存储后端设计

1. 现存的几种云缓存

主要是国外 Amazon 和国内 sina 的云存储系列产品：

Amazon：

Appendix 1: Amazon S3, Amazon SQS, Amazon SimpleDB – When to Use Which?

The table will help explain which Amazon service to use when:

	Amazon S3	Amazon SQS	Amazon SimpleDB
Ideal for	Storing Large write-once, read-many types of objects	Small short-lived transient messages	Querying light-weight attribute data
Ideal examples	Media-like files, audio, video, large images	Workflow jobs, XML/JSON/TXT messages	Querying, Mapping, tagging, click-stream logs, metadata, state management
Not recommended for	Querying, content distribution	Large objects, persistent objects	Transactional systems
Not recommended examples	Database, File Systems	Persistent data stores	OLTP, DW cube rollups

Amazon 除上面三种产品之外还有 Amazon EC2（提供 memcached 服务）、MongoDB on AWS（名字就能看出来）

Sina：

KVDB 是 SAE 开发的分布式 key-value 数据存储服务，用来支持公有云计算平台上的海量 key-value 存储

Storage 是 SAE 为开发者提供的分布式文件存储服务，用来存放用户的持久化存储的文件

2. 场景是这样的

a. 数据量大，但单条记录并不大，主要是记录数多
b. 访问速度，需要很快的访问速度
c. 存储的数据结构简单，类似于 cookie 的结构，key、value、expires
d. 需要为数据提供有效期支持，过了有效期的可以被清除
e. 需要为永不失效的数据提供稳定的存储，也就是这部分数据持久化下来

使用场景上，我们的云存储和上述的都有所不同

3. 存储的数据结构

key	value
appid:1073:userid:6749372:emotion	{emotion:[“smile”,“cry”]}

key：是一个规则的字符串，appid 部分由缓存管理 server 管理，给响应的 app 加上相应的 appid:xxxx

value:redis 和 memcache 不一样的是，memcache 可以保存一个实现了序列化接口的对象，但是 redis 只能保存 String 和 byte[]，保存 json 是没有问题的，前端提出需求，既然是云存储就应该什么格式的数据都可以,这里还没想好，存 byte[]

key 失效时间 expires 有 2 种形式：

1、某个时间点失效，24*60*60 或者类似 2012-3-18 17:46:50 这样的一个时间戳
2、两个时间点，2012-3-18 17:46:50 到 2012-3-22 17:46:50

对于第一种情况，在 redis 中对 key 设置一下 expires24*60*60，在 mysql 数据库中用 expires 字段表示失效日期，转换成 timestamp 的格式
对于第二种的失效方式，我没有想到一个效率高的方式来支持，所以我想先不考虑这点

4. 存储技术选型

依照上述的存储格式，本身带有失效策略的 key/value 存储是非常合适的选择，所以选择 redis+mysql 的方式，redis 作为缓存，mysql 作为数据最终的持久化。

没有选择以下技术的原因：

1、Memcached：虽然 mc 是一个分布式缓存，redis 是一个 nosql 数据库，但是读写性能上 redis 不输 mc，mc 基本只支持简单的 key-value 存储，不支持枚举，不支持持久化和复制等功能，Redis 支持 list,set,sorted set,hash 等众多数据结构，以后支持更多的存储结构要方便，redis 还支持 key 和 value 的操作，提供了 KEYS 进行枚举操作，枚举出所有数据，Redis 还同时提供了持久化和复制等功能。

2、Mongodb： Mongodb 虽然有不少优点如自动分片，但是 mongodb 是基于硬盘的存储，虽然自身做了优化，会把索引和部分数据放到内存里，但是读写性能可能还不能满足需求，另外本身没有自动失效的支持。

为什么选择 mysql，因为 redis 虽然有数据落地功能，但是它是将内存中的数据映射到硬盘文件上或者追加数据的操作日志，但是这两种方式在生产环境下对保证所有数据的稳定是没有意义的，因为内存是有限的，超出内存的部分会被 LRU 掉永远找不回来同时磁盘中也同样没有这份数据，所以 redis 的数据落地功能意义在于灾后重建数据。

5. redis 的使用

1、redis 部署多个节点，不使用 master-slave 的方式，也就是每个节点都是单点的可读可写的 master，所以这里存在单点问题

2、对每一个 key，hashcode 取模，对应到某个 redis 节点

3、由于 redis 在我们应用场景里是作为 cache 而不是 storage 使用，所以 redis 单点问题造成的后果是访问到有故障的那台 redis 服务的请求都会打到 mysql 上，这时候需要重建 redis 数据

4、具体使用：

jedis.set(key,value)

jedis.expire(key,expirestime)

6. mysql 的使用

Id	key	value	expirestime
分表使用的 id	string	Varchar(4000)	timestamp

分库：读写库分离

分表：分为 64 张表，减小单张表的大小

id：业务中给分的 id，根据 id 进行分表

定时清理 expirestime 超出当前时间的记录

如果要做到能够水平的扩容，可以考虑使用 twitter 的 gizzard 用于数据复制和分区