ClickHouse для инженеров и архитекторов БД

MergeTree и типы данных

otus.ru

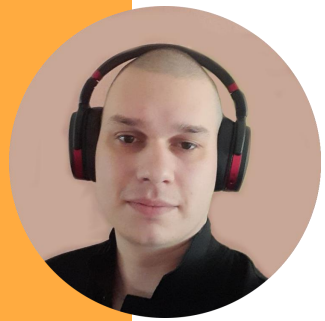
Проверить, идет ли запись

Меня хорошо видно && слышно?

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

Тема вебинара

MergeTree и типы данных



Константин Трофимов

Senior SRE / ClickHouse DBA в VK

Занимаюсь эксплуатацией ClickHouse с первых версий: 5 лет в VK, до этого в AdNow, до этого занимался Vertica. Сотни серверов, десятки кластеров, десятки петабайт данных.

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в учебной группе
#OTUS ClickHouse-2024-08

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

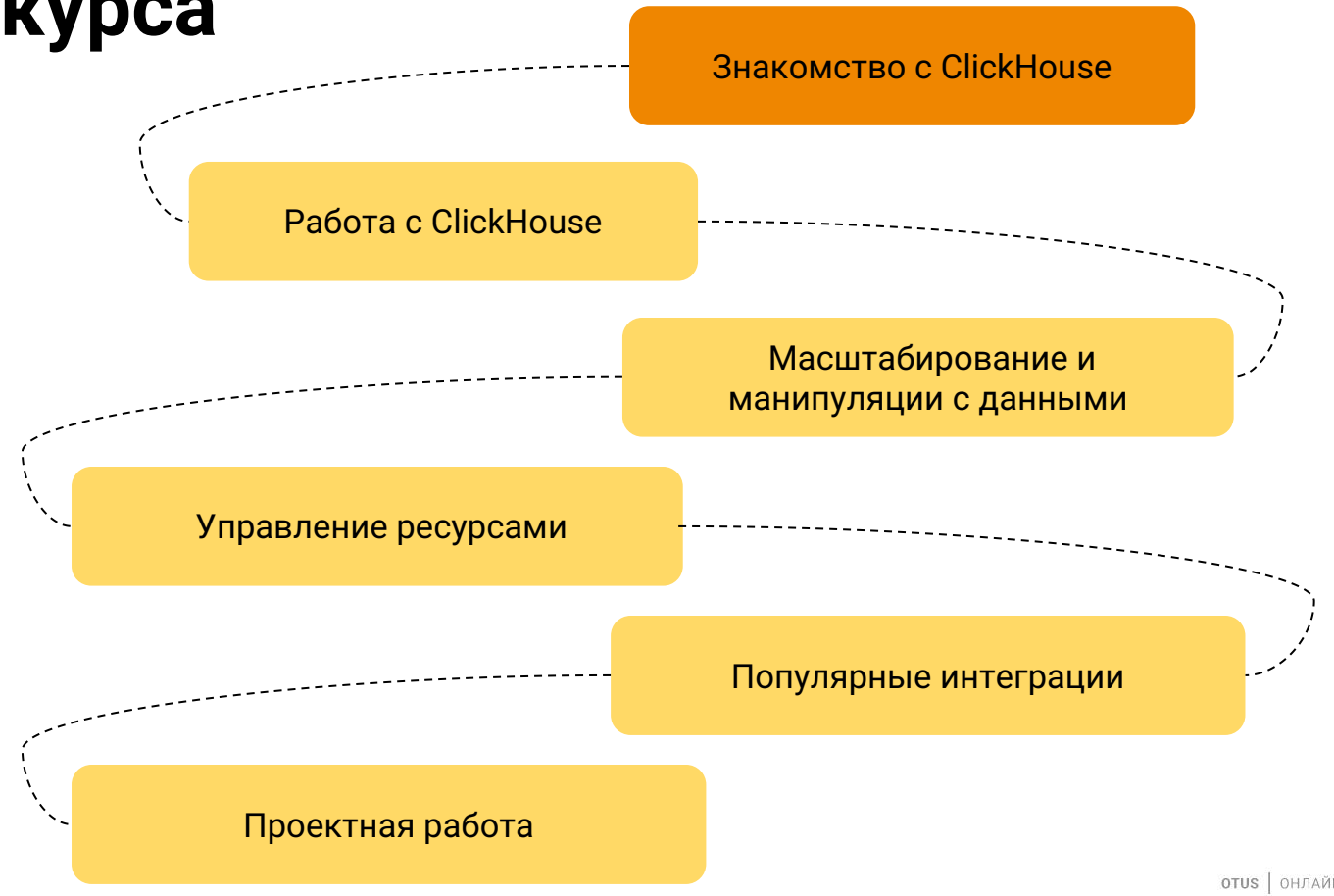
Пишем в чат

Говорим голосом

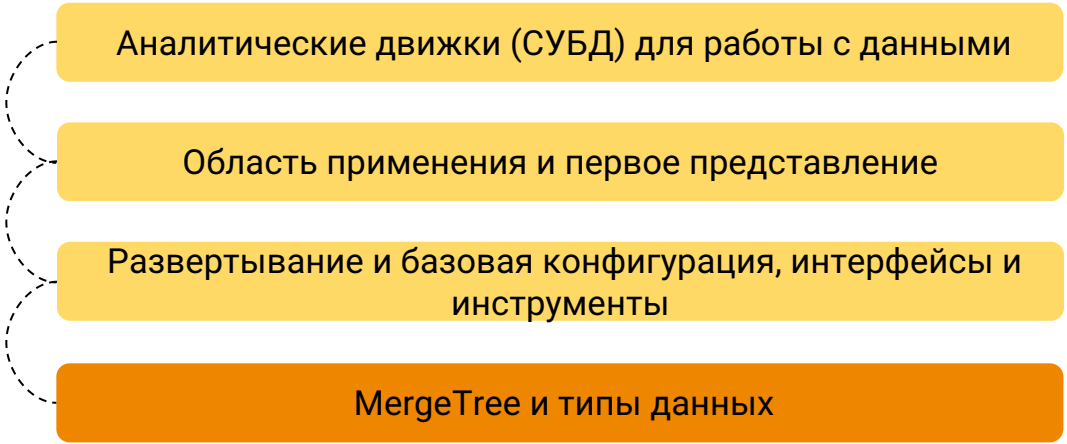
Документ

Ответьте себе или
задайте вопрос

Карта курса


Знакомство с ClickHouse
Работа с ClickHouse
Масштабирование и манипуляции с данными
Управление ресурсами
Популярные интеграции
Проектная работа

Темы модуля «Знакомство с ClickHouse



Маршрут вебинара

Цели вебинара

К концу занятия вы сможете

1. правильно выбирать типы данных
2. правильно выбирать Primary Key и Partition Key

Смысл

Зачем вам это уметь

1. оптимально использовать дисковое пространство, повышая производительность
2. эффективно выбирать данные из ClickHouse

MergeTree

Алгоритм

MergeTree - алгоритм объединения данных, вставленных частями.

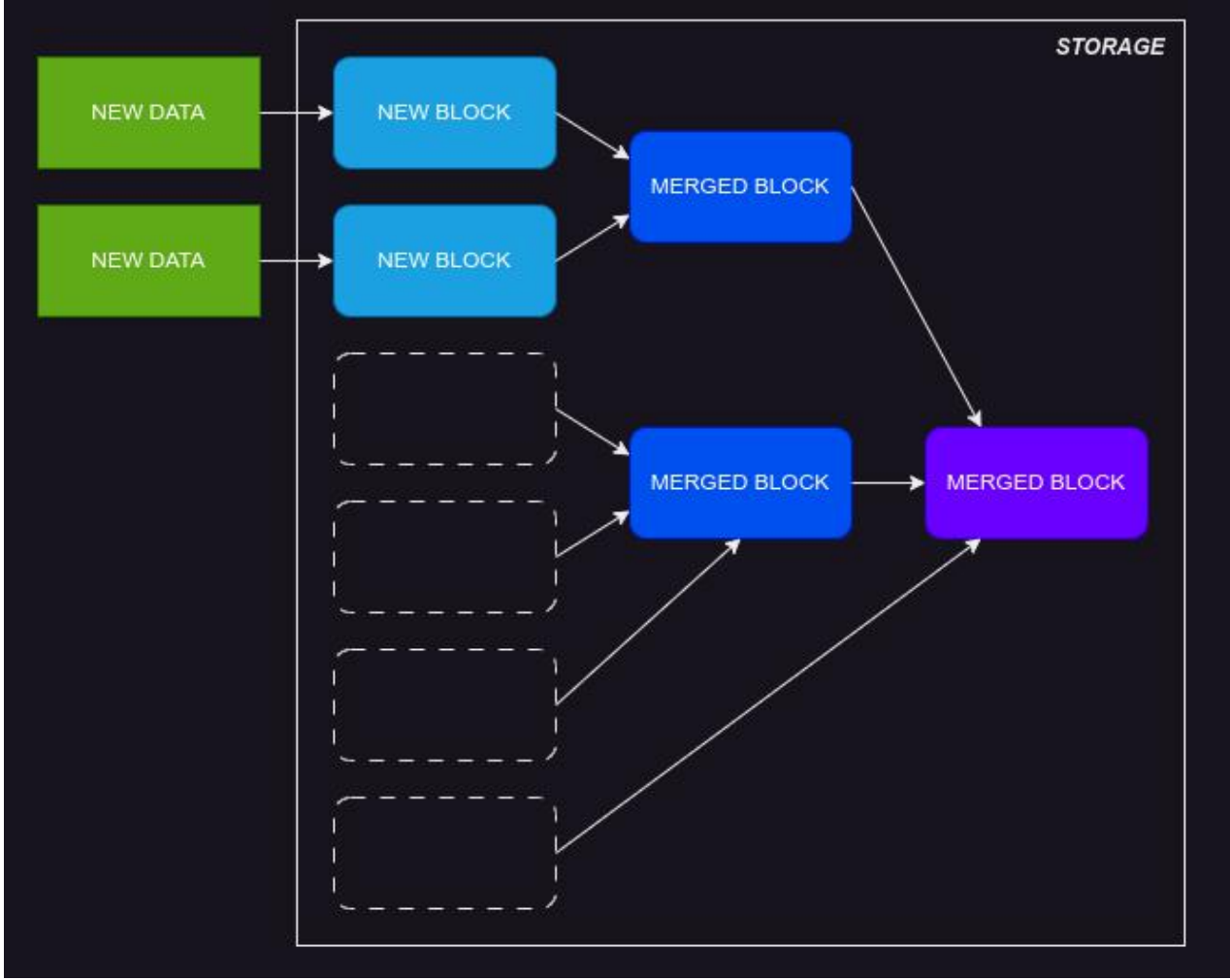
Новые данные при добавлении на хранение, добавляются блоками.

Блоки объединяются в фоновом режиме, в более крупные блоки.

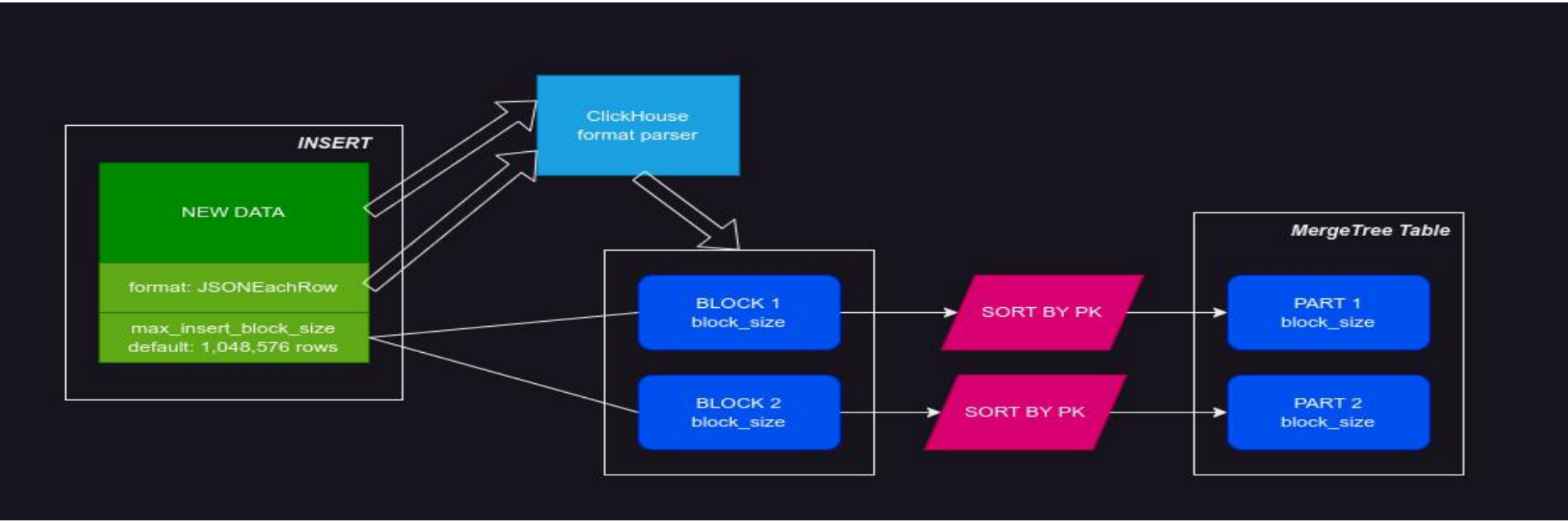
В ClickHouse такой блок называется Part.

Объединение называется Merge.

Концепция визуально напоминает дерево.



Реализация



1) Данные поступают на вход парсера ClickHouse

2) Данные нарезаются на блоки и сортируются по Primary Key

3) Данные сохраняются в таблице отсортированными блоками, такие блоки называются Part

4) Фоном происходит объединение блоков (Merge), объединяемые Part-ы так же сортируются между собой в новый Part

Структура Part

Именование

PARTITION ID - строковое представление значения ключа партиционирования

min_block - порядковый номер блока, для каждого нового блока уникальный

max_block - изначально равный min_block, после слияний будет номером максимального блока учавствовашего в слиянии

level - количество слияний, произошедшее для образования Part-а

Примеры

20240101_1_1_0

Part, образованый самым первым блоком поступившим в таблицу

20240101_1_2_1

Part, полученый слиянием Part-ов 20240101_1_1_0 и 20240101_2_2_0

20230101_1012_7238_18

Part, полученный в результате 18 слияний, разных партов с 1012 номера блока по 7238

Структура Part

Содержание

primary.idx

хранит значение Primary Key для каждой «засечки» - адреса на диске в файле data.bin для колонки.

data.mrk3

хранит сами «засечки», т.е. номер засечки и адрес в файле data.bin, для каждой колонки, для каждой гранулы

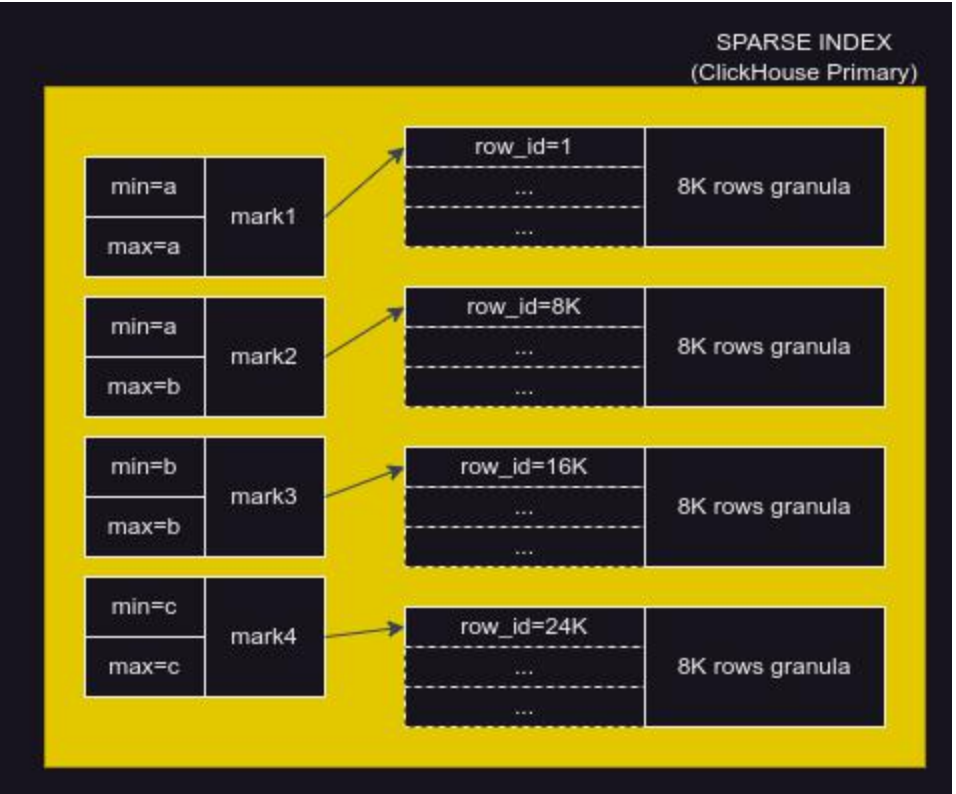
data.bin

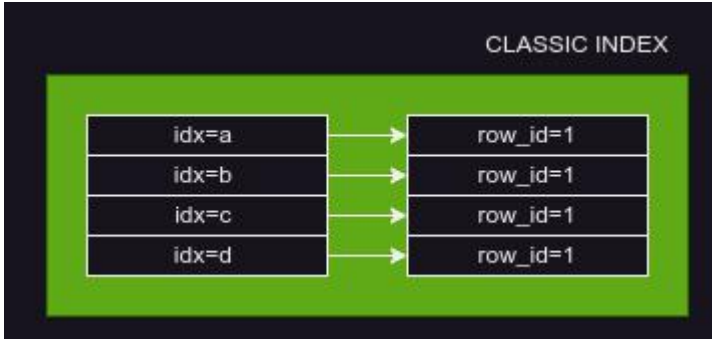
хранит сами данные колонок

Primary Key - minmax index

Основная задача индекса - эффективная выборка данных, прочитать с диска только те данные, которые были запрошены.

С этой целью индексы постоянно держатся в оперативной памяти.



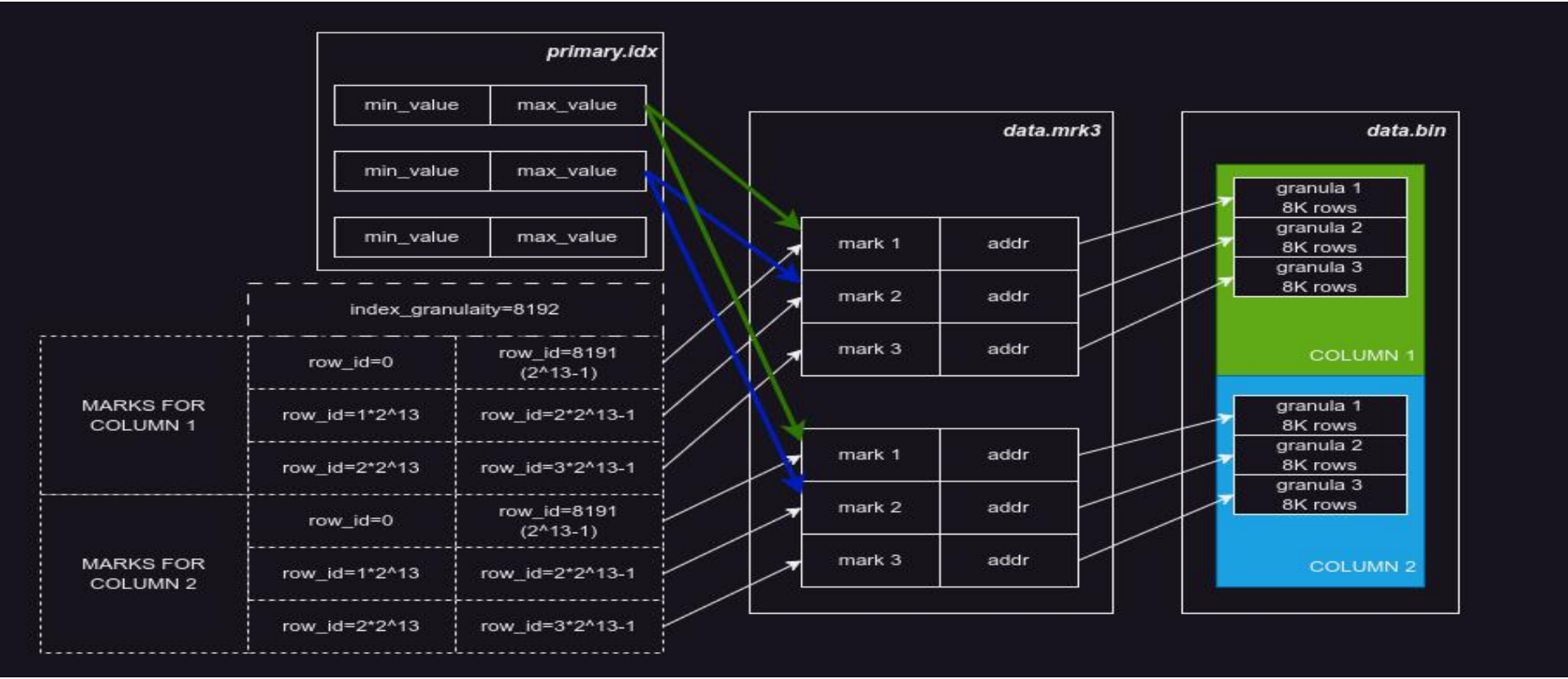


В ClickHouse классические индексы, в силу объема данных, не применимы. Значение Primary Key для каждой строки, при работе с десятками терабайт данных на жестком диске, может занимать сотни гигабайт в оперативной памяти, кроме того, проход всех элементов такого индекса будет дорогим по CPU.

Поэтому в ClickHouse применяется в качестве Primary Key разряженный индекс, так же называемый minmax индекс, так же называемый sparse индекс.

Индексы в ClickHouse позволяют выбирать не конкретные строки, а гранулы по index_granularity строк.

Primary Key и гранулярность - разреженный индекс



Значения хранятся гранулами по index_granularity строк

для каждой гранулы каждого столбца есть засечка с адресом на диске

для каждой засечки хранится значение PrimaryKey

Partition key

Partition Key это не более чем логическое ограничение на Merge Part-ов.

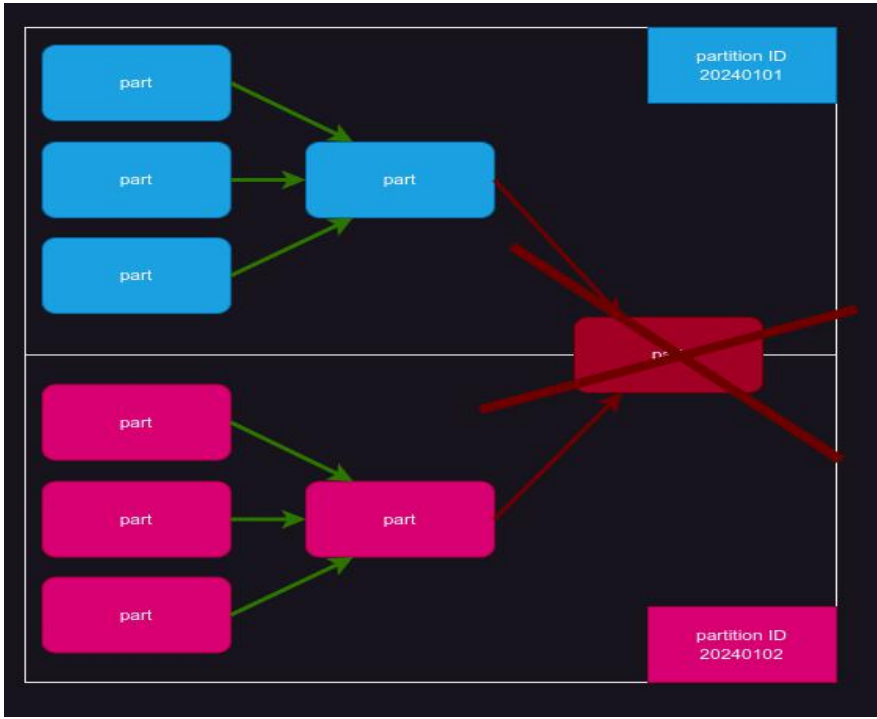
Part-ы с разным Partition Key не объединяются.

Так же Partition Key используется для массового удаления данных.

Поскольку в ClickHouse точечное удаление данных реализовано как мутация всего Part-а, удобно ограничить Merge партов логической границей, чтобы потом удалять данные целыми партами, большой пачкой, соответствующей конкретному значению Partition Key.

ClickHouse умеет выполнять поиск по Partition Key, однако список функций не документирован, и поведение от версии к версии меняется.

Не следует использовать Partition Key в качестве индекса, для этого существует Primary Key.



Спойлер

ClickHouse умеет выполнять дополнительные преобразования с данных, во время операции Merge. Под разные преобразования есть разные модификации движка MergeTree, все вместе движки называются MergeTree Family (тема урока 3 модуля 2).

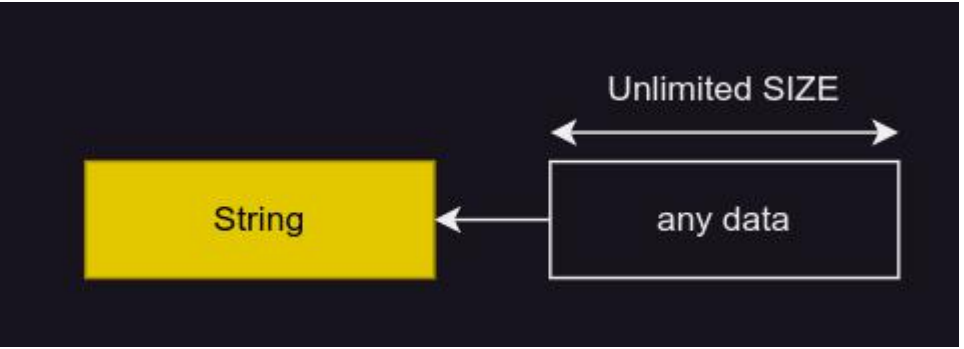
Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

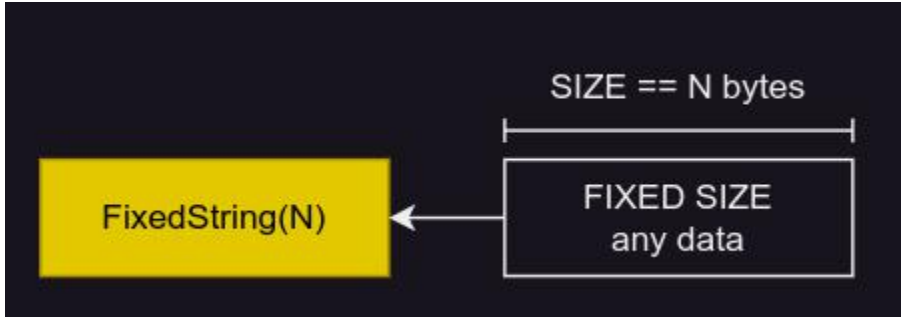
Типы данных

Строковые



String

- размер не ограничен
- можно писать произвольные данные, включая нулевые байты



FixedString(N)

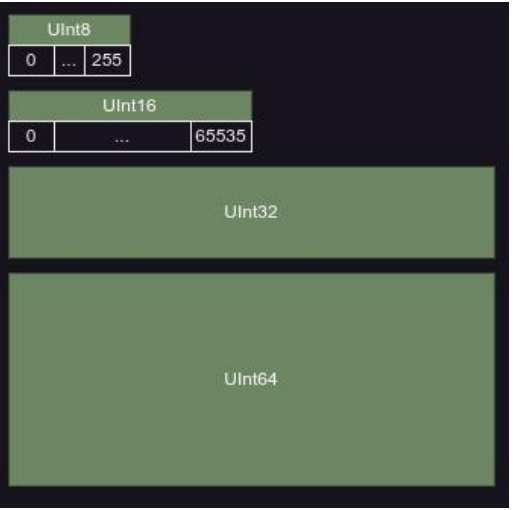
- так же произвольные данные
- размер ограничивается N байт, не символов
- недостающие байты заполняются нулевыми байтами - ‘\0’

Например можно применять для:

FixedString(16) для MD5

FixedString(32) для SHA256

Числовые



UInt8, UInt16, UInt32, UInt64, UInt256

- беззнаковые
- диапазон значений: 0 .. 2^N-1

например:
UInt8 0 .. 255
UInt16 0 .. 65535

Int8, Int16, Int32, Int64, Int128, Int256

- знаковые

диапазон значений: - 2^(N-1) .. 2^(N-1)-1

например:
UInt8 -128 .. 127
UInt16 -32768 .. 32767

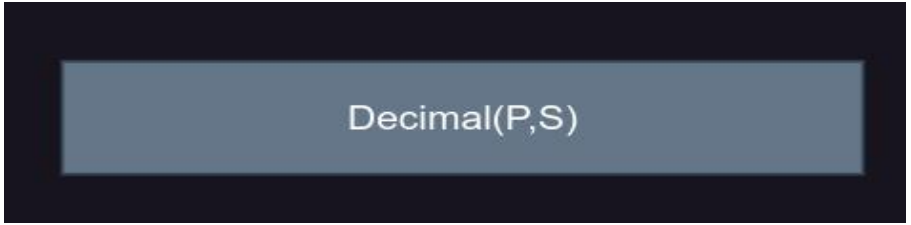
Числовые

Float32, Float64

- с плавающей запятой, аналоги float и double из языка C

от -1*number*2^exponent

до +1*number*2^exponent



Decimal(P, S)

- знаковые дробные, при делении лишние знаки отбрасываются, не округляются

P - сколько всего знаков

S - сколько из них дробных

алиасы:

Decimal32(S) для P=1..9

Decimal64(S) для P=10..18

Decimal128(S) для P=19..38

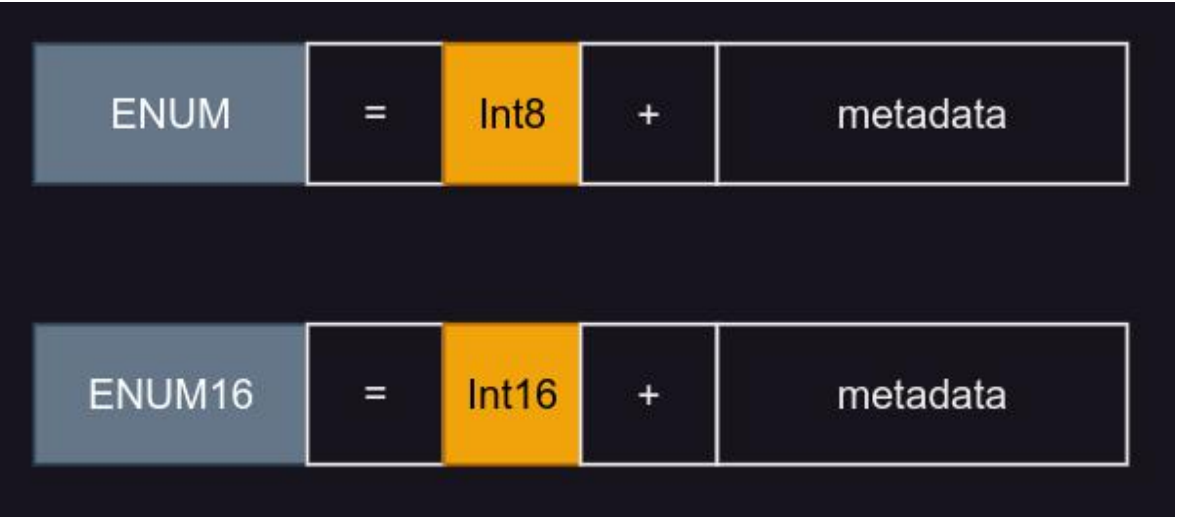
Decimal256(S) для P=39..76

диапазон: -1 * 10^(P - S) .. 1 * 10^(P - S)

Bool и Enum

Bool

- хранится как UInt8, принимает значения 0 и 1
- можно вставлять строки ‘false’ и ‘true’



Enum и Enum16

- хранится как Int8 или Int16 для Enum16

пример объявления:

user_type Enum('admin' = -128, 'user' = 0, ‘moderator’ = 1, ‘author’ = 127)

- можно вставлять строками согласно объявлению типа
- можно вставлять числами, нет проверки на корректность

Дата и время / Дата

Date и Date32

хранятся как UInt16 и UInt32

диапазоны:

Date 1970-01-01 .. 2149-06-06

Date32 1900-01-01 .. 2299-12-31

формат вставки

String ‘\d\d\d\d-\m\m-\d\d’

DateTime(timezone) и DateTime64(timezone,precision)

хранятся как UInt32 и UInt64

диапазоны:

DateTime 1970-01-01 00:00:00, 2106-02-07 06:28:15

DateTime64 зависит от precision

формат вставки

String ‘\d\d\d\d-\m\m-\d\d \H\H:\M\M:\S\S.[0..9]+’

можно вставлять как UInt32/UInt64

для DateTime будет интерпретироваться как Unix timestamp в UTC

timezone - не обязательный параметр, по умолчанию выбирается указанная в конфигурационном файле, вставка строкой будет интерпретирована как время в этой timezone

precision - сколько знаков после точки для долей секунды

Структуры

Array(T)

массив элементов типа **T**

например: SELECT toTypeName([1, 2, 3])

вернет: Array(UInt8)

выбрать элемент N: [...][N] (нумерация от 1!)

Tuple(T1, T2, …)

кортеж любых элементов

например: SELECT toTypeName((1, 2, 3))

вернет: Tuple(UInt8, UInt8, UInt8)

выбрать элемент N: (...).N (нумерация от 1!)

Именованые Tuple(a T1, b T2, …)

можно объявлять при создании таблицы

например: dsp_info Tuple(name String, id UInt16)

можно выбрать элемент N по имени:
dsp_info.name

Map(T1, T2, …)

пары ключ-значение, хранит несколько пар

например: SELECT toTypeName(CAST(arrayMap((x, y) -> (x, y), ['k1', 'k2'], [1, 2]), 'Map(String,UInt64)'))

вернет: Map(String, UInt64)

выбрать элемент по ключу: map_column[‘key’]

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Специальные типы данных

Nested

Вложенный тип данных

Объявляется при создании таблицы

```
(
  a String,
  b Nested (
    id UInt32,
    name String
  ),
  c DateTime
)
```

Раскрывается как при создании как:

```
(
  a String,
  b.id Array(UInt32),
  b.name Array(String),
  c DateTime
)
```

Расширенные

Типы данных, основанные на базовых, поддерживающие дополнительные функции для работы с ними

Point, основан на Tuple(Float64, Float64), применяется для географических данных

IPv4, основан на UInt32, поддерживаются функции работы с IP-адресами

IPv6, на FixedString(16), аналогично

JSON, хранится как динамически меняющийся Named Tuple, вставляется как String, для корректного отображения нужна настройка output_format_json_named_tuples_as_objects = 1

AggregateFunction(function, Type)

применяется для сохранения промежуточных результатов аггрегации в материализованных представлениях

Нельзя объявить явным образом. Можно объявить при создании таблицы.
Можно получить, добавив суффикс -State к аггрегационной функции, например:

SELECT uniqState(column)
FROM table

сохраняет промежуточный результат функции uniq

позднее можно получить результат функции uniq, объединив несколько промежуточных состояний. Для объединения используется суффикс Merge

Например:
select uniq(column) даст такой же результат, как и:
select uniqMerge(MyAggFunc) from (select uniqState(column) AS MyAggFunc from ...)

Interval

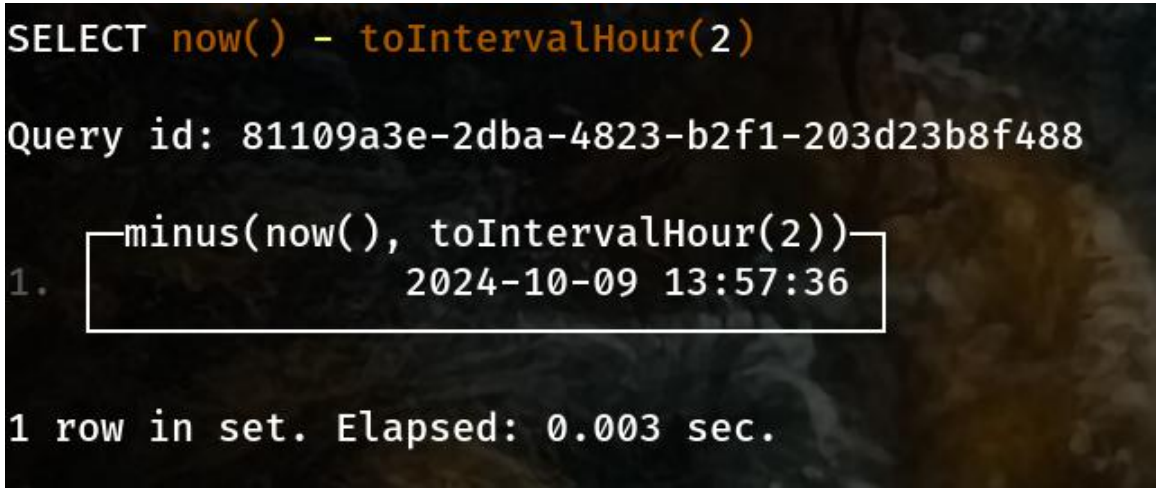
используется для упрощения выполнения арифметических операций с датой и временем

Объявляется как INTERVAL N UNIT, где:

N - количество

UNIT - единица времени (NANOSECOND/MICROSECOND/MILLISECOND/SECOND/MINUTE/HOUR/DAY/WEEK/MONTH/QUARTER/YEAR)

например INTERVAL 2 DAY



Алиасы для типов данных

используется для взаимодействия с другими системами

Для поддержки других систем, в частности работы с MySQL используются алиасы для типов данных.

Так, VARCHAR/MEDIUMBLOB/CHAR являются алиасами для String,
BIGINT к Int64,
SMALLINT к Int16.

Список таких трансляций можно почерпнуть в system.data_type_families таблице

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Рефлексия

Цели вебинара

Проверка достижения целей

1. правильно выбирать типы данных
2. правильно выбирать Primary Key и Partition Key

Вопросы для проверки

1.	какой тип данных вы выберете для хранения SHA256 от пароля пользователя
2.	какие поля из предложеных вы включите в индекс: date (Date), timestamp (DateTime), response_body (String), response_time_ms (UInt16)
	в каком порядке вы включите эти поля, можно приводить свои примеры

Рефлексия

С какими впечатлениями уходите с вебинара?

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

Следующий вебинар

14 октября 2024

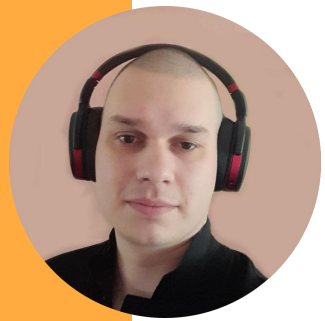
Язык запросов SQL

- Ссылка на вебинар будет в ЛК за 15 минут
- Материалы к занятию в ЛК — можно изучать
- Обязательный материал обозначен красной лентой

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары



Senior SRE / ClickHouse DBA в VK

Занимаюсь эксплуатацией ClickHouse с первых версий: 5 лет в VK, до этого в AdNow, до этого занимался Vertica. Сотни серверов, десятки кластеров, десятки петабайт данных.