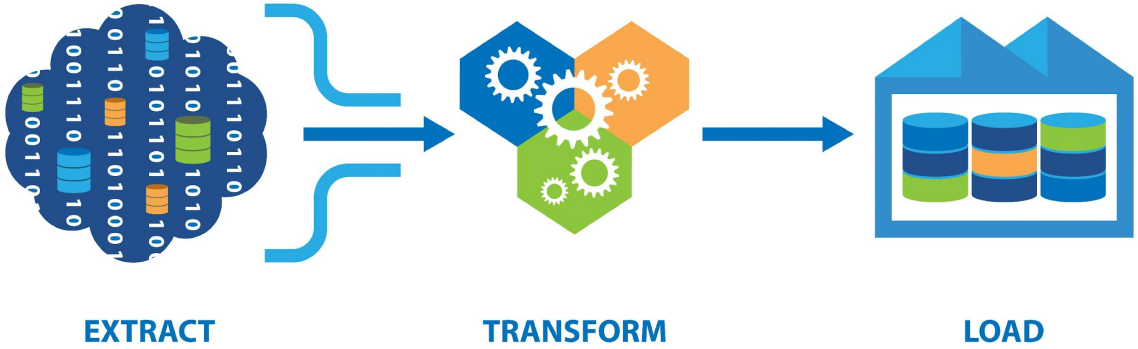
- альтернативная реализация MV, сохраняющая данные в саму таблицу, для представления данных с иным ключом

таблицы, поддерживающие фоновую агрегацию данных:

- удаление дубликатов

- суммирование по primary ключу

- и более сложная логика на функциях агрегации



Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Терминология

ENGINE

DATABASE ENGINE

TABLE ENGINE

Подвид базы данных или таблицы, отличающийся в реализации, в применимых запросах, в назначении.

Бывают семейства MergeTree и специальные.

Например, TABLE Engine=ReplacingMergeTree удаляет дубликаты по основному ключу. TABLE Engine=Kafka специальный Engine для интеграции с Kafka.

MergeTree

Основной ENGINE, одноименный с алгоритмом, сердце ClickHouse. Применяется чаще всего, на нём основана большая часть ENGINE, а именно MergeTree family, дополняющих классический MergeTree.

PART и PARTITION

PART - набор данных, создаваемый в результате INSERT. Хранятся каталогами на файловой системе. Внутри каталога по файлу данных и файлу засечек на каждый столбец, а также файл с primary-индексом, контрольные суммы и метаданные колонок.

PARTITION - набор PART-ов, принадлежащих одному значению PARTITION BY ключа, подлежащих объединению фоновыми операциями Merge.

Слияние / merge

Фоновая операция по объединению данных нескольких PART-ов в новый PART.

Ключи сортировки и семплинга

В дополнении к привычному в других системах PRIMARY KEY и PARTITION KEY, так же есть:
ORDER BY KEY - ключ, по которому сортируются данные в PART.
SAMPLE BY KEY - ключ для семплирования.

Разреженный индекс

Данные в колонках хранятся гранулами - по index_granilarity строк на гранулу, по умолчанию index_granilarity=8123. Данные отсортированы по PRIMARY KEY. Индексируются min и max значения гранул, а не каждое значение.

Теневые копии / shadow

Артефакт резервного копирования. Набор PART-ов вне data-каталога ClickHouse, содержащий hardlink-и.

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Сообщество

Чат / telegram

https://t.me/clickhouse_ru - официальный чат ClickHouse, самое популярное и активное место для получения поддержки и ответа на вопросы

Документация

https://clickhouse.com/docs - место, в которое отправляют в чате, когда не хотят отвечать на простые вопросы

GITHUB

https://github.com/ClickHouse/ClickHouse - код ClickHouse, а так же issue-трекер, с актуальными проблемами

Рефлексия

Цели вебинара

Проверка достижения целей

1. познакомиться с ClickHouse
2. рассмотреть возможности ClickHouse
3. научиться искать ответы на свои вопросы

Вопросы для проверки

1. К каким классам систем относится ClickHouse?
2. Какую проблемы вы бы решили используя ClickHouse, а какую не стали бы им решать?
3. Где можно получить помощь по ClickHouse и куда сообщать о багах?

Рефлексия

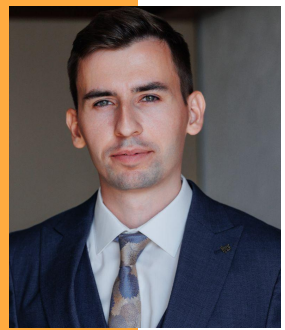
С какими впечатлениями уходите с вебинара?

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

Senior Data Engineer/Architect
Магистратура - ФКН ВШЭ

*Руководитель курсов **DWH Analyst, ClickHouse для инженеров и архитекторов БД, Greenplum для разработчиков и архитекторов баз данных** в OTUS*

*Преподаватель курсов **Data Engineer, DWH Analyst, PostgreSQL** и пр. в OTUS*

LinkedIn