**1 слайд**

(смотри слайд)

**2 слайд**

Некоторые из преимуществ Cassandra:

* высокая масштабируемость и надежность без элементов, отказ которых приводит к выходу из строя всей системы;
* реализация семейства NoSQL Column;
* очень высокая пропускная способность для операций записи и хорошая пропускная способность для операций считывания;
* SQL-подобный язык запросов и поддержка поиска посредством вторичных индексов;
* настраиваемая согласованность и поддержка репликации;
* гибкая схема.

**3 слайд**

Cassandra хранит данные в соответствии с моделью данных "семейство столбцов".

Модель данных Cassandra состоит из столбцов, строк, семейств столбцов и пространства ключей. Рассмотрим каждый элемент в деталях.

Столбец ― это основной элемент модели данных Cassandra, и каждый столбец содержит имя, значение и метку времени.

Строка ― это именованная коллекция столбцов. Cassandra состоит из множества узлов хранения данных и хранит каждую строку в одном из этих узлов. В каждой строке Cassandra всегда хранит столбцы, отсортированные по имени. Благодаря этому порядку сортировки Cassandra поддерживает секционированные запросы, когда пользователь, указав строку, может получить соответствующее подмножество столбцов в заданном диапазоне имен столбцов.

Семейство столбцов ― это именованная коллекция строк. Часто говорят, что семейство столбцов соответствует таблице в реляционной модели.

Пространство ключей ― это группа из многих семейств столбцов, собранных вместе. Оно логически группирует семейства столбцов и обеспечивает изолированные области имен.

**4 слайд**

Из приведенного выше описания модели данных Cassandra следует, что в каждом семействе столбцов данные размещаются в двухмерном пространстве (2D). Чтобы извлечь данные из семейства столбцов, требуется два ключа: имя строки и имя столбца. В этом смысле реляционная модель и Cassandra схожи, хотя существует ряд важных различий.

(смотри слайд)

**5 слайд**

Сразу видно, что Cassandra ― распределенная система. Она состоит из нескольких узлов и распределяет данные между этими узлами.

Для распределения элементов данных по узлам Cassandra использует последовательное хэширование. Проще говоря, Cassandra использует хэш-алгоритм для вычисления хэш-значений ключей каждого элемента данных, хранящегося в Cassandra (имя столбца, ID строки и т.п.). Диапазон хэш-значений или все возможные хэш-значения распределяется между узлами кластера Cassandra. Затем Cassandra назначает каждому элементу данных свой узел, и этот узел отвечает за хранение и управление этим элементом данных.

Такая архитектура предоставляет следующие возможности:

* Cassandra распределяет данные между узлами прозрачно для пользователей. Любой узел может принимать любой запрос (чтение, запись или удаление), и если данные хранятся не в этом узле, перенаправляет его в нужный узел;
* пользователи могут определить необходимое количество реплик, и Cassandra прозрачно обеспечит создание реплик и управление ими;
* настраиваемая согласованность: при хранении и считывании данных пользователи могут выбирать уровень согласованности по каждой операции.
* Cassandra обеспечивает очень быструю запись, более быструю, чем чтение, со скоростью передачи данных порядка 80-360 МБ/с на узел. Это достигается с помощью двух подходов:
  + Cassandra хранит большую часть данных в оперативной памяти ответственного узла, и любые обновления выполняются в памяти, а затем записываются в файловую систему ленивым методом. Однако во избежание потери данных Cassandra регистрирует все транзакции в журнале фиксации транзакций на диске.
  + Если не требуется полная согласованность записей, Cassandra записывает данные в достаточное число узлов без разрешения конфликтов несоответствия, которые разрешаются только при первом считывании. Этот процесс называется «ремонтом при чтении».

**6 слайд**

CQL — это SQL-подобный язык. Аббревиатура от Cassandra Query Language.

**7 слайд**

Таблицы в Cassandra обязаны иметь PRIMARY KEY. Он используется для поиска ноды, в которой хранится искомая строка.

**8 слайд**

Внимание! Это две **внутренние структуры строк**, а не таблицы. Если чуть слукавить, то можно сказать, что каждая строка — это как маленькая таблица.

**9-10 слайд**

Внимание на PRIMARY KEY. Первый из параметров — company — это распределительный ключ, именно он будет использоваться для поиска ноды с этих пор. Второй ключ name — это кластерный ключ (clustering key). Он превращается в колонку.

Т.е. мы данные превращаем в название колонки. Был 'eric' обычными четырьмя байтами, а стал частью названия колонки.

**11 слайд**

Теперь наш распределительный ключ составной — (A,B). Кластерный ключ тоже составной — C, D.

**12 слайд**  
  
Внутренняя структура усложнилась. Такие данные как c, d, g, k, o, p, u, v участвуют в названии колонок наравне с E и F.

**13 слайд**

А че так сложно то?

Это самый быстрый способ записи и хранения бесконечного количества данных в распределённой БД. Cassandra как раз была разработана с упором на скорость записи/чтения.