- **头条篇**

6.1.0 5 个人去一个海岛寻宝，最后一共找到了 100 枚金币。他们约定了一个分配方案。
6.1.1 给你一个有序整数数组，数组中的数可以是正数、负数、零，请实现一个函数，这个函数返回一个整数：返回这个数组所有数的平方值中有多少种不同的取值。
6.1.2 一个环有 10 个节点，编号 0-9。从 0 点出发，走 N 步又能回到 0 点，共有多少种走法？
6.1.3 一个乱序数组，求第 K 大的数。排序方式使用字典序。
6.1.4 一棵二叉树，求最大通路长度。（即最大左右子树高度之和）
6.1.5 进程和线程的区别，使用线程真的能节省时间？
6.1.6 go 协程的调度方式，使用协程真的能节省时间？
6.1.7 水平触发边沿触发的区别？在边沿触发下，一个 socket 有 500 的数据，已读取 200 然后不再处理，是不是剩下的 300 就永远无法读取？
6.1.8 有函数如下，输入 1，返回什么？
6.1.9 设计 http 协议，A 端发送 AAAA，至少让 B 端知道 AAAA 已发送完成。
6.2.0 流量总入口为 api_gateway，api_gateway 挂了会导致全部挂挂，用什么机制增大可用性？
6.2.1 mysql 为什么要用 b+树，不用平衡二叉树做索引结构？
6.2.2 创建数据库索引应该怎么考虑？
6.2.3 使用 int 做 primary key 和使用 string 有什么优劣？
6.2.4 数据库分表的方法？
6.2.5 表结构，订单纪录如下，写一个语句，求卖的最好的 top 10 product_id。
6.2.6 微服务，A 服务请求 B 服务 B1 接口，B1 接口又请求 A 服务 A2 接口。会不会有问题？
6.2.7 不使用高级工具，只使用 Linux 自带的工具，你会如何 debug?
6.2.8 如何预估一个 mysql 语句的性能？
6.2.9 go 函数中，返回值未命名，发生了 panic，但是在函数内 recover 了。函数返回什么值？
6.3.0 socket 中，在 tcp 协议层面，数据分为 10 个报文发放。1-7 次很顺利，第 8 次丢失。这次通信一定失败吗？如果第 8 次数据会重发，那在接收端是不是：先读取到 1-7 次的数据，然后读取到 8-10 次的数据?还是 9-10 次的数据会先到达？
6.3.1 free -h，buffers 和 cached 有什么不同
6.3.2 后台进程有什么特点，如果要你设计一个进程是后台进程，你会考虑什么
6.3.3 僵尸进程是什么，如果产生一个僵尸进程，如何查找僵尸进程
6.3.4 孤儿进程是什么
6.3.5 一个进程有 20 个线程，在某个线程中调用 fork，新的进程会有 20 个线程吗？
6.3.6 tcp/ip 流量控制和拥塞控制
6.3.7 301/302 有什么区别？应用上有什么异同。
6.3.8 50X 相关错误码的内涵是什么？
6.3.9 close wait 和 time wait 是什么？如何排查？有什么意义？
6.4.0 http req 和 resp 的中数据有哪些
6.4.1 什么是连接的半打开，半关闭状态
6.4.2 假如一个业务依赖单点 redis，此 redis 故障将导致业务不可用，如何改进
6.4.3 redis sharding 有哪些做法
6.4.4 当大量数据要求用 redis 保存，单机单点难以满足需要，设计（换寻找）一个负载均衡的方案

6.4.5 当 redis 采用 hash 做 sharding，现在有 8 个节点，负载方案是 pos = hash(key) % 8，然后保存在 pos 节点上。这样做有什么好处坏处？当 8 个节点要扩充到 10 个节点，应该怎么办？有什么更方便扩充的方案吗？（一致性 hash, presharding）

6.4.6 如何保证 redis 和数据库数据的一致性。比如用户名既保存在数据库，又保存在 redis 做缓存。有如下操作 update_db(username); update_redis(username)。但是执行 update_db 后故障，update_redis 没有执行。有什么简单办法解决这个问题。

6.1.0 5 个人去一个海岛寻宝，最后一共找到了 100 枚金币。他们约定了一个分配方案。

6.1.1 给你一个有序整数数组，数组中的数可以是正数、负数、零，请实现一个函数，这个函数返回一个整数：返回这个数组所有数的平方值中有多少种不同的取值。

6.1.2 一个环有 10 个节点，编号 0-9。从 0 点出发，走 N 步又能回到 0 点，共有多少种走法？

6.1.3 一个乱序数组，求第 K 大的数。排序方式使用字典序。

6.1.4 一棵二叉树，求最大通路长度。（即最大左右子树高度之和）

6.1.5 进程和线程的区别，使用线程真的能节省时间？

6.1.6 go 协程的调度方式，使用协程真的能节省时间？

6.1.7 水平触发边沿触发的区别？在边沿触发下，一个 socket 有 500 的数据，已读取 200 然后不再处理，是不是剩下的 300 就永远无法读取？

6.1.8 有函数如下，输入 1，返回什么？

6.1.9 设计 http 协议，A 端发送 AAAA，至少让 B 端知道 AAAA 已发送完成。

6.2.0 流量总入口为 api_gateway，api_gateway 挂了会导致全部挂挂，用什么机制增大可用性？

6.2.1 mysql 为什么要用 b+树，不用平衡二叉树做索引结构？

6.2.2 创建数据库索引应该怎么考虑？

6.2.3 使用 int 做 primary key 和使用 string 有什么优劣？

6.2.4 数据库分表的方法？

6.2.5 表结构，订单纪录如下，写一个语句，求卖的最好的 **top 10 product_id**。

6.2.6 微服务，**A** 服务请求 **B** 服务 **B1** 接口，**B1** 接口又请求 **A** 服务 **A2** 接口。会不会有问题？

6.2.7 不使用高级工具，只使用 **Linux** 自带的工具，你会如何 **debug?**

6.2.8 如何预估一个 **mysql** 语句的性能？

6.2.9 go 函数中，返回值未命名，发生了 **panic**，但是在函数内 **recover** 了。函数返回什么值？

6.3.0 socket 中，在 **tcp** 协议层面，数据分为 **10** 个报文发放。**1-7** 次很顺利，第 **8** 次丢失。这次通信一定失败吗？如果第 **8** 次数据会重发，那在接收端是不是：先读取到 **1-7** 次的数据，然后读取到 **8-10** 次的数据**?**还是 **9-10** 次的数据会先到达？

6.3.1 free -h，**buffers** 和 **cached** 有什么不同

6.3.2 后台进程有什么特点，如果要你设计一个进程是后台进程，你会考虑什么

6.3.3 僵尸进程是什么，如果产生一个僵尸进程，如何查找僵尸进程

6.3.4 孤儿进程是什么

6.3.5 一个进程有 **20** 个线程，在某个线程中调用 **fork**，新的进程会有 **20** 个线程吗？

6.3.6 tcp/ip 流量控制和拥塞控制

6.3.7 301/302 有什么区别？应用上有什么异同。

6.3.8 50X 相关错误码的内涵是什么？

6.3.9 close wait 和 time wait 是什么？如何排查？有什么意义？

6.4.0 http req 和 resp 的中数据有哪些

6.4.1 什么是连接的半打开，半关闭状态

6.4.2 假如一个业务依赖单点 redis，此 redis 故障将导致业务不可用，如何改进

6.4.3 redis sharding 有哪些做法

6.4.4 当大量数据要求用 redis 保存，单机单点难以满足需要，设计（换寻找）一个负载均衡的方案

6.4.5 当 redis 采用 hash 做 sharding，现在有 8 个节点，负载方案是 pos = hash(key) % 8，然后保存在 pos 节点上。这样做有什么好处坏处？当 8 个节点要扩充到 10 个节点，应该怎么办？有什么更方便扩充的方案吗？（一致性 hash, presharding）

6.4.6 如何保证 redis 和数据库数据的一致性。比如用户名既保存在数据库，又保存在 redis 做缓存。有如下操作 update_db(username); update_redis(username)。但是执行 update_db 后故障，update_redis 没有执行。有什么简单办法解决这个问题。