> author：编程界的小学生
>
> date：2021/02/08

1、反向代理如何高可用?

采取keepalived可以保证nginx的高可用，可以弄两台nginx互为主备。也可以上边在挂一层lvs，lvs由keepalived保证高可用。

2、session一致性的问题如何解决?

- session同步法

> 所有的web服务器（比如tomcat）都包含session，性能不好，因为多台web服务器之间要进行session同步。

- 客户端存储法

> session存储在客户端的，缺点：每次http请求都需要携带session，占用外网带宽，并且session来回在网络上传输，不安全，容易泄露被篡改等。

- 反向代理hash一致性

> 比如nginx按照ip进行hash分配请求，保证同一个ip的请求落在同一个web服务器上。

- 后端统一存储法

> 将session统一存储在后端，比如redis，MySQL等。

3、cdn是什么?有什么用?

CDN的核心是就近访问，降低网络拥塞，提高用户访问速度。比较适合存储静态资源（比如css js 静态html等）来加速。

4、动静分离

4.1、什么是动静分离?

动静分离也就是动态页面(需要从服务端获取数据然后渲染出来的页面)进行静态化。

4.2、如何实现动静分离?

提前从服务端获取数据把页面渲染出来，形成html当静态页面进行缓存，下次在请求直接返回html，不再次发生请求到server层获取数据。

4.3、动静分离适合哪些场景?

那些更新频率很低的页面，比如：

- 商品详情页
- 城市页

4.4、动静分离不适合哪些场景?

更新频率很高，或 返回的结果集很大的，比如：

- 帖子
- 回复评论
- 搜索页

5、如何保证三高?

- 高可用：冗余+故障自动切换。也就说不单点部署，部署N个节点，然后故障后自动切换。
- 高性能：避免锁+线程池+连接池这些保证高性能。
- 高并发：垂直拆分+水平拆分+集群部署。

6、微服务拆分粒度越细越好?

不是，一般按照功能模块拆分，一个功能一个微服务最好，这样可能造成微服务过多，很乱，所以可以弄个网关统一对外提供请求。

7、聊聊负载均衡动态算法

静态配置权重或者轮询或者随机等不BB。

- 动态负载：默认给个权重值，然后当大量请求进来的时候负载均衡工具会看这几台机器哪个扛不住了，哪个机器空闲资源最多，会优先分配给空闲的服务器，虽然权重值一样，但是也不会一直轮询把一个机器打死。
- 过载保护（系统阈值）：虽然有了动态负载，但是请求量就是大，服务器都忙不过来了，但是还在一直发请求，这时候就GG了，这时候需要过载保护，算法有静态的设置阈值和动态的估算阈值，那显然动态的比较好，超过阈值后不再接收请求一段时间，相当于熔断。

动态原理：

请求进来后看看是否成功，成功加小分（比如1分），失败扣大分（比如10分），然后根据这个动态重新调整权重值。比如开始三台机器权重都是60%，那么第一台失败了好几次，可能权重值就变成了20%，所以请求就少了，等他一直成功把积分加回来的时候在重新调整权重值。

8、如何提升数据库的查询性能?

- 可以为从库建立索引，主库不建立。因为索引会影响写性能。
- 缓存，需要考虑缓存雪崩的情况。
- 增加多个从节点，需要考虑主从延迟问题。

9、聊聊数据库的垂直拆分

将一个字段特别多的大表拆分成两个小表，将字段长度短的、访问频率高的放到一个表内，其余字段长度长、访问频率低的放入另一个表内。

为什么要把短的访问频率高的放入一个表?

因为数据库缓冲池是以行为单位缓冲数据的，这么做可以让缓冲池放入更多的数据，且命中率较高，因为都是高频被访问的，不浪费空间。这样能减少IO，也大大增加了数据库吞吐量，这就是要垂直拆分的根本原因。

二、缓存

1、进程内缓存

1.1、什么是进程内缓存?

例如：

- 可以是一个带锁的Map
- 第三方库，比如leveldb

1.2、进程内缓存能存储什么?

- json数据
- html页面
- object对象
- ...等等

1.3、进程内缓存有什么好处?

就是起到缓存的作用，可以减少db的访问次数，跟jvm内存缓存、Redis缓存等一样。但是他比Redis等这些中间件好处在于节省了网络开销所以延迟更低。

1.4、进程内缓存如何保持一致性?

可以通过MQ来保证最终一致性。也可以放弃实时性，用定时器定期从后端更新数据到进程内。

1.5、使用建议

尽量别用，因为违背了站点无状态准则，而且不如Redis/memcache这种大型缓存集群靠谱。

2、