

Разбираемся во внутреннем устройстве ClickHouse

Виталий Людвиченко



Знакомство

Виталий Людвиченко

› 2014-2016



› 2016-...



Задачи, для которых подходит ClickHouse

Есть поток событий

- › Действия пользователей на сайте
- › Показы рекламы
- › Финансовые транзакции
- › DNS–запросы
- › ...

Хотим сохранять эти события и делать из них какие-то выводы

Идеология ClickHouse

- › Интерактивные запросы по данным, обновляемым в реальном времени
- › Стараемся заранее ничего не агрегировать
- › Диалект SQL + расширения
- › Нужны очищенные структурированные данные

Типичный запрос в системе веб-аналитики

Считаем для счётчика топ-10 рефереров за неделю

```
SELECT Referrer, count(*) AS count
```

```
FROM hits
```

```
WHERE CounterID = 1234 AND Date >= today() - 7
```

```
GROUP BY Referrer
```

```
ORDER BY count DESC
```

```
LIMIT 10
```

Как выполнить запрос быстро?

Быстро вычисляем

- › Векторная обработка данных целыми блоками
- › Специализация и низкоуровневые оптимизации

Как выполнить запрос быстро?

Быстро читаем

- › Только нужные столбцы: CounterID, Date, Referer
- › Сжатие
- › Отсекаем (приблизительно) ненужные строки по индексу
- › Поддерживаем локальность чтения

Нужен индекс!

Выбираем так же, как в классических БД

› Большинство запросов будут содержать условия на CounterID и Date

(CounterID, Date) подойдёт

› Проверяем, мысленно упорядочив таблицу по выражению

Особенности

› Таблица действительно будет упорядочена по индексу

› Не обеспечивает уникальности

Как работает индекс

(CounterID, Date)

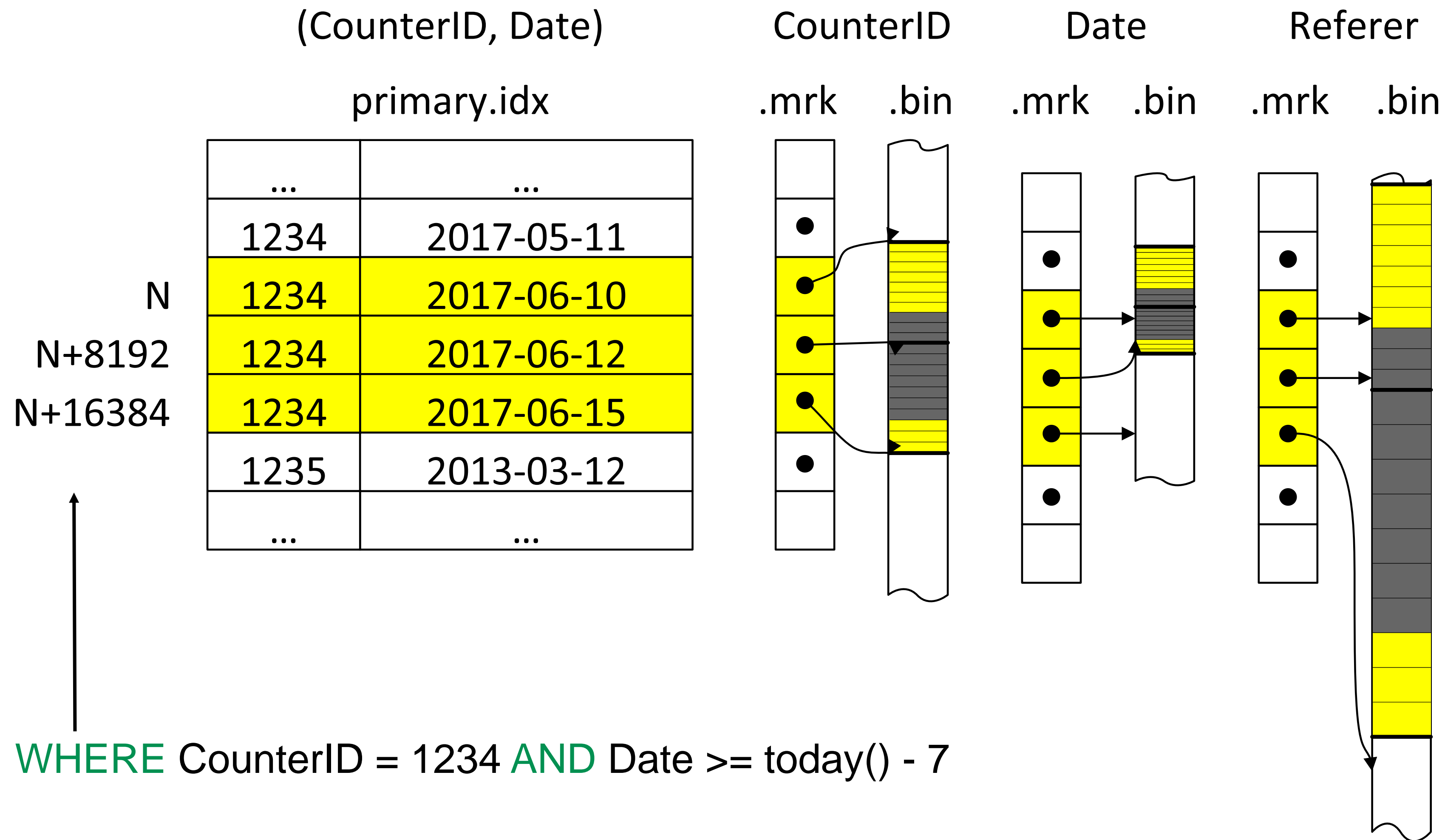
primary.idx

	1234	2017-05-11
N	1234	2017-06-10
N+8192	1234	2017-06-12
N+16384	1234	2017-06-15
	1235	2013-03-12

Одна запись на гранулу –
 index_granularity (=8192)
 последовательных строк таблицы

↑
 WHERE CounterID = 1234 AND Date >= today() - 7

Как работает индекс



Вставка данных

Как обеспечить упорядоченность?

- События поступают (почти) упорядоченными по времени

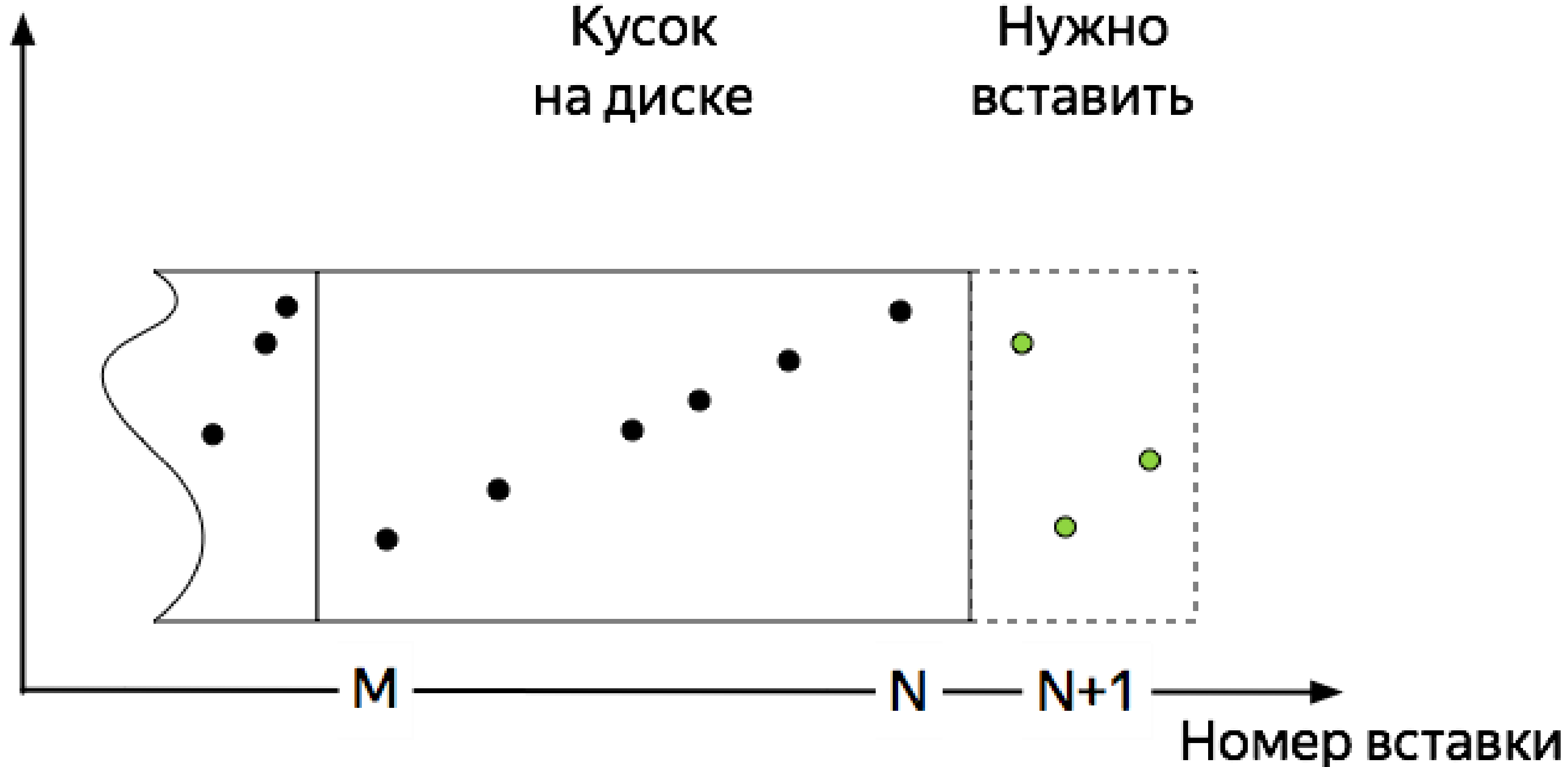
- › А нам нужно по первичному ключу!

- MergeTree: поддерживаем небольшое количество упорядоченных кусков

- › Идея та же, что и в LSM-tree

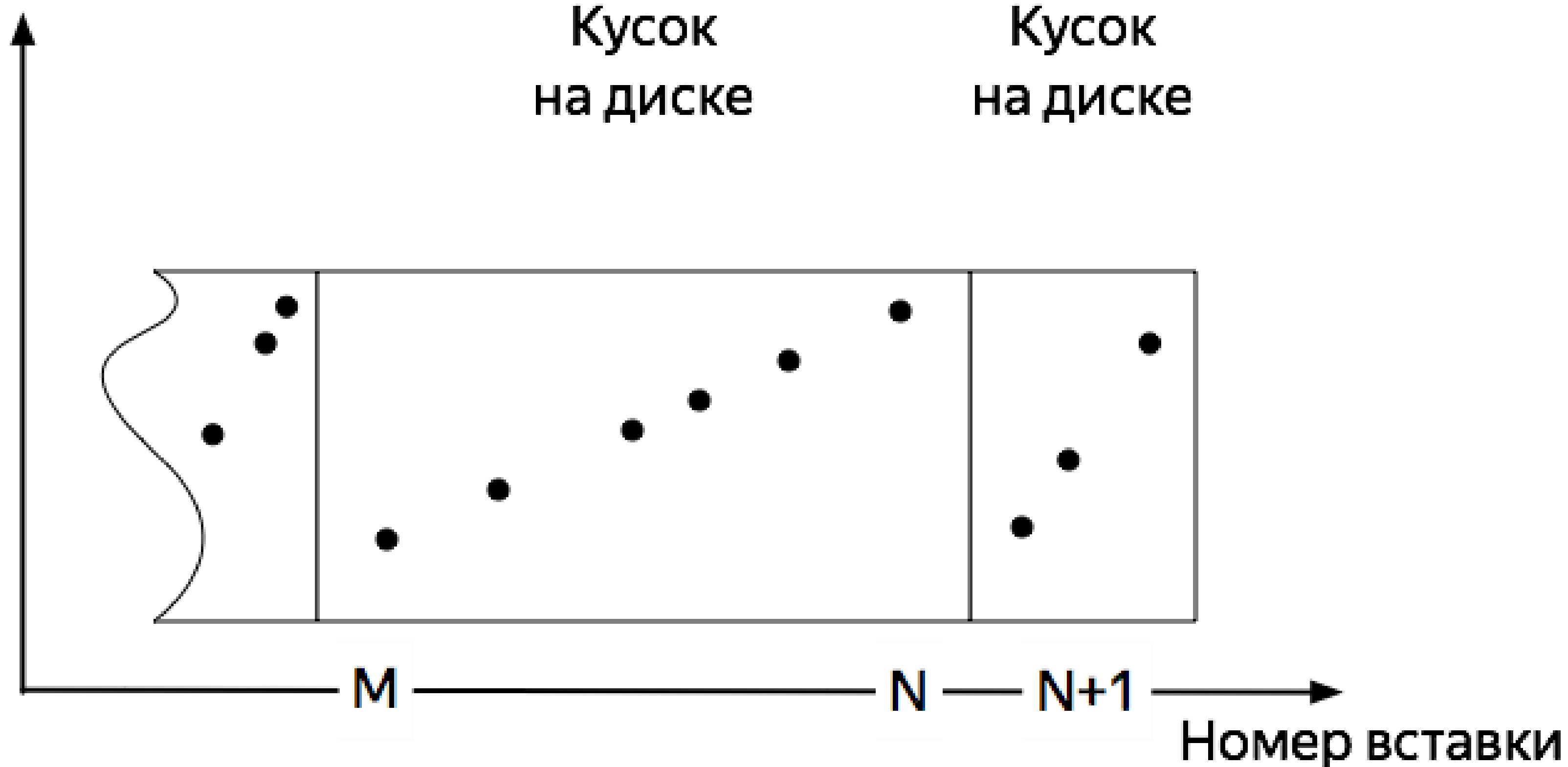
Как обеспечить упорядоченность

Первичный ключ



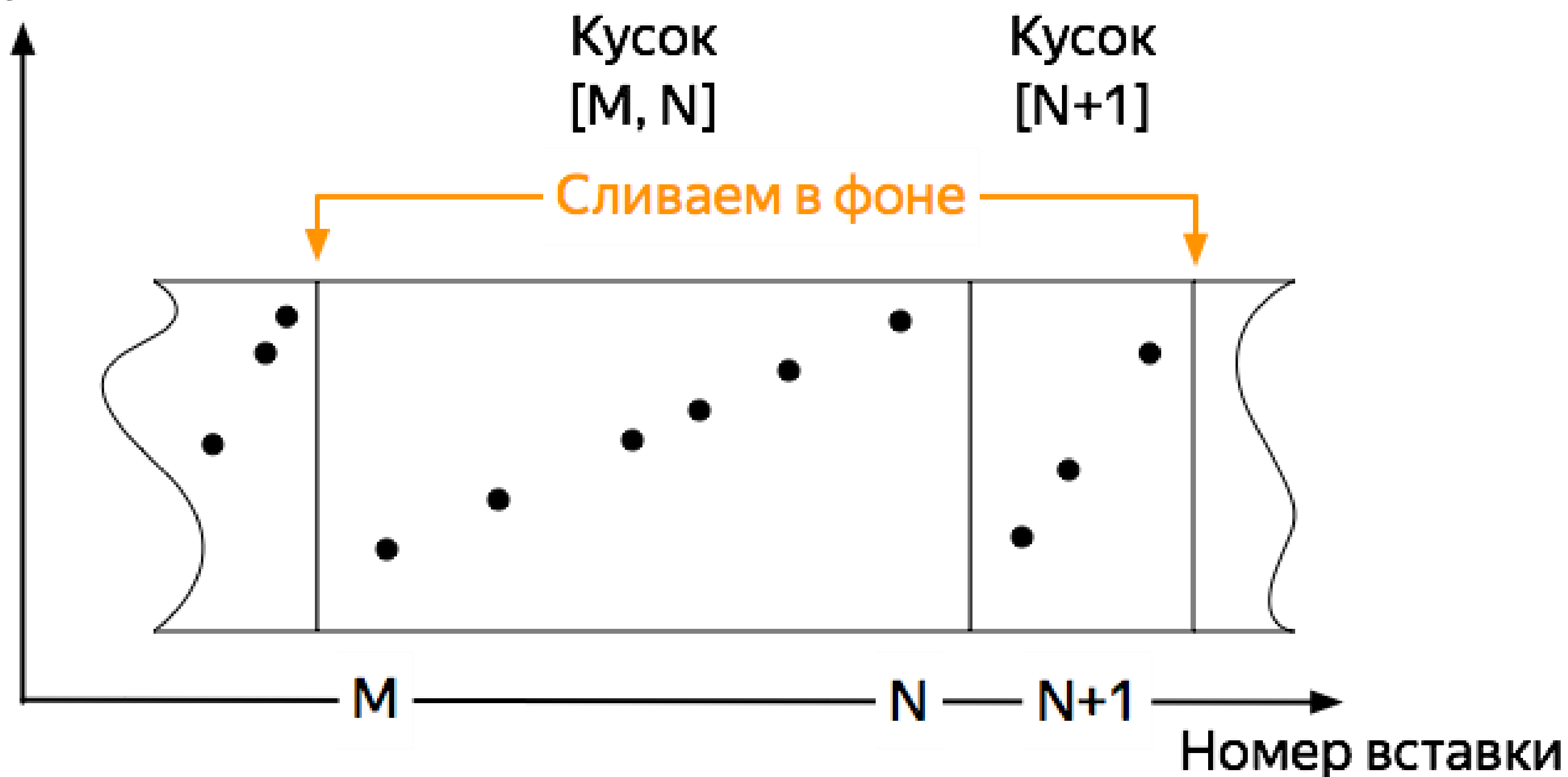
Как обеспечить упорядоченность

Первичный ключ



Как обеспечить упорядоченность

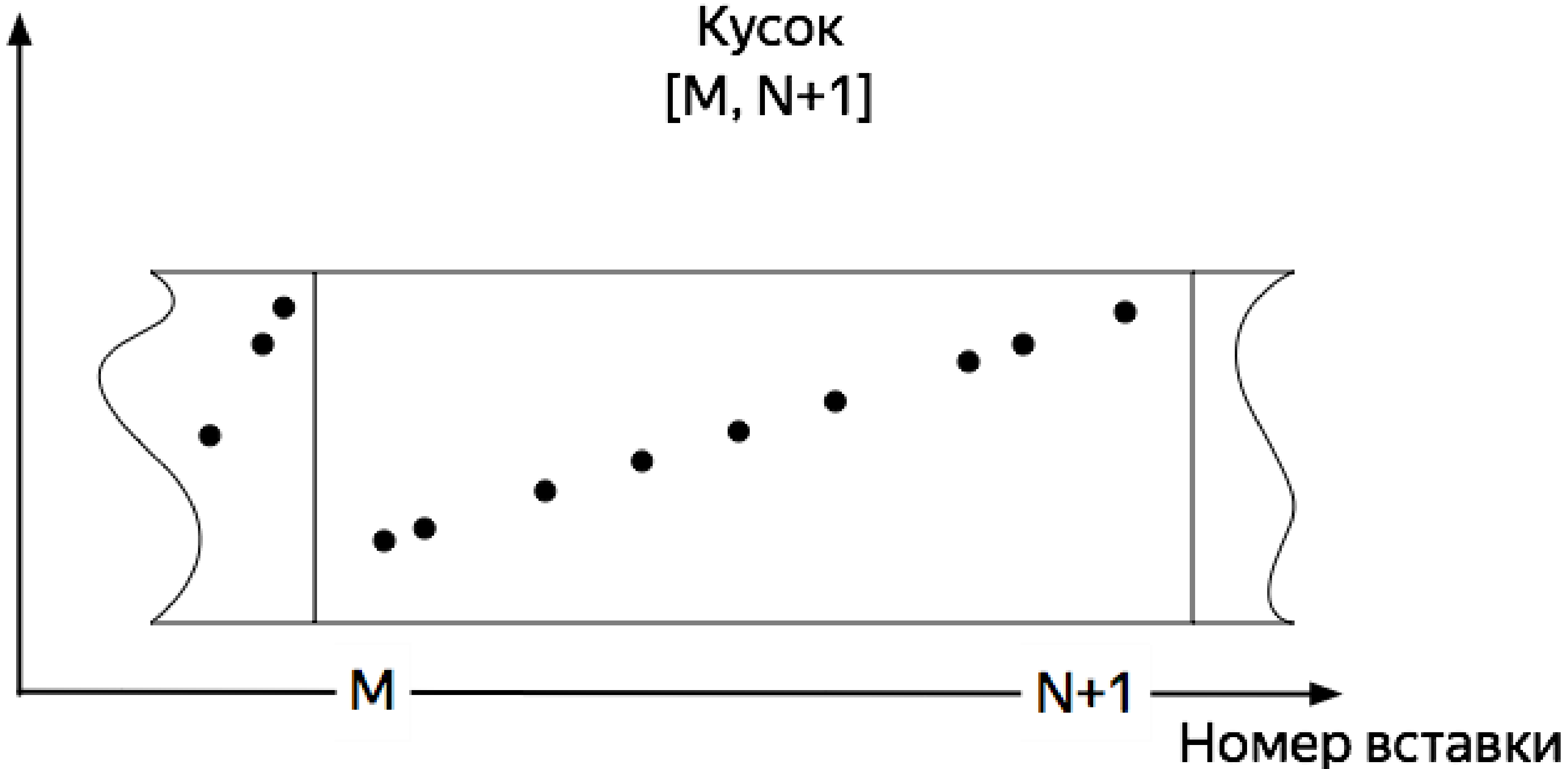
Первичный ключ



Как обеспечить упорядоченность

Первичный ключ

Кусок
[M, N+1]



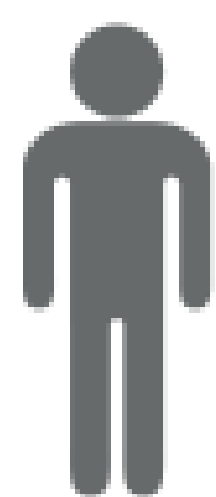
Когда одного сервера не хватает

- › Данные перестали помещаться на один сервер...
- › Хочется ещё ускориться, добавив железа...
- › Несколько одновременных запросов мешают друг другу...

Когда одного сервера не хватает

- › Данные перестали помещаться на один сервер...
 - › Хочется ещё ускориться, добавив железа...
 - › Несколько одновременных запросов мешают друг другу...
- ClickHouse: Шардирование + Distributed таблицы!

Чтение из Distributed таблицы



```
SELECT FROM distributed_table  
GROUP BY column
```

```
SELECT FROM local_table  
GROUP BY column
```



Чтение из Distributed таблицы



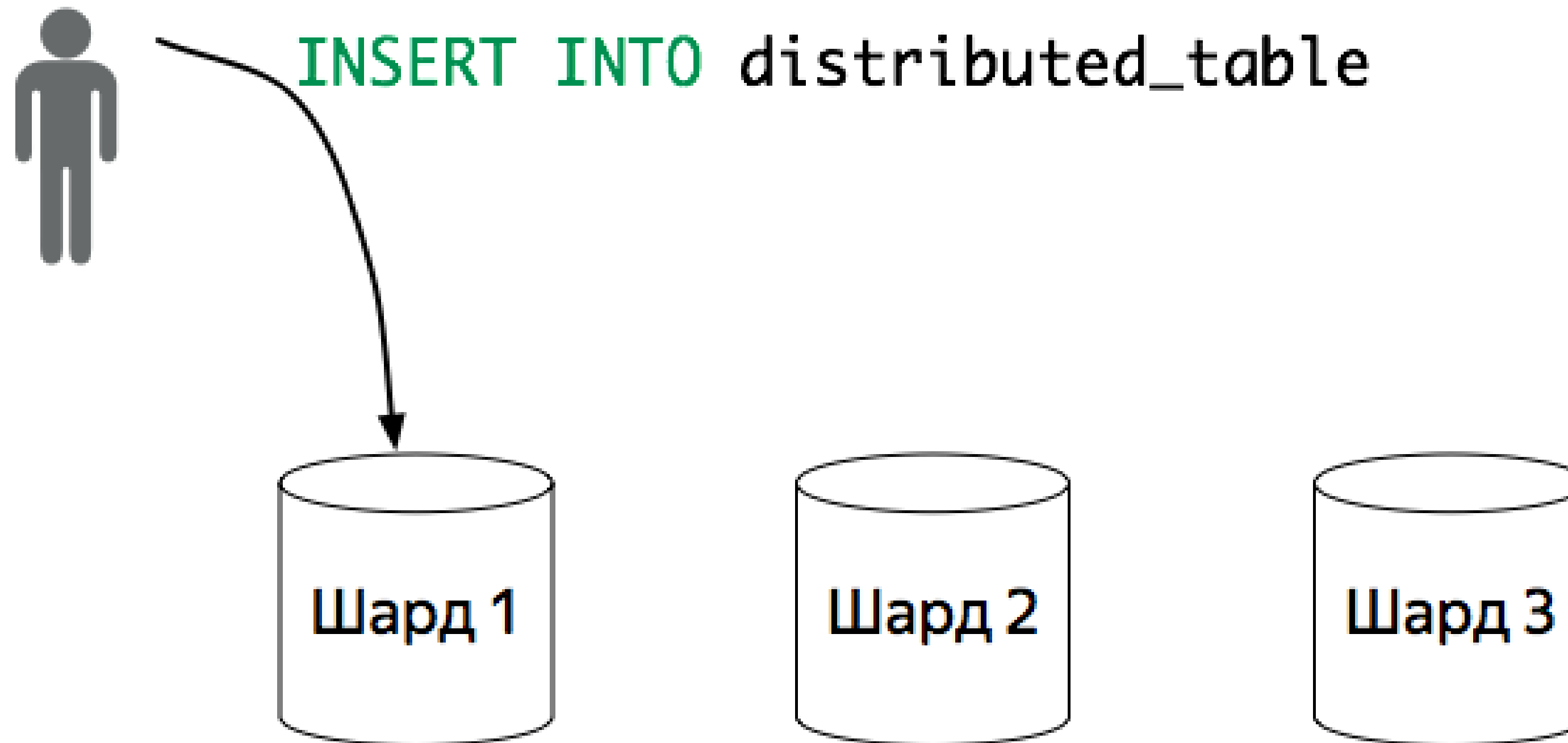
NYC taxi benchmark

CSV 227 Gb, ~1.3 млрд строк

```
SELECT passenger_count, avg(total_amount)
FROM trips GROUP BY passenger_count
```

Шардов	1	3	140
Время, с.	1,224	0,438	0,043
Ускорение		x2.8	x28.5

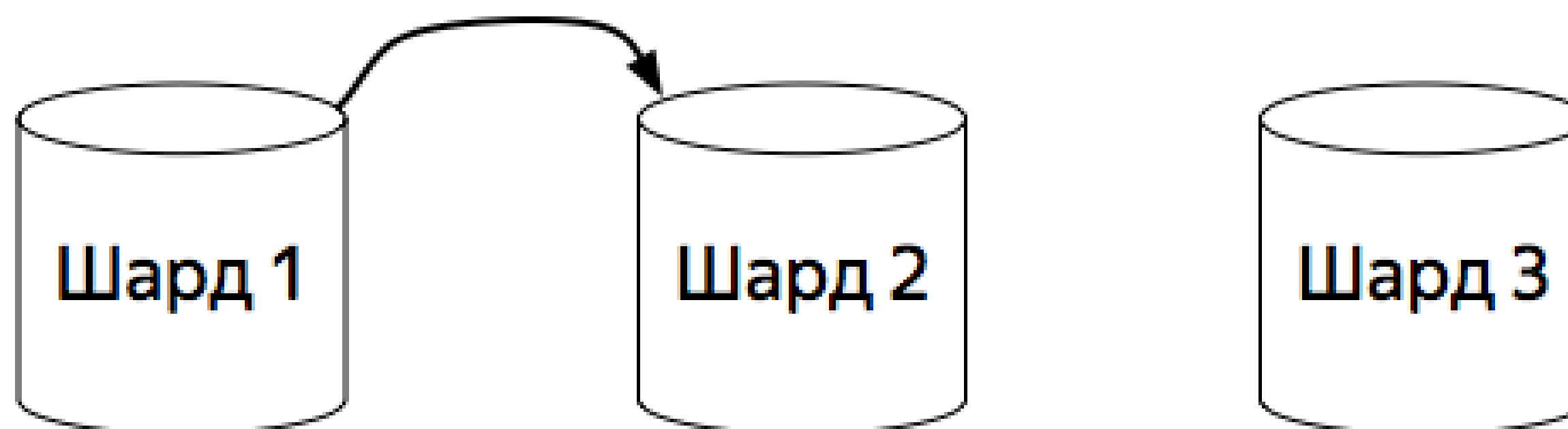
Запись в Distributed таблицу



Запись в Distributed таблицу

Асинхронно в шард номер
`sharding_key % 3`

`INSERT INTO local_table`



Когда нельзя ломаться

- › Хочется защититься от аппаратного сбоя...
- › Данные должны быть доступны на чтение и на запись...

Когда нельзя ломаться

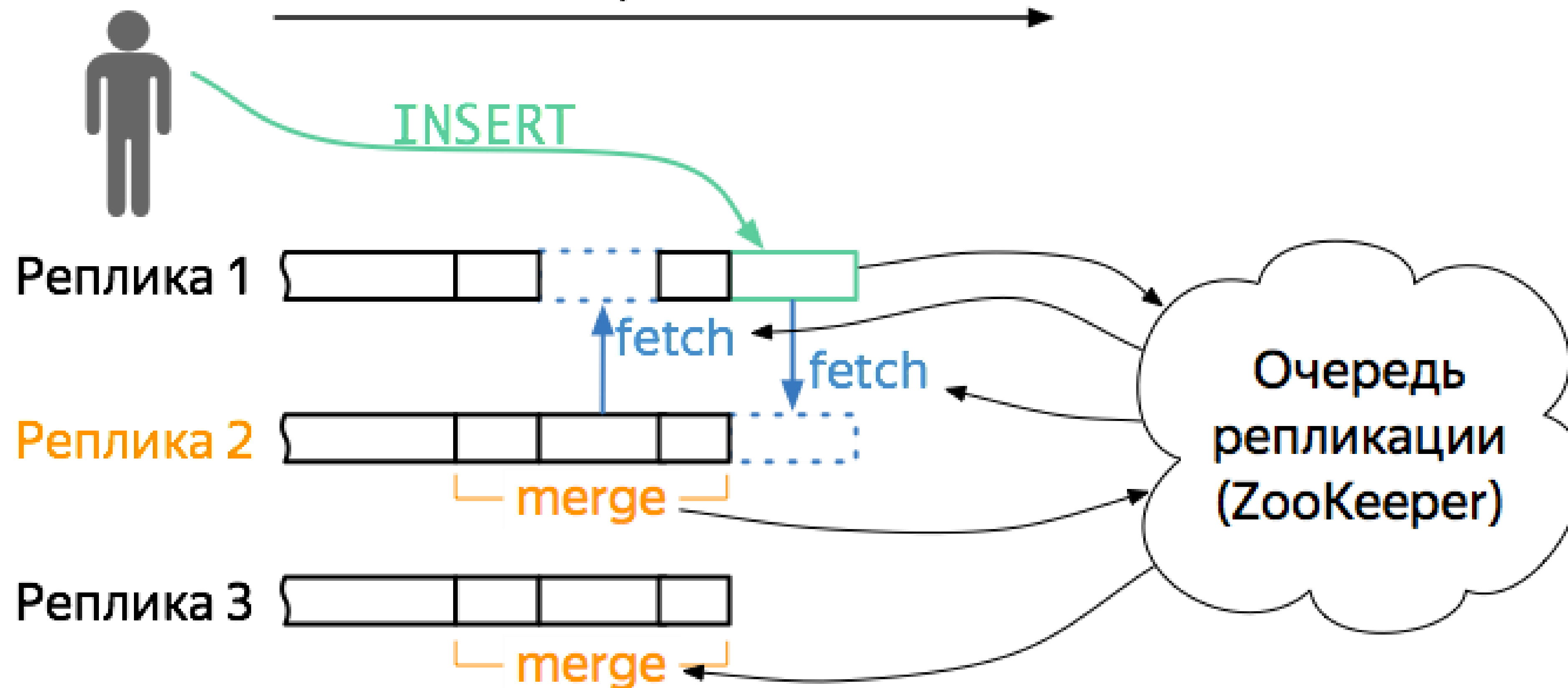
- › Хочется защититься от аппаратного сбоя...
- › Данные должны быть доступны на чтение и на запись...

ClickHouse: движок ReplicatedMergeTree

- › асинхронная мастер–мастер репликация
- › Работает на уровне таблиц

Как работает репликация

Номер вставки



Репликация с точки зрения CAP–теоремы

Что будет в случае сетевого сбоя (partition)?

› Consistency **нет!***

Как и у любой системы с асинхронной репликацией

* Но можно включить

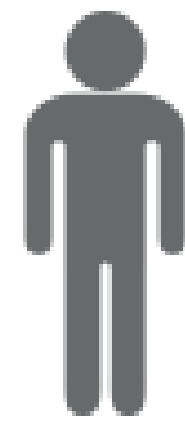
› Availability (почти) **есть!***

Можно отключать один ДЦ, если

ZK в 3-х датацентрах, а реплики минимум в 2-х.

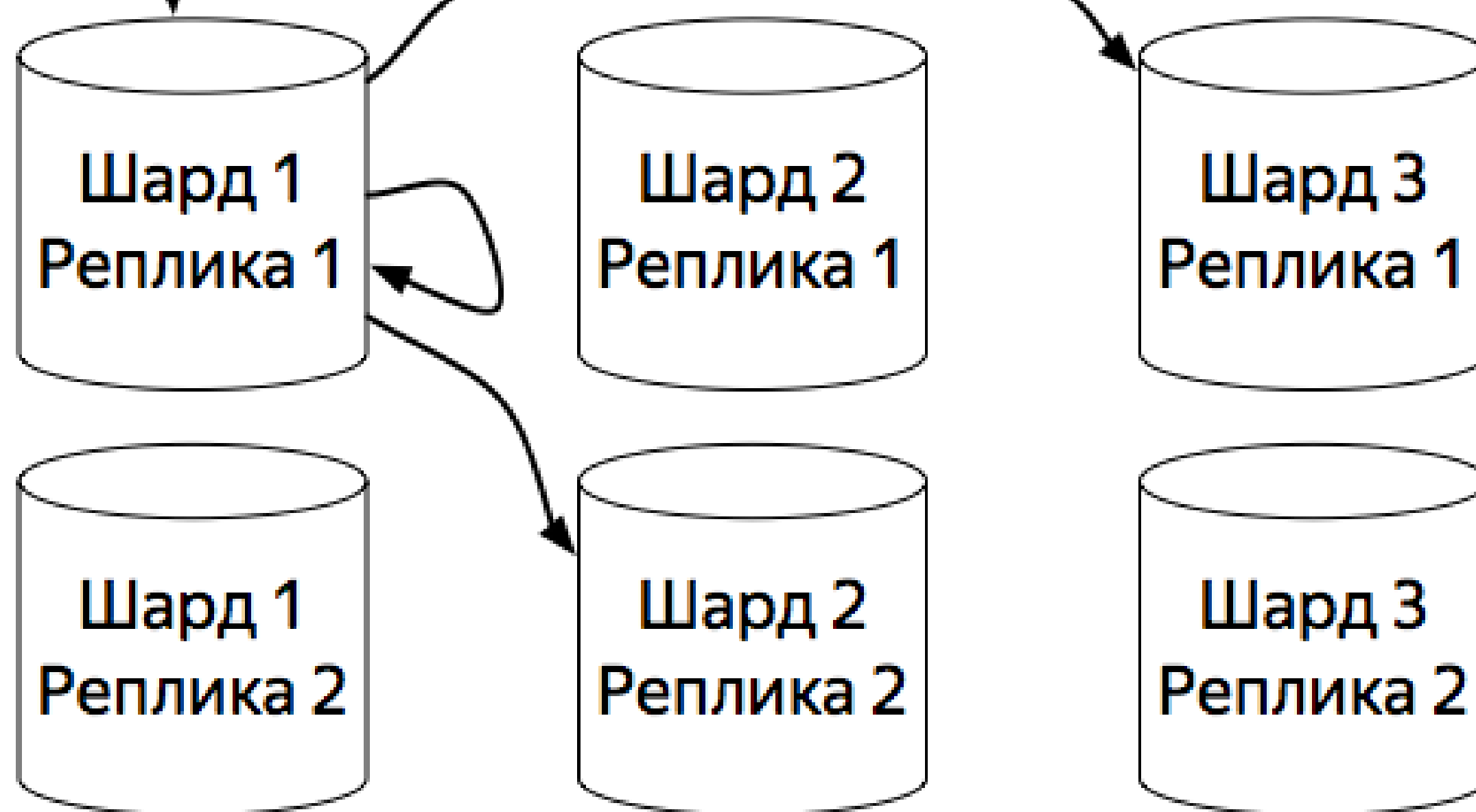
* Нельзя писать в сервер, отрезанный от кворума ZK

Всё вместе



`SELECT FROM distributed_table`

`SELECT FROM replicated_table`



Ещё раз, коротко

- › Column-oriented
- › Сверхбыстрые интерактивные запросы
- › Диалект SQL + расширения
- › Плохо подходит для OLTP, Key-Value, хранения блобов
- › Линейная масштабируемость
- › Отказоустойчивость
- › Open source!

Спасибо

Начните использовать ClickHouse сегодня!

Вопросы? Можно сюда:

- › clickhouse-feedback@yandex-team.ru
- › Telegram: https://t.me/clickhouse_ru
- › GitHub: <https://github.com/yandex/ClickHouse/>
- › Google group: <https://groups.google.com/group/clickhouse>