

Онлайн-образование

Не забыть включить запись!





Меня хорошо видно && слышно?

Ставьте , если все хорошо
Напишите в чат, если есть проблемы

Правила вебинара



Активно участвуем



Задаем вопрос в чат или голосом



Off-topic обсуждаем в Slack #канал группы или #general



Вопросы вижу в чате, могу ответить не сразу



Шардирование



Тюменцев Евгений

Генеральный директор

HWdTech LLC

etyumentcev@gmail.com

Преподаватель



Тюменцев Евгений

- 9 лет руковожу компаний по разработке ПО
- в прошлом занимался разработкой многопоточных кросс-платформенных приложений на C++, серверных приложений на C#
- 20 преподаю ООП, паттерны, C++, C#, Kotlin

Цели вебинара | После занятия вы сможете

1

Использовать партиционирование

2

Использовать горизонтальный
и вертикальный шардинг

3

Разбираться с проблемами
при шардинге

The background of the slide is a high-angle, blue-tinted aerial photograph of a city skyline, likely New York City, showing numerous skyscrapers and buildings. A semi-transparent blue band with a white network pattern of dots and lines runs horizontally across the middle of the image, serving as a backdrop for the title.

01. Шардинг

Шардинг: необходимость

- Масштабирование чтения – репликация

Шардинг: необходимость

- Масштабирование чтения – репликация
- Масштабирование записи
 - Вертикальное масштабирование лидера

Шардинг: необходимость

- Масштабирование чтения – репликация
- Масштабирование записи
 - Вертикальное масштабирование лидера
 - Оптимизация
 - По данным – убираем всё лишнее из БД

Шардинг: необходимость

- Масштабирование чтения – репликация
- Масштабирование записи
 - Вертикальное масштабирование лидера
 - Оптимизация
 - По данным – убираем всё лишнее из БД
 - По настройкам – mySQLtuner, PG Tune

Шардинг: необходимость

- Масштабирование чтения – репликация
- Масштабирование записи
 - Вертикальное масштабирование лидера
 - Оптимизация
 - По данным – убираем всё лишнее из БД
 - По настройкам – `mysqltuner`, `PGTune`
- Шардинг

Шардинг: когда?

- Много данных

Шардинг: когда?

- Много данных
 - Много таблиц примерно одинакового и значительного объёма

Шардинг: когда?

- Много данных
 - Много таблиц примерно одинакового и значительного объёма
 - Таблица с большим количеством записей (медленная выборка даже по индексу)

Шардинг: когда?

- Много данных
 - Много таблиц примерно одинакового и значительного объёма
 - Таблица с большим количеством записей (медленная выборка даже по индексу)
 - Таблица с большим размером строки

Шардинг: когда?

- Много данных
 - Много таблиц примерно одинакового и значительного объёма
 - Таблица с большим количеством записей (медленная выборка даже по индексу)
 - Таблица с большим размером строки
 - Здесь имеет смысл сначала рассмотреть денормализацию

Шардинг: когда?

- Много данных
 - Много таблиц примерно одинакового и значительного объёма
 - Таблица с большим количеством записей (медленная выборка даже по индексу)
 - Таблица с большим размером строки
 - Здесь имеет смысл сначала рассмотреть денормализацию
- Микросервисная архитектура

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер
- Проблемы:
 - Как определить, на какой сервер писать данные?

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер
- Проблемы:
 - Как определить, на какой сервер писать данные?
 - Как определить, с какого сервера читать данные?

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер
- Проблемы:
 - Как определить, на какой сервер писать данные?
 - Как определить, с какого сервера читать данные?
- Виды шардинга:
 - Вертикальный

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер
- Проблемы:
 - Как определить, на какой сервер писать данные?
 - Как определить, с какого сервера читать данные?
- Виды шардинга:
 - Вертикальный
 - Горизонтальный

Шардинг: что такое?

- Шардинг – вынос части данных на другой сервер
- Проблемы:
 - Как определить, на какой сервер писать данные?
 - Как определить, с какого сервера читать данные?
- Виды шардинга:
 - Вертикальный
 - Горизонтальный
 - Партиционирование

The background of the slide is a high-angle, aerial photograph of a city skyline, likely New York City, showing numerous skyscrapers and buildings. The image is overlaid with a semi-transparent blue layer. On this blue layer, there is a network of thin, light blue lines connecting various points, creating a geometric, web-like pattern. The text '02. Партиционирование' is centered in the middle of the slide in a large, white, sans-serif font.

02. Партиционирование

Партиционирование

- Таблица разбивается на части и запрос выполняется параллельно над каждой частью таблицы

Партиционирование

- Таблица разбивается на части и запрос выполняется параллельно над каждой частью таблицы
- Что даёт:
 - Можем поместить больше данных в одну таблицу

Партиционирование

- Таблица разбивается на части и запрос выполняется параллельно над каждой частью таблицы
- Что даёт:
 - Можем поместить больше данных в одну таблицу
 - Удаление старых данных (партиционирование по времени)

Партиционирование

- Таблица разбивается на части и запрос выполняется параллельно над каждой частью таблицы
- Что даёт:
 - Можем поместить больше данных в одну таблицу
 - Удаление старых данных (партиционирование по времени)
 - Оптимизация запросов (можно отбросить часть партиций)

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях
 - Вертикальное – колонки таблицы в разных партициях

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях
 - Вертикальное – колонки таблицы в разных партициях
- Способы разделения данных:
 - По диапазону значений

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях
 - Вертикальное – колонки таблицы в разных партициях
- Способы разделения данных:
 - По диапазону значений
 - По точному списку значений

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях
 - Вертикальное – колонки таблицы в разных партициях
- Способы разделения данных:
 - По диапазону значений
 - По точному списку значений
 - По хэшу

Виды партиционирования

- Виды партиционирования
 - Горизонтальное – строки таблицы в разных партициях
 - Вертикальное – колонки таблицы в разных партициях
- Способы разделения данных:
 - По диапазону значений
 - По точному списку значений
 - По хэшу
 - По ключу

Вертикальное партиционирование

- Как реализовать руками:
 - Денормализуем таблицу, вынося нужные колонки в отдельную таблицу

Вертикальное партиционирование

- Как реализовать руками:
 - Денормализуем таблицу, вынося нужные колонки в отдельную таблицу
 - Связь по primary key

Вертикальное партиционирование

- Как реализовать руками:
 - Денормализуем таблицу, вынося нужные колонки в отдельную таблицу
 - Связь по primary key
 - Можно сделать VIEW, чтобы не переделывать запросы

Вертикальное партиционирование

- Как реализовать руками:
 - Денормализуем таблицу, вынося нужные колонки в отдельную таблицу
 - Связь по primary key
 - Можно сделать VIEW, чтобы не переделывать запросы
- **Важно!** Если данные из новой таблицы нужны в каждом запросе, получим замедление вместо ускорения

Примеры в MySQL: по диапазону

```
CREATE TABLE orders_range (  
    id INT NOT NULL AUTO_INCREMENT,  
    customer_surname VARCHAR(30),  
    store_id INT,  
    salesperson_id INT,  
    order_date DATE,  
    note VARCHAR(500)  
) ENGINE = MYISAM  
PARTITION BY RANGE( YEAR(order_date) ) (  
    PARTITION p_archive VALUES LESS THAN(2018),  
    PARTITION p_2018 VALUES LESS THAN(2019),  
    PARTITION p_2019 VALUES LESS THAN(MAXVALUE)  
) ;
```

Примеры в PostgreSQL: по списку

```
CREATE TABLE orders_range (  
    id SERIAL,  
    customer_surname VARCHAR(30),  
    store_id INT,  
    salesperson_id INT,  
    order_date DATE,  
)  
PARTITION BY LIST(store_id);
```

```
CREATE TABLE orders_range_moscow PARTITION OF orders_range  
    FOR VALUES IN (1,2,3,10,15);
```

```
CREATE TABLE orders_range_region PARTITION OF orders_range  
    FOR VALUES IN (4,5,6,7,8,9,11,12,13,14);
```




03. Вертикальный шардинг

Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования

Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования
- Приёмы реализации:
 - Отдельное соединение на каждую БД

Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования
- Приёмы реализации:
 - Отдельное соединение на каждую БД
 - Использовать lazy loading

Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования
- Приёмы реализации:
 - Отдельное соединение на каждую БД
 - Использовать lazy loading
 - На один шард выделяем таблицы, которые взаимосвязаны

Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования
- Приёмы реализации:
 - Отдельное соединение на каждую БД
 - Использовать lazy loading
 - На один шард выделяем таблицы, которые взаимосвязаны
 - Если нет такой возможности, то
 - Объединение в коде

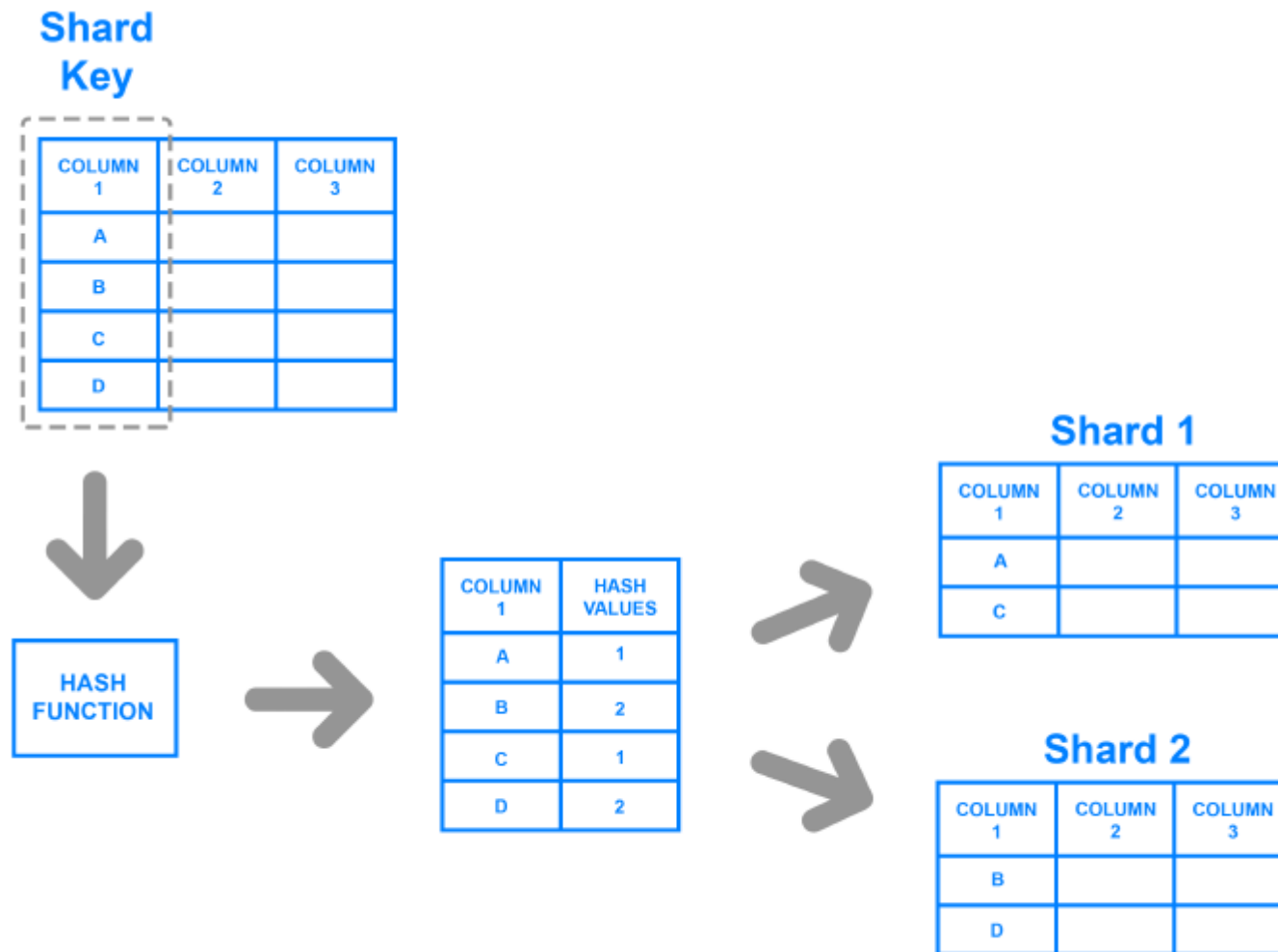
Вертикальный шардинг

- Вертикальный шардинг – развитие идеи вертикального партиционирования
- Приёмы реализации:
 - Отдельное соединение на каждую БД
 - Использовать lazy loading
 - На один шард выделяем таблицы, которые взаимосвязаны
 - Если нет такой возможности, то
 - Объединение в коде
 - Инвертированный индекс

The background of the slide is a high-angle, aerial photograph of a city skyline, likely New York City, featuring numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. A network of thin, light-blue lines connects various points across the blue area, creating a web-like pattern that suggests connectivity or data flow. The text '04. Горизонтальный шардинг' is centered in white, bold font.

04. Горизонтальный шардинг

Горизонтальный шардинг: ключ



Горизонтальный шардинг: ключ

- Что нужно учитывать при выборе ключа:
 - Равномерность распределения по шардам

Горизонтальный шардинг: ключ

- Что нужно учитывать при выборе ключа:
 - Равномерность распределения по шардам
 - Количество данных

Горизонтальный шардинг: ключ

- Что нужно учитывать при выборе ключа:
 - Равномерность распределения по шардам
 - Количество данных
 - Хэш для длинных строк

Горизонтальный шардинг: ключ

- Что нужно учитывать при выборе ключа:
 - Равномерность распределения по шардам
 - Количество данных
 - Хэш для длинных строк
 - Устаревание записей

Горизонтальный шардинг: ключ

- Что нужно учитывать при выборе ключа:
 - Равномерность распределения по шардам
 - Количество данных
 - Хэш для длинных строк
 - Устаревание записей
 - Вычислимость ключа до выполнения insert

Горизонтальный шардинг: ключ

- Функция должна иметь вид $F(x,n) = y$

Горизонтальный шардинг: ключ

- Функция должна иметь вид $F(x,n) = y$
 - x – значение ключа (хэш)

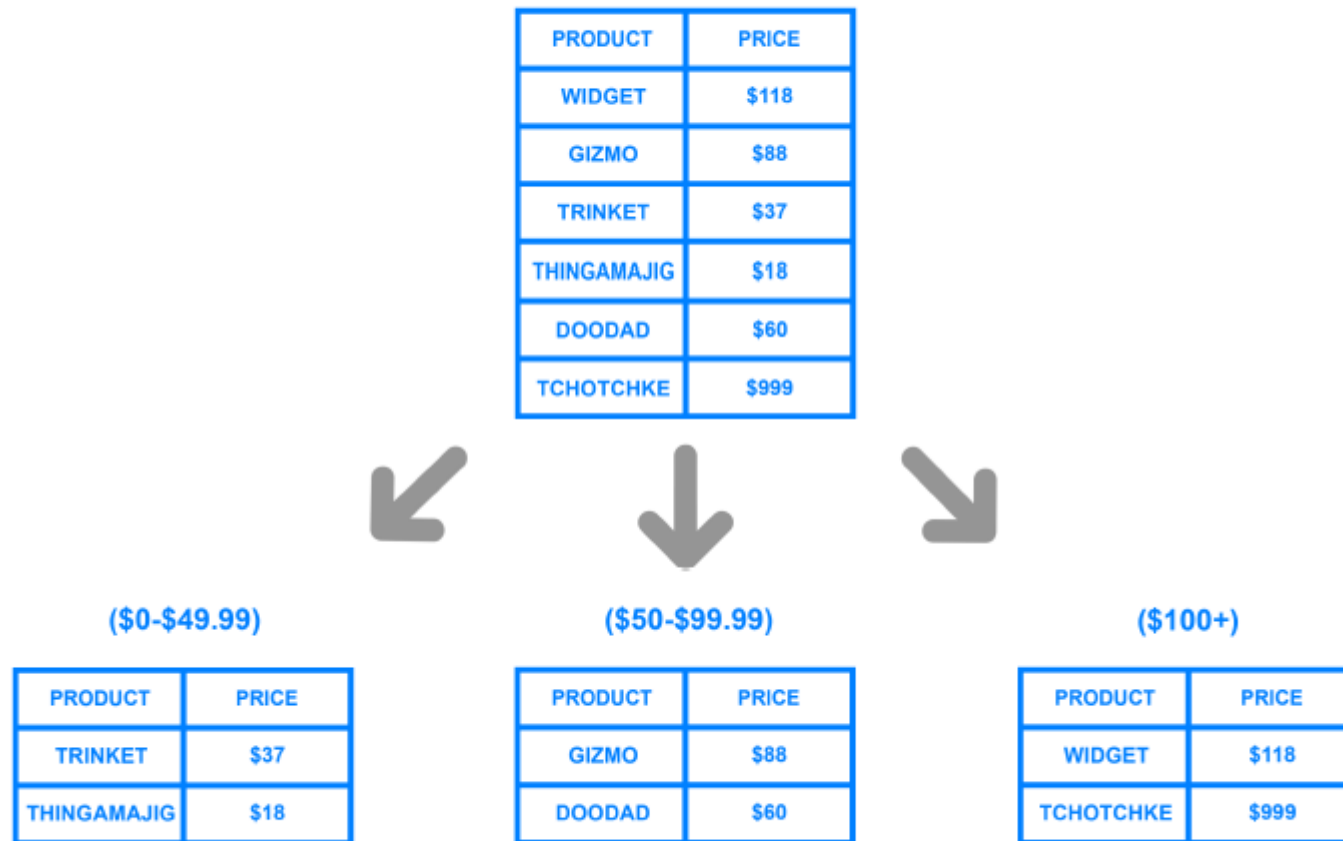
Горизонтальный шардинг: ключ

- Функция должна иметь вид $F(x,n) = y$
 - x – значение ключа (хэш)
 - n – количество серверов

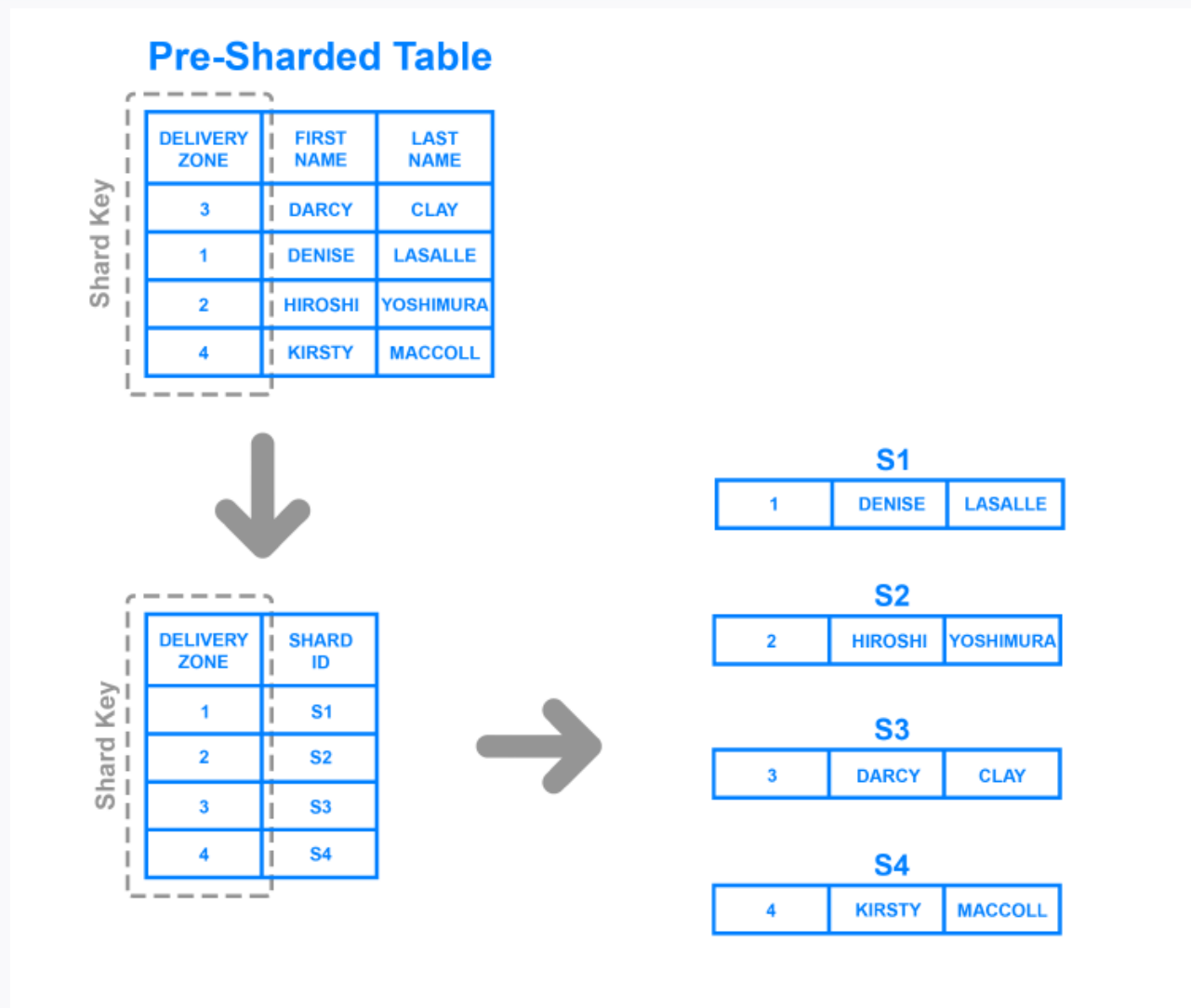
Горизонтальный шардинг: ключ

- Функция должна иметь вид $F(x,n) = y$
 - x – значение ключа (хэш)
 - n – количество серверов
 - y – номер сервера

Горизонтальный шардинг: диапазон



Горизонтальный шардинг: реестр



Маршрутизация: умный клиент

- Преимущества:
 - Легко реализуем

Маршрутизация: умный клиент

- Преимущества:
 - Легко реализуем
 - Нет лишних точек отказа

Маршрутизация: умный клиент

- Преимущества:
 - Легко реализуем
 - Нет лишних точек отказа
 - Нет лишних промежуточных узлов

Маршрутизация: умный клиент

- Преимущества:
 - Легко реализуем
 - Нет лишних точек отказа
 - Нет лишних промежуточных узлов
- Недостатки:
 - Обновление требует обновлять все зависимые сервисы

Маршрутизация: умный клиент

- Преимущества:
 - Легко реализуем
 - Нет лишних точек отказа
 - Нет лишних промежуточных узлов
- Недостатки:
 - Обновление требует обновлять все зависимые сервисы
 - Решардинг затруднён

Маршрутизация: прокси

- Преимущества:
 - Для приложения шардинг прозрачен

Маршрутизация: прокси

- Преимущества:
 - Для приложения шардинг прозрачен
 - Код приложения остаётся неизменным

Маршрутизация: прокси

- Преимущества:
 - Для приложения шардинг прозрачен
 - Код приложения остаётся неизменным
- Недостатки:
 - Промежуточные узлы

Маршрутизация: прокси

- Преимущества:
 - Для приложения шардинг прозрачен
 - Код приложения остаётся неизменным
- Недостатки:
 - Промежуточные узлы
 - Лишняя точка отказа

Маршрутизация: прокси

- Преимущества:
 - Для приложения шардинг прозрачен
 - Код приложения остаётся неизменным
- Недостатки:
 - Промежуточные узлы
 - Лишняя точка отказа
 - Если прокси несколько, то им нужно синхронизироваться

Маршрутизация: координатор

- Преимущества:
 - Прямое соединение к БД со стороны приложения

Маршрутизация: координатор

- Преимущества:
 - Прямое соединение к БД со стороны приложения
 - Код приложения меняется незначительно

Маршрутизация: координатор

- Преимущества:
 - Прямое соединение к БД со стороны приложения
 - Код приложения меняется незначительно
 - Экономия общей нагрузки на сеть по сравнению с прокси

Маршрутизация: координатор

- Преимущества:
 - Прямое соединение к БД со стороны приложения
 - Код приложения меняется незначительно
 - Экономия общей нагрузки на сеть по сравнению с прокси
- Недостатки:
 - Лишний запрос для получение точки входа в БД

Маршрутизация: координатор

- Преимущества:
 - Прямое соединение к БД со стороны приложения
 - Код приложения меняется незначительно
 - Экономия общей нагрузки на сеть по сравнению с прокси
- Недостатки:
 - Лишний запрос для получение точки входа в БД
 - SPOF* (можно масштабировать)

The background of the slide is a high-angle, blue-tinted aerial photograph of a dense urban skyline, likely New York City. Overlaid on this image is a semi-transparent blue band across the middle, which contains a white network diagram of interconnected nodes and lines. The title '04. Примеры' is centered within this band in a large, white, sans-serif font.

04. Примеры

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 1. Вертикальное партиционирование:

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 1. Вертикальное партиционирование: отделить картинки и видео от товаров
 2. Вертикальный шардинг:

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 1. Вертикальное партиционирование: отделить картинки и видео от товаров
 2. Вертикальный шардинг: отделяем таблицы заказов и контрагентов

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 3. Горизонтальное партиционирование:

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 3. Горизонтальное партиционирование: список заказов по годам
 4. Горизонтальный шардинг:

Шардинг: примеры

- **Задача:** хотим масштабировать базу данных интернет-магазина, содержащую три большие таблицы:
 - Каталог товаров с картинками и видео
 - Список контрагентов
 - Список заказов (+ таблица связей товаров и заказов)
- **Решения:**
 3. Горизонтальное партиционирование: список заказов по годам
 4. Горизонтальный шардинг:
 - Заказы – по хешу номера заказа
 - Товары – по хешу кода товара

Шардинг: примеры

- **Задача:** хотим масштабировать систему проведения соревнований, есть три таблицы:
 - Участники соревнований
 - Результаты соревнований
 - Время участников на контрольных точках
- **Решения:**
 1. Горизонтальное партиционирование:

Шардинг: примеры

- **Задача:** хотим масштабировать систему проведения соревнований, есть три таблицы:
 - Участники соревнований
 - Результаты соревнований
 - Время участников на контрольных точках
- **Решения:**
 1. Горизонтальное партиционирование: время участников по дате соревнования
 2. Горизонтальный шардинг:

Шардинг: примеры

- **Задача:** хотим масштабировать систему проведения соревнований, есть три таблицы:
 - Участники соревнований
 - Результаты соревнований
 - Время участников на контрольных точках
- **Решения:**
 1. Горизонтальное партиционирование: время участников по дате соревнования
 2. Горизонтальный шардинг: участники соревнований и их время – по хешу id участника

Шардинг: примеры

- **Задача:** хотим масштабировать систему проведения соревнований, есть три таблицы:
 - Участники соревнований
 - Результаты соревнований
 - Время участников на контрольных точках
- **Решения:**
 3. Вертикальный шардинг:

Шардинг: примеры

- **Задача:** хотим масштабировать систему проведения соревнований, есть три таблицы:
 - Участники соревнований
 - Результаты соревнований
 - Время участников на контрольных точках
- **Решения:**
 3. Вертикальный шардинг: отделяем время участников и дублируем на шард таблицу соревнований и часть таблицы участников

The background of the slide is a high-angle, blue-tinted aerial photograph of a city skyline, likely New York City, showing numerous skyscrapers and buildings. A semi-transparent blue band with a white network pattern of dots and lines runs horizontally across the middle of the image, serving as a backdrop for the title.

05. Проблемы при шардинге

JOIN

- Если JOIN-операция делается не по ключу шардирования, то придётся перебирать все шарды

JOIN

- Если JOIN-операция делается не по ключу шардирования, то придётся перебирать все шарды
- Способы решения:
 - Шардируем не только «основные», но и связанные данные

JOIN

- Если JOIN-операция делается не по ключу шардирования, то придётся перебирать все шарды
- Способы решения:
 - Шардируем не только «основные», но и связанные данные
 - Дублируем связанные данные на все шарды

JOIN

- Если JOIN-операция делается не по ключу шардирования, то придётся перебирать все шарды
- Способы решения:
 - Шардируем не только «основные», но и связанные данные
 - Дублируем связанные данные на все шарды
 - Map-Reduce

Перебалансировка

- Когда нужна?

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные
- Алгоритм:
 1. Строим список бакетов, которые требуют переноса в формате (бакет, serverFrom, serverTo)

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные
- Алгоритм:
 1. Строим список бакетов, которые требуют переноса в формате (бакет, serverFrom, serverTo)
 2. Распространяем список на клиентов

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные
- Алгоритм:
 1. Строим список бакетов, которые требуют переноса в формате (бакет, serverFrom, serverTo)
 2. Распространяем список на клиентов
 3. Переносим данные

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные
- Алгоритм:
 1. Строим список бакетов, которые требуют переноса в формате (бакет, serverFrom, serverTo)
 2. Распространяем список на клиентов
 3. Переносим данные
 4. Меняем словарь по списку

Перебалансировка

- Когда нужна?
 - Понимаем, что есть перегруженные шарды, а есть недогруженные
- Алгоритм:
 1. Строим список бакетов, которые требуют переноса в формате (бакет, serverFrom, serverTo)
 2. Распространяем список на клиентов
 3. Переносим данные
 4. Меняем словарь по списку
 5. Удаляем список, оповещаем клиентов

Перебалансировка: перенос данных

- Read-only mode
 - Запрет на запись на serverFrom в переносимые бакеты на всё время переноса

Перебалансировка: перенос данных

- Если данные неизменяемые

Перебалансировка: перенос данных

- Если данные неизменяемые
 - Запись на serverTo, чтение с serverTo и serverFrom

Перебалансировка: перенос данных

- Если данные неизменяемые
 - Запись на serverTo, чтение с serverTo и serverFrom
 - Преимущества:
 - Синхронизация мобильных клиентов

Перебалансировка: перенос данных

- Если данные неизменяемые
 - Запись на serverTo, чтение с serverTo и serverFrom
 - Преимущества:
 - Синхронизация мобильных клиентов
 - Версионность данных

Перебалансировка: перенос данных

- Если данные неизменяемые
 - Запись на serverTo, чтение с serverTo и serverFrom
 - Преимущества:
 - Синхронизация мобильных клиентов
 - Версионность данных
 - Предсказуемая производительность

Перебалансировка: перенос данных

- Логическая репликация между serverFrom и serverTo
 - Ждём полной синхронизации, работая только с serverFrom, потом переключаемся на serverTo

Решардинг

- Полное изменение схемы шардинга

Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)

Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)
- Изменение числа бакетов
 - Номер бакета по остатку от деления

Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)
- Изменение числа бакетов
 - Номер бакета по остатку от деления
 1. Удваиваем количество бакетов

Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)
- Изменение числа бакетов
 - Номер бакета по остатку от деления
 1. Удваиваем количество бакетов
 2. Меняем словарь так, чтобы новые бакеты попадали на старые сервера

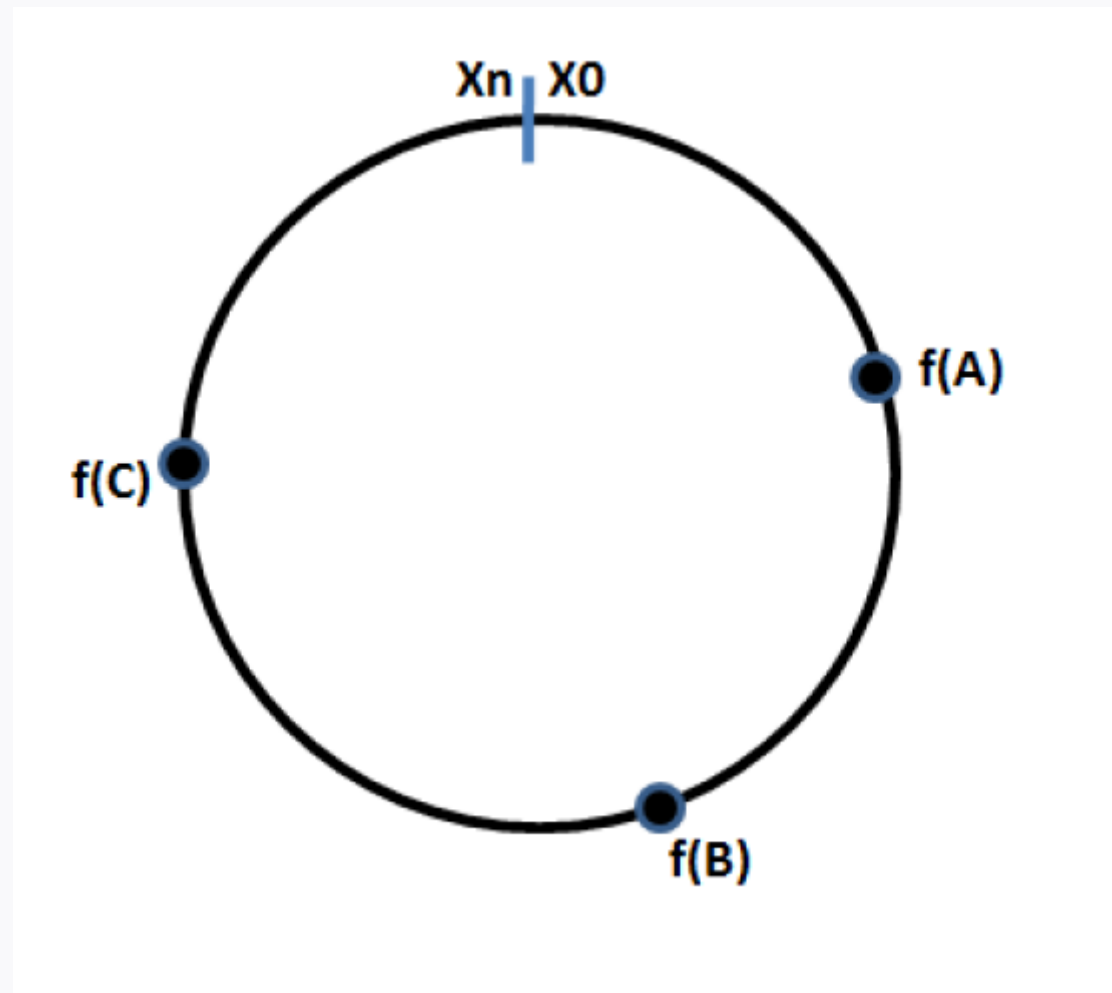
Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)
- Изменение числа бакетов
 - Номер бакета по остатку от деления
 1. Удваиваем количество бакетов
 2. Меняем словарь так, чтобы новые бакеты попадали на старые сервера
 - Более сложный алгоритм
 - Можем добавлять произвольное количество бакетов

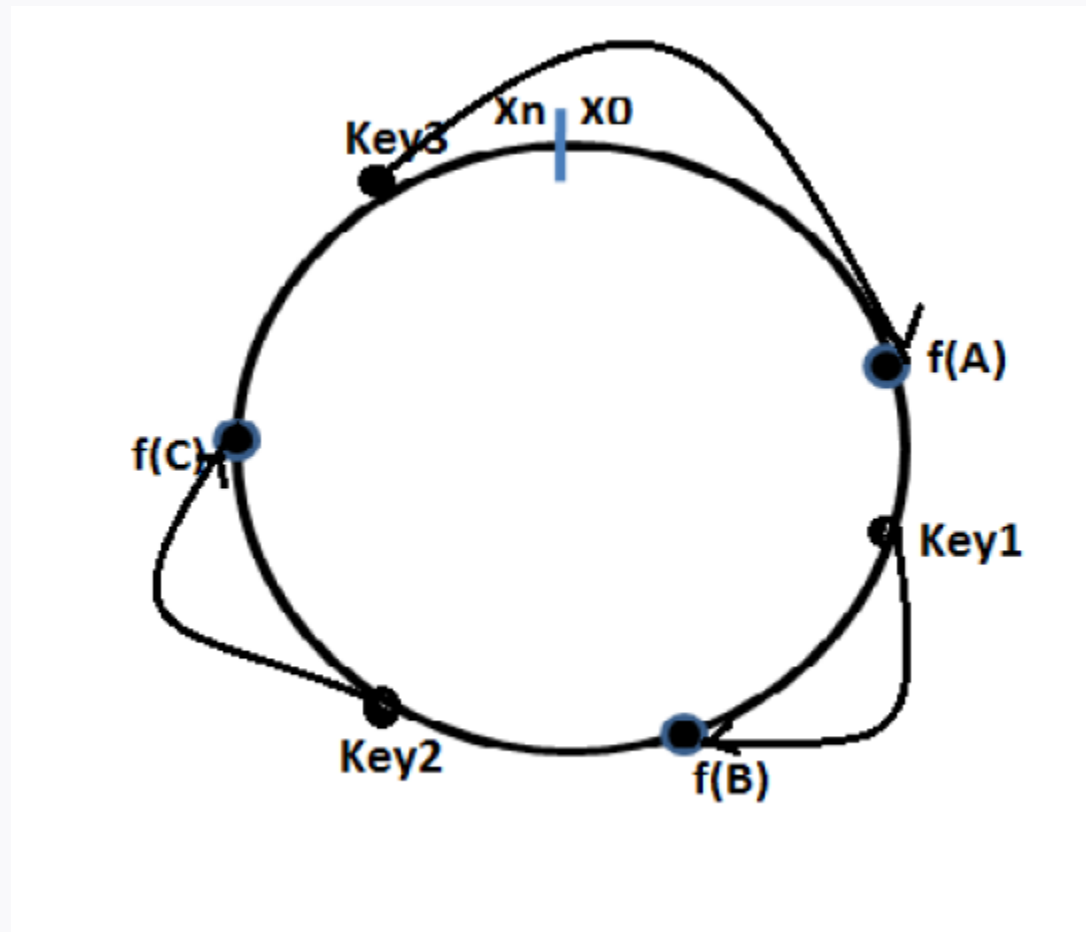
Решардинг

- Полное изменение схемы шардинга – лучше **никогда** так не делать, единственное исключение – шардинг реализовать полностью на стороне приложения (в коде)
- Изменение числа бакетов
 - Номер бакета по остатку от деления
 1. Удваиваем количество бакетов
 2. Меняем словарь так, чтобы новые бакеты попадали на старые сервера
 - Более сложный алгоритм
 - Можем добавлять произвольное количество бакетов
 - Реализуется **только** на стороне приложения

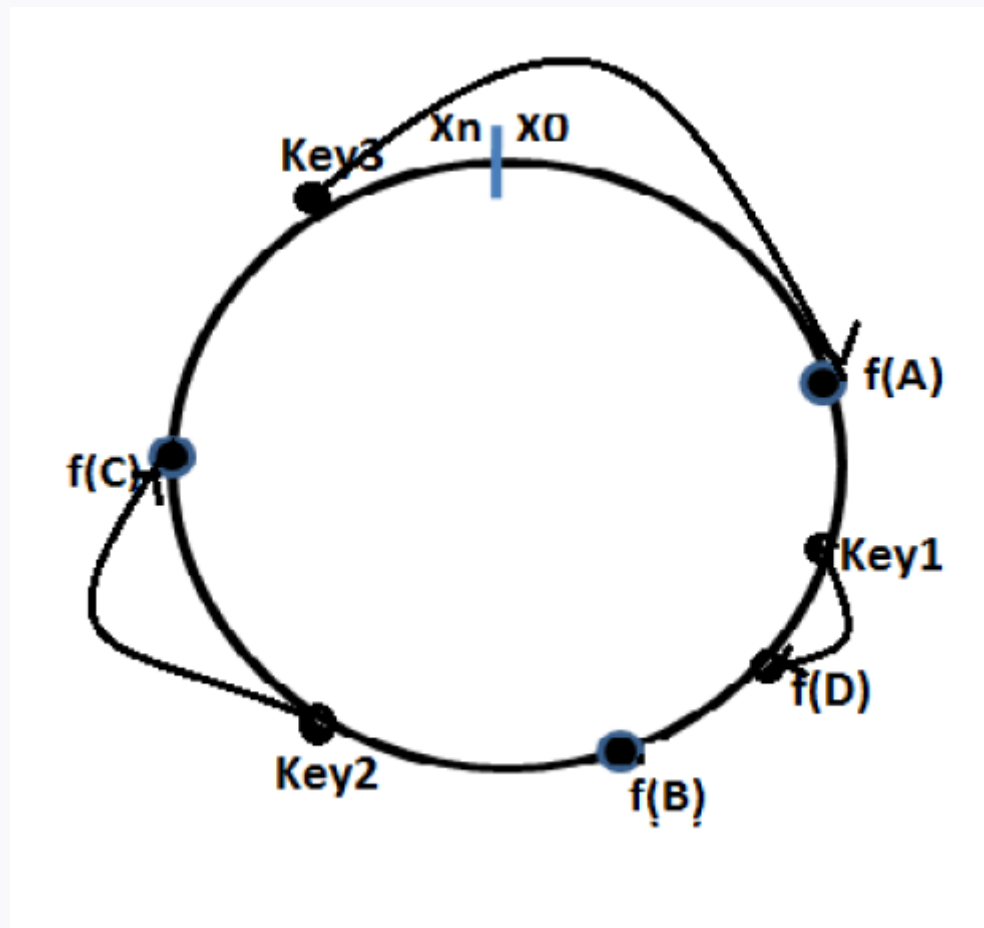
Согласованное хэширование



Согласованное хэширование



Согласованное хэширование



Согласованное хэширование

- Одна хэш-функция для нод и данных

Согласованное хэширование

- Одна хэш-функция для нод и данных
- Выбирается ближайший по часовой стрелке узел

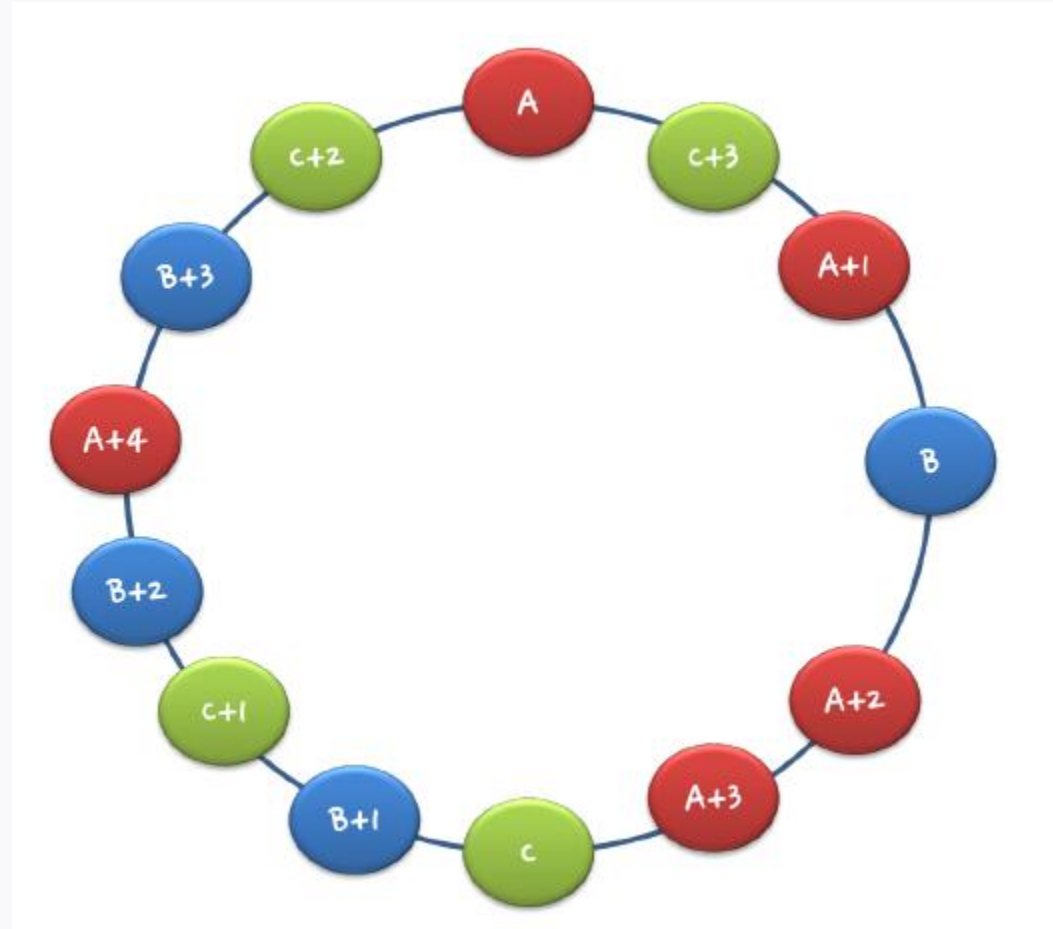
Согласованное хэширование

- Одна хэш-функция для нод и данных
- Выбирается ближайший по часовой стрелке узел
- При добавлении/удалении узла затрагивается только часть данных (формально, не больше K/N , где K – количество ключей, N – количество серверов)

Согласованное хэширование

- Одна хэш-функция для нод и данных
- Выбирается ближайший по часовой стрелке узел
- При добавлении/удалении узла затрагивается только часть данных (формально, не больше K/N , где K – количество ключей, N – количество серверов)
- **Равномерность распределения не гарантируется**

Согласованное хэширование с бакетами



Согласованное хэширование с бакетами

- Более равномерное распределение данных

Согласованное хэширование с бакетами

- Более равномерное распределение данных
- Количество физических серверов остаётся прежним

Производительность

- Запросы по ключу шардирования вероятно ускорятся

Производительность

- Запросы по ключу шардирования вероятно ускорятся
- Запросы не по ключу обойдут все шарды

Производительность

- Запросы по ключу шардирования вероятно ускорятся
- Запросы не по ключу обойдут все шарды
- Запросы по диапазону ключей **могут** обойти все шарды

Производительность

- Запросы по ключу шардирования вероятно ускорятся
- Запросы не по ключу обойдут все шарды
- Запросы по диапазону ключей **могут** обойти все шарды
- Aggregate/join – обсудили ранее

Закон Амдала

- **Было:** $\text{Total} = \text{Serial} + \text{Parallel}$
- **Стало:** $\text{Total} = \text{Serial} + \text{Parallel} / N + X_{\text{serial}}$

Закон Амдала

- **Было:** $\text{Total} = \text{Serial} + \text{Parallel}$
- **Стало:** $\text{Total} = \text{Serial} + \text{Parallel} / N + X_{\text{serial}}$

Two independent parts **A** **B**

Original process



Make **B** 5x faster



Make **A** 2x faster



Базы данных и автошардинг

- MySQL не умеет (есть Vitesse)
- PostgreSQL умеет через расширение FDW
- Cassandra, MongoDB умеют

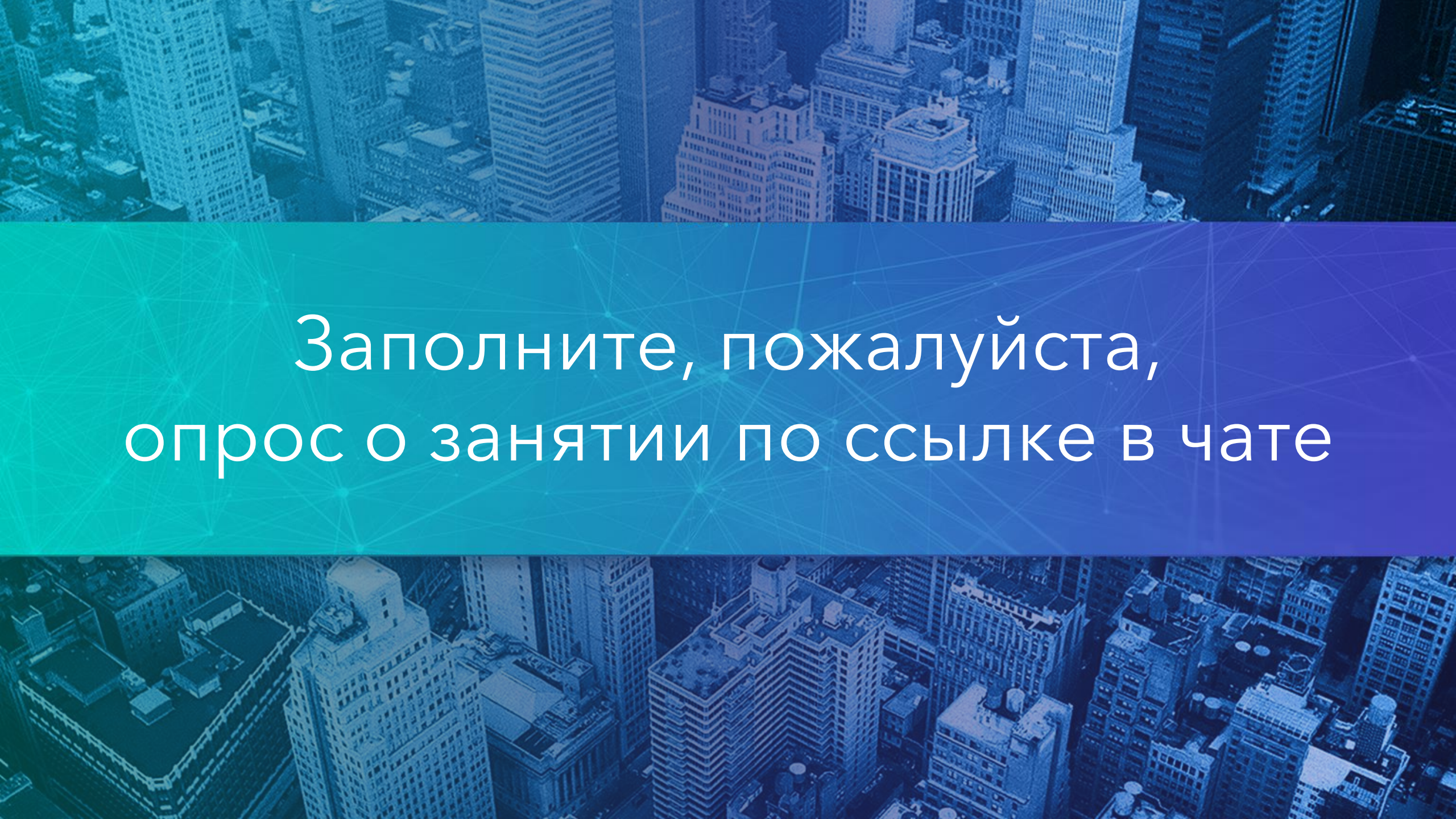
PostgreSQL FDW

```
CREATE TABLE temperature (  
    id BIGSERIAL PRIMARY KEY NOT NULL,  
    city_id INT NOT NULL,  
    timestamp TIMESTAMP NOT NULL,  
    temp DECIMAL(5,2) NOT NULL  
);
```

```
CREATE TABLE temperature_201904 (  
    id BIGSERIAL NOT NULL,  
    city_id INT NOT NULL,  
    timestamp TIMESTAMP NOT NULL,  
    temp DECIMAL(5,2) NOT NULL  
);
```

PostgreSQL FDW

```
CREATE EXTENSION postgres_fdw;  
GRANT USAGE ON FOREIGN DATA WRAPPER postgres_fdw to app_user;  
CREATE SERVER shard02 FOREIGN DATA WRAPPER postgres_fdw  
    OPTIONS (dbname 'postgres', host 'shard02', port '5432');  
CREATE USER MAPPING for app_user SERVER shard02  
    OPTIONS (user 'fdw_user', password 'secret');  
  
CREATE FOREIGN TABLE temperature_201904 PARTITION OF temperature  
    FOR VALUES FROM ('2019-04-01') TO ('2019-05-01')  
    SERVER remoteserver01;
```


The background of the image is an aerial photograph of a city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer that features a white geometric network pattern of lines and dots. The text is centered within this blue layer.

Заполните, пожалуйста,
опрос о занятии по ссылке в чате