ОНЛАЙН-ОБРАЗОВАНИЕ

OTUS

Кеширование.

Иван Ремень

Как меня слышно и видно?

> Напишите в чат

+ если все хорошо

- если есть проблемы со звуком или с видео

!проверить запись!

План занятия

- Рассмотрим клиентское кеширования.
- Рассмотрим серверное кеширование
- Изучим проблемы кеширования.
- Поговоим о реальных кейсах.
- Ответы на вопросы.

Что такое кеширование

Кэш или кеш(англ. cache, от фр. cacher — «прятать»; произносится [kæʃ] — «кэш») — промежуточный буфер с быстрым доступом к нему, содержащий информацию, которая может быть запрошена с наибольшей вероятностью. Доступ к данным в кэше осуществляется быстрее, чем выборка исходных данных из более медленной памяти или удалённого источника, однако её объём существенно ограничен по сравнению с хранилищем исходных данных.

Принцип локальности

В локальные моменты времени используется лишь небольшое подмножество данных.

- Для чатов - последние сообщения
- Для новостей - последние новости
- Для сервиса такси - активные заказы.

Алгоритмы вытеснения

- Алгоритм Белади (идеальный алгоритм)
- Least recently used (LRU)
- Most Recently Used (MRU)
- Псевдо-LRU
- Least-Frequently Used
- Adaptive Replacement Cache

Правило №1

Вы должны держать нагрузку без кеша. Задача кеша - ускорить ответ, а не держать нагрузку.

Проводите учения. Убедитесь, что вы работаете без кеша.

Виды кеширования

- Кеширование в браузере
- Кеширование страниц (или блоков)
- Кеширование данных базы

Кеширование в браузере

Кешируем только GET.

- ETAG
- If-Modified-Since
- Cache-Control
- LocalStorage

Ожидается, что GET - идемпотентен.

Кеширование в браузере

Кеширование в браузере - инвалидация

Самый простой вариант - указывать версию в гет-параметрах. Еще более правильный вариант - делать название файла md5-суммой от содержимого.

Кеширование на nginx

Можно кешировать страницы на nginx.

```
proxy_cache_path /data/nginx/cache keys_zone=cache_zone:10m;

map $request_method $purge_method {
    PURGE   1;
    default 0;
}

server {
    ...
    location / {
        proxy_pass http://backend;
        proxy_cache cache_zone;
        proxy_cache_key $uri;
        proxy_cache_purge $purge_method;
    }
}
```

https://habr.com/ru/post/428127/

Кеширование на сервере

Походы в базу данных могут быть достаточно дорогими.

В этом случае, результаты запросов имеет смысл сохранять в кеш.

Самыми распространенными вариантами являются:

- Храниние данных в ОЗУ/shared memory
- memcache
- redis

redis vs memcache

Redis:

- Большое колличество команд
- Есть встроенный lua
- Сложный типы данных
- Есть персистентный режим
- Кластерный
- Сложный в настройках

Memcache:

- Простой
- Многопоточный
- Для кластерных конфигураций есть moxy.
- Tarantool поддерживает протокол memcache

Cache-miss

```
AverageTime = DbAccessTime * CacheMissRate + CacheAccessTime
```

Пусть:

DbAccessTime = 100ms CacheAccessTime = 20ms

Тогда при CacheMissRate > 0.8 - кеш вреден!

Перестройка кеша (Thundering herd)

При отсутствии ключа есть большой риск перегрузить базу. Для избежания проблем с перегрузкой базы необходимо ставить локи.

Хорошие пример - новая запись в блоге под высокой нагрузкой.

```
Получаем доступ к кэшу cache, его срок жизни истёк.
Пытаемся заблокироваться по ключу user_cache_lock.
 Не удалось получить блокировку:
 ждём снятия блокировки;
 не дождались: возвращаем старые данные кэша;
 дождались: выбираем значения ключа заново, возвращаем новые данные
(построенный кэш другим процессом).
 Удалось получить блокировку:
 строим кэш самостоятельно.
```

Старт с холодным кешом

После аварии кеш, скорее всего будет инвалидирован. А в случае неперсистентных хранилищ кеша не будет точно при пропадании питания.

Подняться с непрогретым кешом сложная задача.

Общий рецепт:

- Заранее напишите скрипт прогрева кешей
- Возвращайте нагрузку плавно
- Помните о правиле №1

Инвалидация кеша

- Полнота инвалидации
- Избыточность инвалидации
- Сложность механизма инвалидации

Инвалидация по времени

- Указываем TTL
- После устаревание, автоматическое обновление

Данные могут быть не консистентны при большом TTL.

При малом TTL будет высокий cache miss.

Инвалидация по событию

Удаляем данные из кеша при изменении в базе

Опасно из-за риска мгновенной инвалидации.

Инвалидация по ключу

- Меняем данные в базе
- В ключ включаем версию
- Версию храним отдельно
- Старые версии протухают со временем

Плюс - высокая консистентность.

Тегирование ключей

```
Версии тэгов:
  tag1 -> 25
  tag2 -> 63

Кэш выборки:
  [
    срок годности: 2008-11-07 21:00
    данные кэша: [
                      …
                  ]
    тэги: [
         tag1: 25
         tag2: 63
          ]
   ]
```

Консистентное хэширование

Кэш или база? Tarantool!

- Кеш
- Персистентный
- ACID
- Репликация
- Хранимые процедуры

Может быть репликой MySQL.

Реальные кейсы

Вопросы?

Результаты занятия

- Рассмотрели клиентское кеширования.
- Рассмотрели серверное кеширование
- Изучили проблемы кеширования.
- Поговорили о реальных кейсах.

Опрос

Заполните пожалуйста опрос

https://otus.ru/polls/3938/

Спасибо за внимание!