分布式/集群等高级主题

Session 集群同步问题。

首先 Session 集群的话会有两种方式存放 Session。

一种是将同一个用户每次都负载均衡到同一个请求处理服务器，这样不需要考虑 Session 同步的问题

一种是每台 Session 服务器都保存所有的 Session 信息，当一台 Session 服务器中的 Session 有更新了就将更新同步到其他所有的 Session 服务器上

Session 的持久性类型

内存

数据库

文件系统

负载均衡原理

负载均衡有硬件和软件两种，一般用的都是基于软件的负载均衡。软件负载均衡一般分为四层和七层：

四层的工作在 TCP/IP 协议栈上通过修改数据包的 IP 和 端口号 来转发，效率相对七层的来说会更高一点，一般在前端服务器中使用。LVS 就是用这种方式实现的。

七层的工作在应用层，可以做到更智能的负载均衡。一般部署在前台服务器和应用服务器之间。Apache 和 Nginx 都是七层的。

负载均衡算法

随机：负载均衡方法随机的把负载分配到各个可用的服务器上。

轮询：按顺序将新的连接请求分配给下一个服务器

加权轮询：每台服务器接受到的连接数按权重分配，一般是用在应用服务器的处理能力大小不同的情况下。

最少连接：把新连接分配给当前连接最少的服务器

BLABAL...

分布式数据库

分布式数据库提供了原来集中式数据库不具备的高可用性和拓展能力

分布式数据库的架构一般有以下几个

厂商提供的集群产品

数据分片。将原来集中式数据库中的数据进行切分，分别存放在不同的服务器中。

优点： 集群扩展能力很强，几乎可以做到线性扩展，而且整个集群的可用性也 很高，部分节点故障，不会影响其他节点提供服务

缺点：如果需要跨不同的节点做 join，或者统计类型的操作，将会变得非常困难

读写分离。使用数据库的复制技术，每台服务器上都有完整的数据副本。读和写分布在不同的处理节点上。比较适合读操作较多的应用

优点：可以通过增加提供读操作的服务器数量来线性增加读操作的性能

缺点：每个节点都必须保存完整的数据，在数据量很大时集群的拓展能力还受限于单个节点的存储能力。不适合写操作较多的应用

一般来说缓存 + DB 的读写分离架构用的是比较多的。缓存层使用 key-value 的形式来承载大量的读操作，DB 承载数据持久化。一般过程如下（基于 memcached）：用户的读请求会首先访问缓存，如果缓存命中，则返回；如果没有命中则将数据加载到缓存中再返回。对于插入、修改、删除操作，一般使用延缓加载的策略，也就是直到下次读取时才更新缓存。（删除和修改操作会将缓存中的数据标记为已失效）