Scenario:

* QPS: Twitter DAU: 150M
  + 一天点 60 下 150M \* 60 / 86400 = 100K
  + Peak: 100K \* 3 = 300K
  + Fast Growing: in 3mon 300K \* 2 = 600K
* Read Heavy: 300K QPS
* Write QPS： 5K -- 大部分人都是读

一台Web Server 承受量 是 1K

一台SQL 承受两 是 1K

一台 NoSQL （cassandra）大概是 10K QPS

MemcacheD：是 SQL 的 1000倍，大概是 1Million QPS

* 描述最重要的feature ()：
  + Post a tweet
  + **Timeline： 某一个用户所发的所有帖子**
  + **NewsFeed：我朋友 和 我 发的 所有 帖子按照某种顺序排列的整合**
  + Follow / Unfollow user
  + Register / Login

## Service:

* UserService: Register / Login
* TweetService: Post a tweet / News Feed / TImeline
* MediaService: Upload Image/ Upload Video
* Friendship Service: Follow / Unfollow

Service 是 逻辑处理的单元，分开去做

## 

## Storage 存储：

* SQL Database / NoSQL
  + User Table 适合 放在 SQL中（年龄，性别，身高，体重，一个 class里面有一些 fields）
  + Tweets / Followers 适合 放在 NoSQL Database
* File System
  + 非结构化数据：存储 图片/视频 Media Files（比如 Amazon S3 就是 分布式文件系统）

文件系统是放在 数据库 系统上，文件上面包装了 一层 DB

对于一个 文件：我们只能根据一个 file path，去：

* get://filepath
* set://filePath

提供的option 比较少

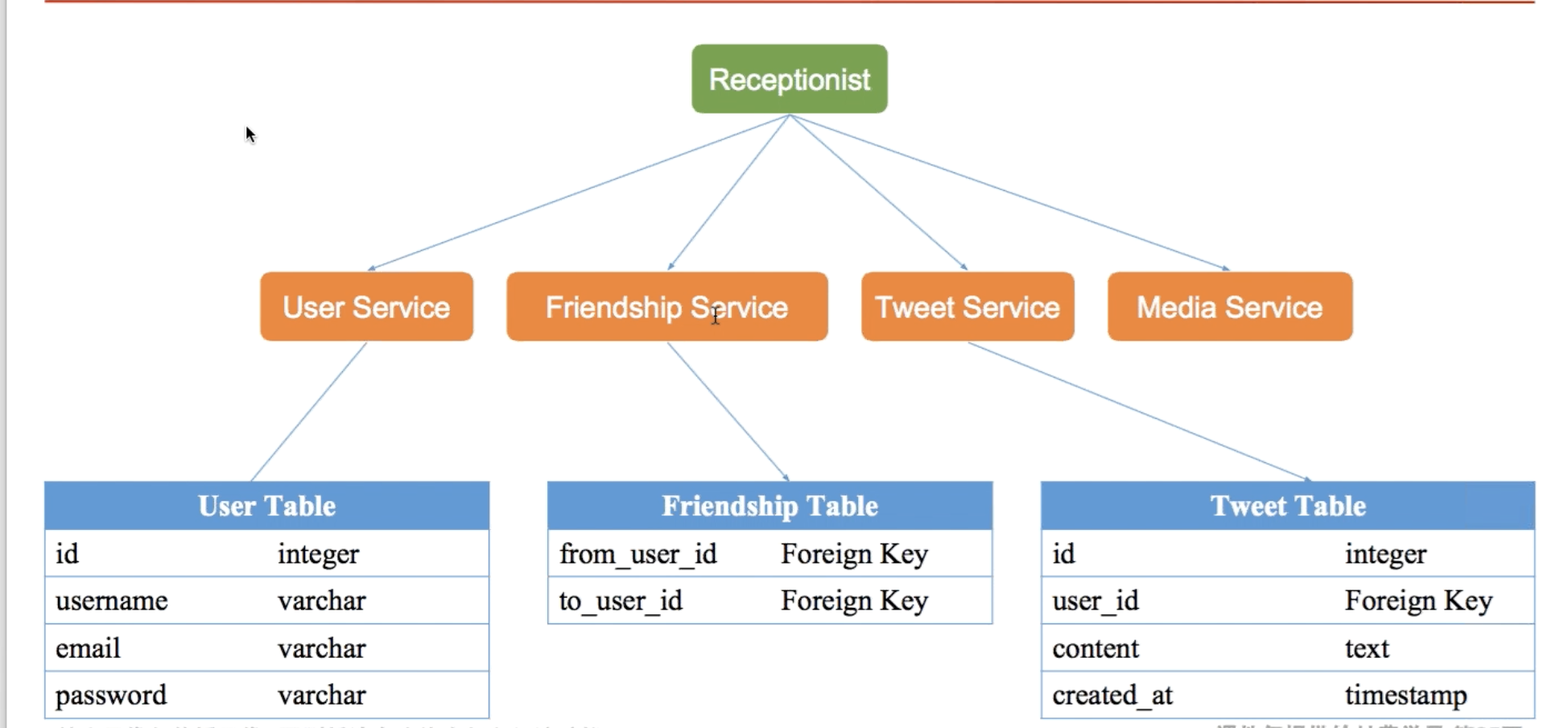
所以我们希望通过 DB 对 file system进行包装，进行 查询。Ultimately 还是 通过 File System去获取

系统 = 服务 + 数据存储

service里面包含一些 logic 和 数据存储

分好了Service 之后，一定要知道每个 service 下面对应的 DB 长什么样子

比如：



Friendship service： 我们可以存成 类似于NoSQL的 方式：

from\_user\_id,

to\_user\_id,

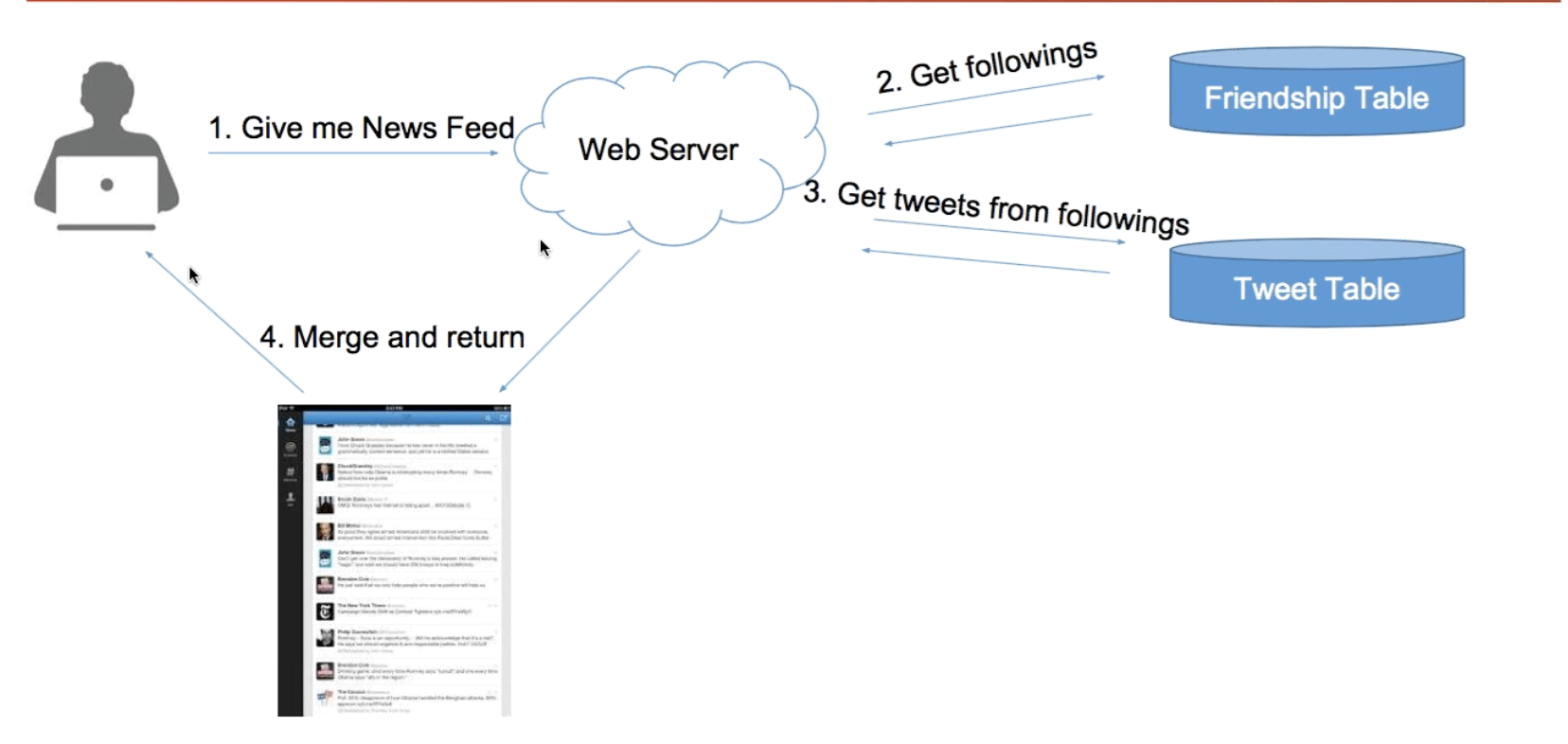
NewsFeed 如何存取？

Lady Gaga 发 Tweet 的话 会发生什么

如何存取 Likes

如何实现 Follow / Unfollow

## NewsFeed 如何存取是 最重要的 Interview 问题：

* FB 和 博客系统 的 区别 就是 每个人的 信息流都不一样
* 典型的新鲜事 系统？
  + Facebook
  + Twitter
  + 朋友圈
  + RSS Reader -- 唯一作用就是成为了一个面试题。新闻源不会很多。RSS相当于 大V，可以针对只有大V的情况进行优化
* 根据前面的 table， 最naive的写法是：
  + 在friendship table中找到 friend list
  + 根据friend list 在 tweet table里面找，然后排序
  + 算法：获取每个好友 前 100条 tweet，合并出前 100条 News Feed
  + K 路归并： 100K LogK
  + 复杂度分析：
    - 假如有N个关注对象，则为 N 次 DB Reads 的 时间 + K 路归并的时间（可以忽略）
    - Post a tweet = 1次 DB Write 的时间
* 用流程图的话：  
  
* 先拿到 all friends，然后 get Tweets from all friends
* 问题： 慢！！！虽然可以用 SQL IN query，但是还是慢。因为并没有 利用到 index的 特点 （userId 被index了，但是我们却需要get到多个 userId 对应的 tweet）
* 并且这个过程 发生在 信息获取过程中，用户一直在等着，Pull 模型

还有一种Storage 模型： Push 模型：

* 只要有好友发了帖子，就把这个帖子送到给 它的 可能受众
* 这个时候 建立一个 新的 table叫做 News Feed Table：
  + id
  + owner id : foreign key （某个 受众的 id，lady gaga的 follower）
  + tweet id： foreign key
  + created at： timestamp
* owner id 会 被 index， 查起来就会很快，但是也只是 查 ownerid == 某一个 value的时候，才很快， 如果查一个 范围 还是 很慢
* 如果 一个 人 发了 post，就 把它 的 follower 每个 人 都 添加 一个 news feed table entry
* 这个名字叫做 fan-out（一个一个把信息推送给需要的人）-- 把读的慢，变成了 写的慢
* 这个时候，
  + 如果读的时候，只需要 1次 DB Read owner == 郭靖
  + 如果是 写的 时候，需要 写 N 次 DB Write （并且这个过程可以 异步执行，也就是 大V 发帖了之后，立即显示成功，在后台复制很多分，后台执行
  + 在 系统里面需要 Async Tasks Server，用来执行那些**慢**的 任务
    - 生活中，有哪些任务是异步执行的，发邮件 就是异步的，因为要提供重试机构；OJ的评测也是异步的
    - 需要用 MessageQueue<Tasks> 来更新写操作：
      * RabbitMQ
      * Kafka
      * SQX MQ
      * Redis也可以做 Message Queue，是一个生产者/ 消费者模型。根本原因是 生产速度和 消费速度不一样。
      * 现在存 不会 存在 存不下的问题，取的时候怎么快 才是问题
* Push 模型有什么缺陷：
  + Duplicate Data（其实这个不重要，不会影响到用户访问的性能）：
    - 很多时候直接 在 News Feed Table 直接 存 tweet\_data（只有140个字节） ， 或者 tweet\_id 对应的 推文内容存在 cache 里面
  + 主要 缺陷： delay：吴亦凡发个帖子
    - follower 太多了， （FB 和 wechat 好友上限设置了2000，但是对于Twitter这种 单向关注的 大V 粉丝很多）， 比如 有的粉丝 1个小时之后才能 收到更新

Pull vs. Push:

* 业界：清一色都用 Pull
* Push 的 delay 缺陷 太难解决
* 解决Pull的 缺陷的方法：
  + 最慢的部分发生在用户读的时候， 如果访问的不是 DB，而是 Cache，访问数据就快了 上千倍，可以cache 什么东西：
    - cache 每个 用户的 timeline
    - 这样N 次 DB query 就变成 了 N 次 cache get
    - 为了防止数据库 爆掉，就cache 最近 1000 条，最近3个月
    - 同时还要cache 每个用户的news feed
      * 对于没有cache news feed 的用户，归并 N 个 用户最急你的 100条tweet，并 最后取出结果的 前 100条
      * 有 cache news feed 的用户，归并 N 个 用户在某个时间时候的所有 Tweet，并merge 进入到当前 结果中
* 解决Push 模型 缺陷：
  + Push 只是 放更多东西到 Disk中 （与 Pull模型 将News Feed 放到memory中，完全不是个事）
  + 不活跃用户： 僵尸粉 将粉丝 排序 （比如 rank by last login time）
  + 粉丝数目 followers >> following Fanout 挑战
    - lady gaga 问题 不太好解决 -- 就是有几十个 million 粉丝要 deliver。需要 2个小时100台机器
    - 怎么办？ 完全切回pull 吗？
      * 错。特殊情况 特殊处理。比如说，如果lady gaga 发帖子，标记她为明星用户， 如果用户获取消息的时候，就 特殊把 明星的 news feed 取一下，合并到 news feed table中
      * 比较土的方法是： followers > 1m。 可能会出现摇摆问题， 大家都取消关注，这个时候如果 followers 又涨上来了，就又没有

什么时候用 Push:

* 资源少
* 想偷懒
* 实时性 要求不高
* 用户发帖比较少
* 双向好友关系
* 没有明星问题

什么时候用 Pull？

* 资源充足 （很多 cache）
* 实时性要求高
* 用户发帖多 （优化写）
* 有明星问题

Scale 继续扩展：

如何实现 follow/Unfollow:

* insert 到 friendship table
* merge timeline into news feed asynchrounously
* Unfollow之后，asynchrounouly remove his tweets from your news feed

为什么需要 async：因为这个过程慢。给用户的感觉是，已经成功了。 异步的坏处是，refresh洁面，发现删除的人还是

Likes：

LIke Table：

* id, user\_id, tweet\_id, created\_at
* 如果实时计算： 如果当前 news feed 有10个 tweets，则需要对每个 tweet 查询其 likes table 中的 数量 select Count
* 可以在 tweet table 中 添加 like\_nums 直接 将 number 记录在里面 -- 一种denormalize的实现（本来要去 like table中 cnt tweet\_id == cur ) 的， 现在直接get到了。只是写的时候稍微费了点时间

Lady Gaga 发个 自拍 回发生：

* 惊群效应： thundering herd:
* 搜索 九章 中 雪崩 效应 的 文章！！