<http://www.cnblogs.com/felixfang/p/3420462.html>

内存分页管理

详细的问了一下当前的project设计，之后一道design，分布式文件系统，类似于有10个node，每个node有1TB的文件存储空间，这个cluster给单一的客户用，要支持上传+删除+覆盖文件（每个文件大小固定64KB），但是要求最给这个客户4TB的空间，问怎么设计？其实就是HDFS吧，每个文件有最多2个副本，怎么把文件distributed（range v.s hashing v.s. cosistent hashing），这些文件的信息需不需要存到memory里面方便查找，如果是需要什么信息，如果同时支持删除和新增文件如何保证文件状态是正确的（用commit log，或者soft delete），总之都是很底层的HDFS设计理念

你是市长助理，设计一个 smart 红绿灯。当是提出应该apply rule：同向通行，左右转弯，安装很多传感器什么的。虽然大叔频频点头，但是一直在提醒我还剩多少时间。  
后来想明白了，我说的没错，但不是大叔要的。他可能相要：分成3个子系统，1信息收集，2信息处理，3如何优化。

设计netflix：如何实现HA，数据库选哪个，QPS怎么考虑，如何估算每个service需要多少个instance，如何authentication，如何实现“从上次落下的地方继续播放”，如何存储电影文件。问题并不难，所有搭建过microservices和了解distributed system的人都应该知道。

如何设计微信

设计一个youtube

设计一个亚马逊评论系统，不仅提供亚马逊网内的评论综合，还要抓取其他网站，比如yelp和别的。

设计chat sy[stem](http://www.1point3acres.com/%E7%BE%8E%E5%9B%BD%E5%9B%BD%E5%9C%9F%E5%AE%89%E5%85%A8%E9%83%A82012%E5%B9%B4%E5%BA%A6%E6%9B%B4%E6%96%B0stem%E4%B8%93%E4%B8%9A%E5%90%8D%E5%8D%95-%E7%9C%8B%E7%9C%8B%E4%BD%A0%E7%9A%84%E4%B8%93%E4%B8%9A/)，其中问到太大并发和局部异常如何处理，如何能够精确找到对方，聊的过程中忘了分析在线和不在线的情况怎么处理了

设计一个照片分享系统，最主要的是如何优化

要在网页上显示ku cun信息，数据从好几个数据库里面抓，当然速度慢，但是，数据库不能动它因为还有别的系统依赖，怎么优化。不就是要加缓存吗。各种讨论优化，非常会引导我，跟这样的面试官交流简直是一种享受。都是些开放性问题，答的有理有据就行，最后分析下trade off

现在设计个爬虫

如果有很多台机器上面都有log, 设计一个系统来处理他们．题目就这样，没有给任何要求，所以我就一直问说，想要什么样的需求，譬如说纯粹只是要把Log内容转存到DB, 还是需要count top K activities, 考官说要存起来就好，之后想search particular strings. 所以我就先回答说在各个机器上面写个python program在local parse the log and send the data to the web servers, 考官一直问说怎么跑python program, 我回答用cron job每小时parse一次他觉得满意．如果loading很大无法在local parse该怎么办，我就回他说把log copy 到 several web servers然后在server parse. 问怎么存到DB, DB容量不够怎么办，怎么map DB, consistent hashing, map该存在哪里, add cache layer, 怎么选择DB, depands on your data type, key value pair data: use NoSQL, relational data: MySQL or Postgres. 如果round down to 3 DB choices, 该用什么来评断你要怎么选DB? 我回不是很好，回说看大公司用相同类似功能的公司都怎么选，譬如说ＦＢ在做News Feed用MySQL, 若好读也在做news feed那就也可以考虑用MySQL, 还有MySQL在做replica很方便，我会在网路上多做研究才决定怎么选择．考官觉得ＯＫ. 这轮问蛮多问题的，但就是从1个web server开始画，多个ＤＢ然后再加cache就可以了.

plagiarism checker sy[stem](http://www.1point3acres.com/%E7%BE%8E%E5%9B%BD%E5%9B%BD%E5%9C%9F%E5%AE%89%E5%85%A8%E9%83%A82012%E5%B9%B4%E5%BA%A6%E6%9B%B4%E6%96%B0stem%E4%B8%93%E4%B8%9A%E5%90%8D%E5%8D%95-%E7%9C%8B%E7%9C%8B%E4%BD%A0%E7%9A%84%E4%B8%93%E4%B8%9A/)，从简单的讨论，然后到分布式，讨论了service和database的设计

主要思路是按文章类别存不同的数据库，每条数据存的是每个段落的word和数量(remove停用词)，给一篇文章，先判断出哪个类别，然后在对应的数据库里 与每条数据计算相似度，发现相似度很高的就有剽窃的嫌疑了，主要有classification service, similarity service, verification service等等。面试官问了关系数据库和nosql的区别，这里用哪种比较好

设计kindle highlight service，用户在看一本书的时候要能highlight， 能显示用户以前的highlight以及显示其他用户读这本书的topk个highlight。

design video streaming sy[stem](http://www.1point3acres.com/%E7%BE%8E%E5%9B%BD%E5%9B%BD%E5%9C%9F%E5%AE%89%E5%85%A8%E9%83%A82012%E5%B9%B4%E5%BA%A6%E6%9B%B4%E6%96%B0stem%E4%B8%93%E4%B8%9A%E5%90%8D%E5%8D%95-%E7%9C%8B%E7%9C%8B%E4%BD%A0%E7%9A%84%E4%B8%93%E4%B8%9A/)

就是设计一个用优惠券的系统，如何产生coupon码？如何散发产生的coupon码？我在他的提示下就设计了一个数据库的表

设计聊天系统

设计twitch  重点在分析直播和录播的差异

我有一堆item，要选5个分数最高的放在[Amazon](http://https//www.amazon.com/b?_encoding=UTF8&tag=1point3acres-20&linkCode=ur2&linkId=89c11e2c5b86155c5422f19cca1e9880&camp=1789&creative=9325&node=5" \t "_blank)首页上该怎么选。我先问了他DAU什么的，算出QPS，发现一台server一台SQL数据库就够了，于是就跟他说用最简单的设计模式就行，Client -- server-- database。然后他让我写了个选top5的函数，用priorityqueue就行了。然后问我如果item太多内存放不下怎么办，我答用多台机器分开做然后K路归并，再多的话map reduce。然后让我写了k路归并，并且让我算时间复杂度。

最后还问了我这个是每个用户请求一遍就重新算一遍还是多久算一遍还是怎么设计，我答平时存在内存里就行，定期重新算。具体多久重算根据需求进行trade off，算的越频繁越费资源但是越准确

白小哥给了一个id generator的 class， 让我写unit test，写得比较慢，但都写出来了，白小哥表示满意。  
      然后直接开始问performance test怎么设计。扯了一堆基本概念。follow up是如果想用比较小的数据做压力测试，怎么搞，我说减少resource. 看起来也还算满意。

1. Design a Uber 设计一个简单的Uber，包括检测周围空闲的车，用户打车付账流程和到目的地时间估计.  
(将城市化成许多个矩形block（区），可以借鉴二维k-d tree那个思想。每个车实时更新当前位置坐标和是否available，找用户最近八个区的空闲的车，然后时间就是车速和距离的关系，这个没错。地图api这种你需要什么和interviewr说就好了，如果不是考察项目的话一般都会说可以默认给出的。)  
  
2.TinyURL  
(Write heavy? improve Security? 怎么scale? 一个region上的服务出问题了怎么处理?)  
  
3. Repository system, design commit fuction and branch function.  
-google 1point3acres  
4. Video/Movie System (given a list of videos, return top 5 rated videos)  
  
5. Sotre the livestreaming video and watch it later function  
  
6. [cc150](http://https//www.amazon.com/Cracking-Coding-Interview-Programming-Questions/dp/0984782850/?&_encoding=UTF8&tag=1point3acres-20&linkCode=ur2&linkId=9b955ce5e3d3a4ad316bdbd14bc87c3e&camp=1789&creative=9325) JukeBox  
  
7. [Amazon](http://https//www.amazon.com/b?_encoding=UTF8&tag=1point3acres-20&linkCode=ur2&linkId=89c11e2c5b86155c5422f19cca1e9880&camp=1789&creative=9325&node=5) gift card printing machine. (实就是general的说一下你对一个application architecture的理解，面试官会引导你，比如用啥样的db，用rest/soap，某一步失败了怎么办，data consistency一类的随便扯一扯). 鍥磋鎴戜滑@1point 3 acres  
  
8. One hour delivery system  
  
9. Explain Agile, Waterfall, Pro & Cons  
  
10. Predict User purchase  
(先分析什么因素判断用户买不买这个商品，浏览记录，购买记录，在页面停留时间，浏览这类商品的次数，现在火的top 100商品等等。然后分析架构，  
给的答案是首先master slave避免single point failure，用户点击商品后先通过dymanic dns look up找到距离最近的CDN，然后http request传过来给那个cluster的master server, mater本身有cache看看这个请求的结果是不是已经cache过了有的话直接返回（这里cache的是这个请求对应的购物车html页面），没有的话master做负载均衡下传给空闲slave server（rmi call）, slave有自己的local cache因为对这个预测结果每个slave cache可以不consistent， 可以不用时刻recon每个不同的server cache。所有的数据存储都用in memory database并设置time to live， 因为这个是一个读取大于写的系统数据也不需要持久化不用支持transaction, scale也更容易。master如果挂了重启就可以，因为都是预测数据丢失了也无所谓。如果要更优化可以在浏览器端也做一层cache，如果用户反复点击同样的商品，就不用每次都make http call了)  
  
11. Card game , and write shuffle method.鐣欏璁哄潧-涓€浜�-涓夊垎鍦�  
  
12. Amazon Locker 就是Amazon买东西可以运到一个Locker然后pick up的那个.  
(仔细想一下你就会发现就是一个Parking Lot.Package有Small,Medium,Large.一个Location的 Lockers也有Small,Medium, Large。面试官主要想知道一个送货小哥去的时候怎么分配给他个大小合适的Locker。要写那个method。我就按照Parking Lot做的。我觉得一模一样。恩恩，我那时侯一开始考虑也想是不是Order生成的时候就匹配了一个Locker，还有挑选哪个Location,但跟面试观官交流以后，他就说假设只有一个Location然后主要想知道送货小哥去的时候怎么分配，别的先不考虑。)  
  
13. Reader System  
  
14. Parking Lot, Airport etc...([http://stackoverflow.com/questio ... n-an-oo-parking-lot](http://stackoverflow.com/questions/764933/amazon-interview-question-design-an-oo-parking-lot" \t "_blank) )  
  
15. Amazon address book  
   (1. What's in web server. more info on [1point3acres.com](http://1point3acres.com/)  
    2. What's the API for address service  
    3. What's in the storage  
    4. How to improve the performance)  
(user 发送与address 相关的请求到web server，　然后web server 获取／更新相关的记录。大概说了下DNS根据user ip访问临近的web server, 问到了web server 里头有哪些与address 相关的function 以及参数，具体说了下getAddressLists. 期间提了下cache 面试官表情不对了。后边问了有多少个表，怎么设计，怎么提高profermance. 楼主提到说作no sql 做sharding。面试官说时间到了没有给反馈，多半是设计太狗血)  
  
16. Design system to store user info and address info. Address info changes frequently. Need to notify address change.  
  
17. Design a Log application. (就是开发从上面获取 bug log 信息，用户可以往上提交。开始我并不知道log application怎么工作的，是程序员往上提bug 还是test人员往上提。跟面试官一步步讨论的，然后写了些前后端的 process。中间很多问题，他们会顺着你的思路往下问). 鍥磋鎴戜滑@1point 3 acres  
  
18. 给了一堆 商品，每个有不同的分类tag 比如 book, music, sports... 然后按顺序输出，就是输出这里他描述的特别不清楚，于是开始先按tag sort了，然后顺着输出。他说不行，想要控制每行的个数，然后就控制个数输出。然后他比较满意，follow up 了随意更改每行的个数，不要求写代码，也写上去了  
  
19. OOD。设计一个SQL parser（不是compiler，这里强调如何解析SQL语句中的变量，比如table name，和关键字， 比如select 语句），关键在于怎么用好interface.鐣欏璁哄潧-涓€浜�-涓夊垎鍦�  
. 鐣欏鐢宠璁哄潧-涓€浜╀笁鍒嗗湴  
20. 设计一个big integer类。实现其中的add和multiply。先是讨论了要使用的数据结构（数组和链表）并对比优缺点，然后用数组实现（非高效压缩那种的，一个数组元素是大数里的一位数）。multiply只要写伪代码，出了错，不过他不介意  
  
21. 设计个API，满足两个function，一个是往list里面丢string，还有一个是统计top k frequent element. 对于API实现scalability.  
. visit [1point3acres.com](http://1point3acres.com/) for more.  
22.  Design pattern: strategy, observer   设计模式那个就是duck/toyduck的变形。Observer那个也是比较教科书的东西. from: [1point3acres.com/bbs](http://1point3acres.com/bbs)   
  
23. 如果有一个service, 要求设计方法支持query，比如最后一秒的访问数，最后一分钟的访问数，。。。。最后一小时的访问数。。。  
(1.把timestamp 写到磁盘上，然后用hadoop 来算。面试官问pros and cons. 1point 3acres 璁哄潧  
2. 为了改善读的速度，我说把这些timestamp存到in-memory buckets里面，最后还是 hadoop  
根据我的buckets排序思路提示我，可以不用存timestamp。。。。我有点惊呆了，但是转念一想，你这些query的时间段是last second/minute/hour。我觉得就可以只有这三个buckets，每次call 这个service的同时，检查当前的timestamp，如果超过1秒，我说就去aggregate/update last minute和last hour的总数就好了。。。 到最后我脑子里有点捣浆糊了，但是面试官看到我最后的思路解释说可以work))  
  
24. 手机公司的bill系统，手机计划有免费时段，比如晚7点到第二天早7点和周末免费。短信和数据有各自的价格，用超过计划允许的数量怎么办。最后完成的感觉还算不错。这一轮也问了oo的一些基础知识，比如inheritance 和 composition的区别，什么时候用哪种？