浙江大学

本科设计报告

课程名称：大数据存储与计算技术;

姓名学号：胡若凡 3200102312
郭伟京 3200102538
管嘉瑞 3200102557
王凯昱 3190104895

学　　院：计算机学院

专　　业：计算机科学与技术

指导老师：陈华钧

2023 年 6 月 6 日

总体报告

1. 存储需求分析

1.1 数据存储需求分析

1.1.1 数据存储类型

微博内的数据可分为**用户基本信息数据、微博数据、评论互动数据。**

其中用户基本信息数据包括"粉丝表"、"关注表"等关联表，其可以通过二维表结构来逻辑表达和实现，严格地遵循数据格式与长度规范，主要通过关系型数据库进行存储和管理，属于结构化的数据。这部分数据可以使用结构化的存储系统存储。

用户发布的动态性微博与评论性微博中，数据包含图片、文本、音频等，属于非结构化的数据。这部分数据可以使用非结构化的存储系统存储。

而用户与其他微博间的互动关系，微博间的评论关系，也可通过二维表结构来逻辑表达和实现，使用关系型数据库进行存储和管理。

1.1.2 数据存储数量

根据估算，每一条微博包含一张大小为1KB的附带多媒体数据，那么1亿条微博的数据量为10^6MB，约在1TB左右.

每一条微博下的评论大约有10条，为了维护评论的对应关系，1亿条微博与10亿条评论之间产生的一条关联记录大小为1KB左右，则此关联记录大约为1TB左右。而3亿用户大约每天会和10条微博有所互动，包括点赞、收藏等行为，其互动记录大约为3TB左右。

而对于用户基本信息，假设微博用户为3亿，则用户基本信息表的记录数为3亿，假定每一条记录大小在1KB左右，则用户基本信息大小在300GB左右。

考虑到对于每个用户，需要存储其"关注表"和"粉丝表”，假定每个用户关注人数平均为100人(实际远低于这个值)可推测出用户的"关注表"和"粉丝表”大约有600亿条记录，因此其"关注表"和"粉丝表”的基本信息的大小在30TB左右。

根据以上估算，微博所有信息的总存储量**大概在PB级别。**

存储系统需要能存储以上所述的大量数据。

1.1.3 数据存储性能

对于发布的微博数据，根据估计单日1亿条微博会存入1PB左右的数据，同时这些数据可能会被百万级别的用户同时访问，而对于用户的基本信息，同一时间也同样有可能被百万级别的用户同时访问。为了保证用户的体验，我们认为访问和存储的响应时间均不应超过2s。

因此存储系统需要有极高的**IO**性能，足够大的IO带宽可以支持百万级别的IO并发查询和存储操作；存储系统选择**数据库引擎**时，需要考虑其对高并发读写操作的支持，以及针对非结构化数据的支持情况。

1.1.4 数据更新频率

微博存储系统的数据更新频率**非常高**，用户关注、取消关注、点赞、取消点赞、评论等操作也都会对数据进行更新。因此，对于微博存储系统的数据更新频率有以下几个方面的因素：

- 微博的发布频率：微博平台每天都会有大量新的微博发布，这些数据需要及时保存到微博存储系统中。
- 用户关注和取消关注频率：用户可以关注和取消关注其他用户和话题，这些关注操作会对关注表进行频繁的更新。
- 点赞和取消点赞频率：用户可以对微博进行点赞和取消点赞操作，这些点赞数据也需要及时保存到用户-微博关系表中。

因此，微博存储系统的数据更新频率非常高，要保证系统的实时性和高并发性，需要设计和实现高效的数据更新机制，并加以优化，以提高系统的性能和稳定性。

1.1.5 数据生命周期

数据生命周期是指数据从创建到删除的整个过程，包括数据的创建、存储、使用、备份、恢复和销毁的整个过程。以下是微博存储系统数据的生命周期：

- 数据创建：微博存储系统需要支持数据的实时创建，包括微博、用户基本信息等数据的创建。
- 数据存储：系统需要根据数据的类型采用不同的存储类型，如结构化数据与非结构化数据则采用不同方式存储。
- 数据使用：存储的数据需要支持快速的查询、排序等操作，并能够快速响应用户的需求。
- 数据备份：为避免数据的意外丢失，微博存储系统需要进行数据备份，包括定期的全量备份和增量备份，以确保数据的安全性。
- 数据恢复：在数据丢失或者系统故障的情况下，系统需要支持数据的快速恢复，以确保数据的完整性和可用性。
- 数据销毁：由于微博中可能包含用户的隐私信息，因此在数据到期时，对数据进行销毁时，需要遵循数据保护法规，采用专业的数据销毁工具对数据进行彻底的销毁，并在销毁的过程中确保数据的安全性

1.2 用户行为分析

在微博存储系统中，用户行为分析是指通过对用户在该存储系统中的各种行为进行分析，了解用户使用习惯、需求以及提供更好的服务等方面的需求。下面是一些可能会涉及的用户行为：

- 登录/注册：记录用户的登录/注册次数以及登录/注册时所使用的设备和IP等信息。
- 发布微博：记录用户发布微博的内容、主题、发布时间以及发布设备等信息，以便了解用户的兴趣偏好及使用习惯。
- 点赞/评论/转发：记录用户对微博的点赞/评论/转发的数量、频率、时间等信息，以了解用户对微博内容和其他用户互动的偏好。
- 关注/粉丝：记录用户关注和被关注的数量和频率，以及关注和被关注的用户的身份和兴趣等信息，构建社交网络表，推测该用户的社交习惯。
- 搜索：记录用户在系统中的搜索关键词和搜索结果等信息，以了解用户的搜索需求和偏好。

1.3 存储系统的可拓展性

由于微博用户数量和数据量非常庞大，因此，系统需要实现分布式存储和弹性扩容能力。通过数据分区，将数据分为多个分区，将每个分区存储在不同的节点或服务器上，并通过负载均衡将请求分发到不同的节点上，可以使得微博存储系统实现更好的横向扩展性。

此外，微博存储系统需要支持实时访问和高并发处理，因此应该使用分布式数据库架构，比如采用MySQL Cluster、Apache Cassandra等分布式数据库解决方案，来提高系统的读写扩展性和处理能力，以满足系统高并发和实时数据处理的需求。

最后，在存储系统的架构设计过程中，还需要考虑到数据安全和数据备份等问题。对于重要数据需要进行备份，避免数据丢失，同时采用权限系统和数据加密等安全措施，提高系统的数据安全性和可靠性。

2. 数据模型设计

2.1 用户信息数据模型

- User表：主要存储用户信息，包括用户ID、用户名、密码、邮箱、地域信息等字段。其中，用户ID是主键，保证唯一性和快速查找。
- Follow表：主要用于记录用户之间的关注关系，包括关注者的用户ID和被关注者的用户ID两个字段。同时，要对这个表建立索引，提高访问效率。
- Fans表：主要记录用户粉丝关系，类似Follow表，在关注中被关注者角色反转。
- Block表：主要存储用户拉黑信息，包括哪些用户被当前用户拉黑。Block表的id是主键，要做到防重复设计，防止重复拉黑同一用户。
- FriendRequest表：用于记录发送好友请求的用户ID、被请求者的用户ID、发送的时间戳和申请是否被接受的状态。
- Message表：主要存储用户私信等信息，包括发送者ID、接收者ID、内容、时间等字段。

2.2 微博内容数据模型

关于微博内容数据模型中结构化数据的设计思路：

- Weibo表：主要存储微博**基础信息**，包括微博ID、微博内容、发布时间、发布者ID、访问量、点赞数、转发数等字段。其中，微博ID应该是主键，保证唯一性和快速查找，发布者ID和发布时间等也应该建立索引，提高查询效率。
- WeiboComment表：主要存储微博**评论信息**，包括评论微博的ID、评论内容、评论时间、评论者ID、原微博ID等字段。其中，评论ID应该是主键，微博ID应该建立索引，方便查询特定微博的评论信息。
- WeiboLike表：主要存储微博**点赞信息**，包括点赞ID、点赞者ID、微博ID、点赞时间等字段。其中，点赞ID应该是主键，微博ID和点赞者ID都应该建立索引，方便查询特定微博的点赞者或者特定点赞者点赞过的所有微博。
- WeiboRetweet表：主要存储微博**转发信息**，包括转发ID、转发者ID、被转发者ID、微博ID、转发内容、转发时间等字段。其中，转发ID应该是主键，微博ID和转发者ID都应该建立索引，方便查询特定微博的转发者或者特定转发者转发过的所有微博。

关于微博内容数据模型中非结构化数据的设计思路：

- Media对象：用于存储微博中的非结构化数据，如图片、视频、音频等，每个Media对象应该包括媒体文件的ID、类型、URL、上传时间、**存储路径**、所属微博的ID等信息。微博ID为外键，**与Weibo表关联**。通过Media对象的存储，可以实现微博内容的多媒体展示，提高微博信息的信息量和可读性。
- WeiboContentIndex对象：用于存储微博内容的索引信息，包括微博ID、发布时间、所属用户ID、微博的主题标签等。在微博数据量非常大的情况下，采用索引对象来对微博内容进行分类和检索，可以大大提高检索效率和性能。

3. 存储媒质的选择

3.1 考虑要素

从成本而言，考虑到大量微博数据的存储成本主要为存储系统开发的成本和硬件成本。在满足存储性能的要求下，需设计较为廉价的方案，在存储系统开发成本和硬件成本之间达成平衡。

从存储容量而言，考虑到微博每天都会产生海量的数据，而且每个用户还会时差去翻阅查询许久之前的微博内容，需要使用大容量的存储媒质，并且要具有不易丢失数据和永久保存的特点。

3.2 选择方案

综上，本系统中采用磁盘存储数据，磁盘存储的特点是不易丢失数据，可以永久保存，而且写入和访问能力主要取决于磁盘的读写能力，可以控制成本。磁盘主要分为HDD盘、SSD盘，也就是机械盘和固态盘，两者的QPS都在千级别，可以满足微博的大数据存取需求。

SSD盘相比于HDD盘拥有更快的读写速度、更高的QPS，能够更快地响应读写请求，大幅提高存储性能和系统效率。但SSD盘的造价远高于HDD盘，容量和寿命也明显受限制，需要更加注意其容量适用范围和维护成本。

接下来我们考虑热数据、温数据和冷数据，分别挑选不一样的存储媒质。

- 热数据：

  用户信息数据部分：用户的基本信息、粉丝表、关注表会被经常访问，该表可以作为热数据，放在SSD盘中。

  微博数据部分：考虑发布微博的访问和时间关系紧密，可以将近三天的微博信息储存在SSD盘中，以便于快速访问。此外，将一周内微博数据的热点部分（即访问频率高的部分）也缓存在SSD盘中。

- 温数据：

  用户信息数据部分：用户的黑名单等表格访问的几率较小，可以使用HHD盘存储。

  微博数据部分：近一月的微博，都有一定的访问几率，将这一部分使用HHD盘存储。

- 冷数据：

  微博数据部分：由于微博需要对多年前的内容也能够进行访问，但是用户往往可以对多年前的微博访问进行一定的等待，因此，将这一部分的微博数据放于冷存储硬盘HotsCoin中。

4. 网络存储架构设计

网络存储架构是指存储系统中不同存储单元之间的连接方式和架构组成。在一个存储系统中，不同存储单元（例如硬盘、服务器等）之间需要通过网络进行连接和通信，才能构成一个完整的存储系统。网络存储架构设计即决定了这些存储单元之间的接口及拓扑结构等细节配置，以满足存储系统的功能需求。针对海量微博数据存储系统的网络存储架构设计主要分为数据服务、网络拓扑、路由协议、数据存储、传输安全五个部分。

其逻辑为：

- 数据服务：数据服务作为整个系统的核心部分，在开始设计时明确，这是后续网络存储架构设计各个方面的基础。
- 网络拓扑：在数据服务的基础上，需要设计合理的网络拓扑结构，以确保数据的高效传输和可靠存储。
- 路由协议：路由协议是指数据在网络拓扑中的传输路径，需要根据实际情况选择合适的路由协议，并制定相应的路由策略。
- 数据存储：数据存储是海量微博存储系统的重要组成部分，需要根据实际需求选择合适的存储设备和技术，并制定相应的存储策略。
- 传输安全：数据在网络传输过程中需要保证通信的安全性，需要采用合适的加密和认证技术，以及安全的传输协议，确保数据不被非法获取或篡改

4.1 数据服务

微博有着上亿用户的使用，进行着不同的行为活动，比如说关注、点赞、发微博、评论、转发等等服务。考虑到业务的多样性和存储的分散性和巨量性，因此，需要使用分布式计算框架来实现海量数据的处理和计算。

在分布式计算中，我们选取了MapReduce，它可以将大规模数据集划分成小的数据块，然后并行处理每个数据块，最后将结果合并起来得到最终结果。其优点主要体现在以下几个方面：

- 分布式计算：可以支持分布式计算，在分布式环境下可以方便地扩展计算资源，从而实现高性能计算。
- 高可靠性：任务执行过程中可以进行容错处理，包括任务失败恢复、数据备份和容灾等机制，保证计算的可靠性。
- 适用范围广：简单易用、容易扩展和适用于多种计算模型等优势，所以它可以应用于多种计算场景下，并且能够帮助用户更好地处理和分析海量数据。

4.2 网络拓扑

正如数据服务的分析所言，微博用户地域位置分布广泛，因此要对微博的相关数据和存储阵列进行分布式部署。而在网络拓扑模块，我们决定在服务器和存储阵列之间主要采用 **IP SAN** 进行连接，这主要是基于如下的考虑：

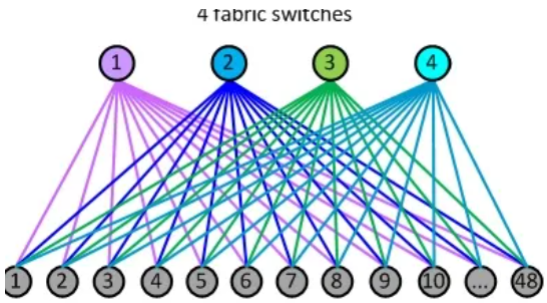
- 利用无所不在的IP网络，解决了微博用户群体大的问题
- IP存储超越了地理距离的限制。IP能延伸到多远，存储就能延伸到多远，十分适合于对微博海量数据的远程备份。
- IP网络技术成熟。IP存储减少了海量微博数据配置、维护、管理的复杂度。

此外，考虑到每天都会产生的海量数据、高并发、高IO的需求，我们打算采用 Clos 网络拓扑架构作为整个数据存储的基础。这主要是基于如下的考虑

- 易于管理维护：Clos网络结构具有模块化的架构，每个节点之间互相独立，故障不会影响整个系统。这样可以保证微博数据存储系统的可靠性和稳定性，并且容易进行管理和维护。
- 无阻塞交换：Clos网络结构的每个级别之间都是全连接的，避免了瓶颈和拥塞，从而降低了阻塞。这意味着即使在高负载情况下，数据仍然可以快速稳定地传输，从而进一步提高微博数据存储系统的性能和可靠性。
- 高带宽、低时延：Clos网络结构中每个节点都与其他节点直接连接，通过多层交叉连接实现了高带宽、低时延的数据传输。这对于微博数据存储系统来说，尤其重要，使得我们分布式架构中的mapper节点可以快速地将数据转发给reducer节点

如图所示是最简单的Clos网络架构。

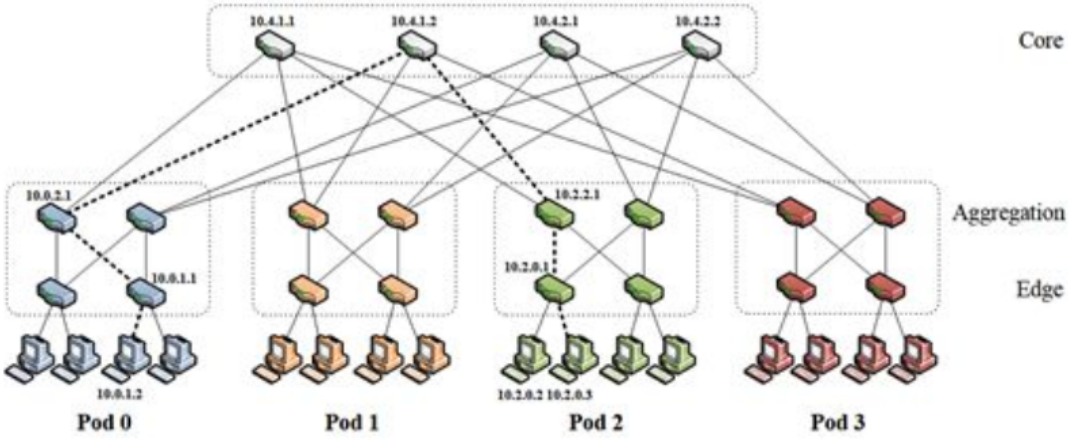
Clos网络结构可以通过增加Leaf节点和Spine节点的数量实现扩展，这种扩展方式非常灵活，并且可以在不影响整个系统的情况下快速完成。



从连接一致性来看，每台服务器通常与任意其他服务器之间都有三个网络跃点。我们可以看到连接矩阵非常丰富，因而可以有效地处理故障问题。单一故障乃至多个链接故障，都不会导致两个服务器之间完全的失去连接，而只会导致带宽的部分损失，易于**管理维护**。

而在这种**非阻塞的架构**中，Leaf节点和Spine节点之间的带宽容量应该与Leaf节点和服务器之间的带宽容量一样大，可以连接的服务器的总数是$\frac{n^2}{2}$，为交换机的端口数量。

还可以在二层 Clos 网络上进一步扩展，把每个二层 Clos 网络通过新增一层 Spine 交换机连接。此时每个二层 Clos 网络成为一个 cluster。在这样的网络中，假设在每一层都使用相同的交换机，那么可以连接的服务器总数就提升到了$\frac{3n^3}{4}$。



我们可以发现Clos网络具有足够的水平带宽能力，较低的CPU开销，更高的路由容量，同时方便在每个设备和链路基础上对路由策略进行更加精细的控制。

4.3 负载均衡和路由选择

将数据划分为多个分片，并将每个分片存储在不同的存储节点上。通过采用哈希函数或范围分片的方式，将数据均匀地分布到不同的节点上，以实现负载均衡。当有新的数据写入或查询时，可以通过哈希函数计算出相应的分片，并将请求发送到对应的节点上。

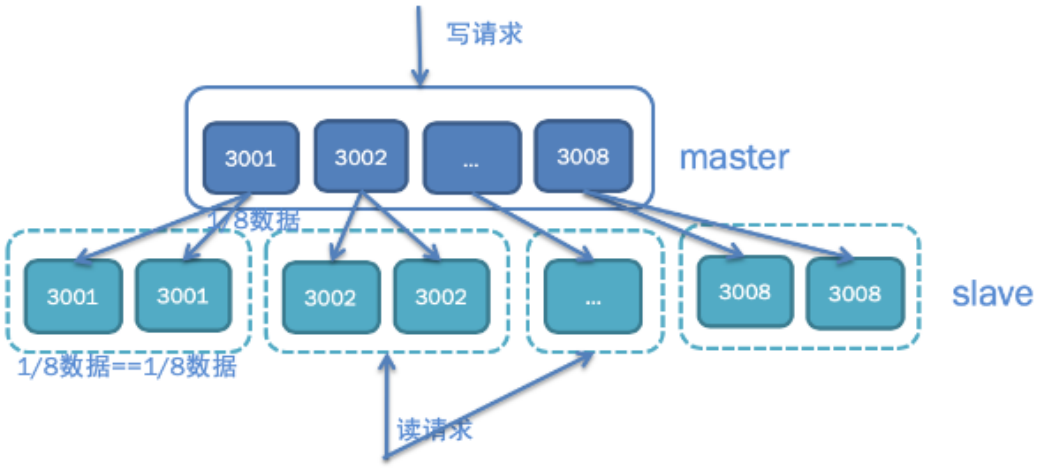
在微博数据存储系统的设计中，EBGP可以用于路由选择和传播。它具有以下优点：

- 路由灵活性：EBGP支持自治系统之间的路由信息交换，使得不同自治系统之间可以进行路由选择和传播。这对于微博数据存储系统来说，可能涉及到多个自治系统之间的数据传输和访问，使用EBGP可以实现灵活的路由控制和路由策略的定义。
- 路由策略控制：EBGP允许管理员定义和控制路由策略，包括路径选择、策略过滤和路由传播等。这对于微博数据存储系统中的负载均衡、容灾备份和故障恢复等方面非常有用，可以根据需要灵活地配置和调整路由路径。
- 可扩展性：EBGP适用于大规模网络，能够处理大量的路由信息，并提供高度的可扩展性。在微博数据存储系统中，可能需要处理大量的数据流量和路由信息，使用EBGP可以满足系统的扩展性和性能要求。

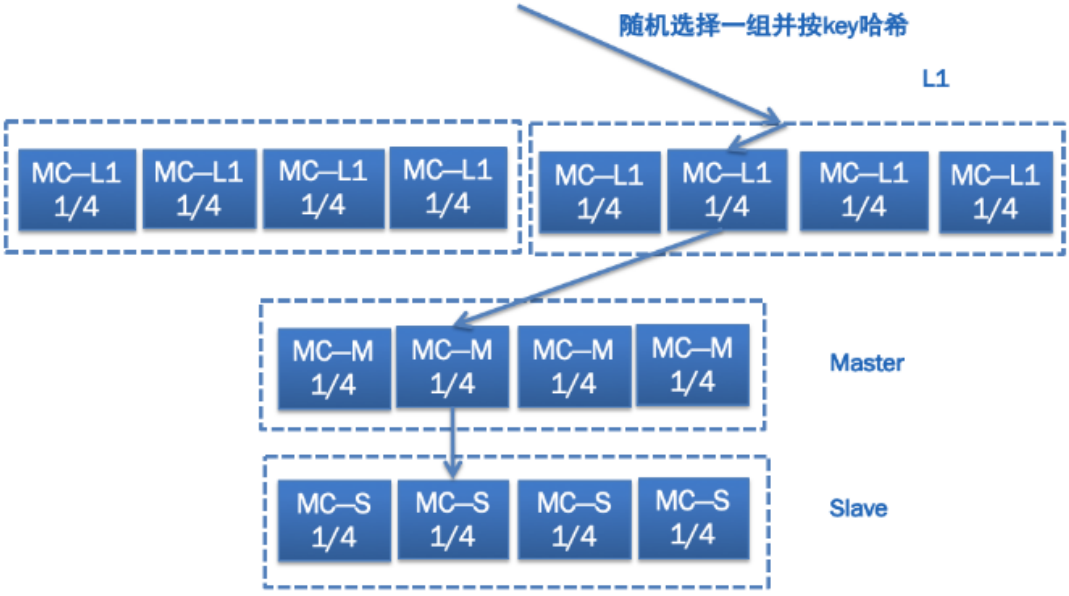
4.4 数据存储

微博数据存储使用两层架构：MySQL数据库和Memcached缓存，此外还使用了redis。

部署的Memcached 缓存，减少了Mysql 99% 的访问压力，只有 1% 的请求会访问数据库。然而对于微博业务来说，这 1% 的请求也有几万 QPS，对于单机只能扛几千 QPS 的 MySQL 数据库来说还是太大。为此需要对数据库端口进行拆分，如下面的示意图，每个用户的 UID 是唯一的，不同 UID 的用户按照一定的 Hash 规则访问不同的端口，这样的话单个数据库端口的访问量就会变成原来的 1/8。除此之外，考虑到微博的读请求量要远大于写请求量，所以有必要对数据库的读写请求进行分离，写请求访问 Master，读请求访问 Slave，这样的话 Master 只需要一套，Slave 根据访问量的需要可以有多套，也就是“一主多从”的架构。



Memcached采用多级结构，即 L1-Master-Slave，它们的作用各不相同。L1 主要起到分担缓存带宽压力的作用，并且如果有需要可以无限进行横向扩展，任何一次数据请求，都随机请求其中一组 L1 缓存，这样的话，假如一共 10 组 L1，数据请求量是 200 万 QPS，那么每一组 L1 缓存的请求量就是 1/10，也就是 20 万 QPS；同时每一组缓存又包含了 4 台机器，按照用户 UID 进行 Hash，每一台机器只存储其中一部分数据，这样的话每一台机器的访问量就只有 1/4 了。Master 主要起到防止访问穿透到数据库的作用，所以一般内存大小要比 L1 大得多，以存储尽可能多的数据。当 L1 缓存没有命中时，不能直接穿透到数据库，而是先访问 Master。Slave 主要起到高可用的目的，以防止 Master 的缓存宕机时，从 L1 穿透访问的数据直接请求数据库，起到“兜底”的作用。



微博的存储除了大量使用MySQL和Memcached以外，还有一种存储也被广泛使用，那就是Redis。，比如微博的转发、评论、赞的计数便可以使用redis来进行存储，此外可以进行冷热处理，热数据放到内存里，冷数据放到磁盘上，并使用LRU，如果冷数据重新变热，就重新放到内存中。

4.5 传输安全

在设计时，我们同时也要注意安全问题。

Clos网络的安全问题主要是物理安全和数据隐私问题。在Clos网络中，节点之间需要进行大量数据交换，网络安全性直接影响到数据的安全性。而物理安全问题则包括节点被未授权用户访问、破坏或损坏等威胁。

eBGP的安全问题主要是源认证和路由欺骗。由于eBGP需要外部节点之间的物理连接和相互信任，因此如果有攻击者伪造源地址，那么攻击者可以向外部节点发送虚假的路由信息，导致网络出现安全漏洞。此外，攻击者还可以进行源地址欺骗，以获得非法的网络访问权限或进行流量劫持等恶意行为。这些安全问题可能会给网络带来严重的影响，如网络瘫痪，数据泄露等。

我们的系统设计中，也会使用如下的手段

- 认证和加密：对于Clos网络，主要是采用基于证书的身份认证机制和SSL/TLS协议等加密通信方式，保证节点与节点之间的通信和数据传输过程中的安全性。针对eBGP的攻击和网络拓扑变化，采用源地址认证机制和利用密钥对路由信息进行加密，确保在路由信息传输过程中没有被篡改或窃听。
- 访问控制：访问控制主要针对网络内部和外部的威胁，可以通过配置ACL、开启端口保护、ARP欺骗防范和MAC限制等，确保只有经过授权的节点才可以访问网络。此外，还可以利用VRF技术隔离路由信息，从而避免非法网络流量的插入。
- 防火墙：在Clos网络中，防火墙可以通过VLAN隔离和访问控制等技术，确保只有授权用户才能访问网络并保护数据密度度。针对eBGP网络，防火墙可以在BGP路由和通信核心节点上设置，通过过滤联网节点的BGP消息和数据流量，避免来自其他节点的非法连接和攻击。
- 监控和审计：在Clos网络中，可以采用弱点扫描和网络威胁评估等技术，进一步提高网络安全性。另外，我们会建立网络监控和审计机制，检测和跟踪所有网络流量和BGP路由跳数，识别潜在的安全漏洞和网络漏洞，并采取相应的应对措施，避免网络失效或数据泄露。此外，

5. RAID级别的选择及解释

此次微博数据存储系统的RAID级别的选择为RAID10，原因有以下几点：

- 读写性能：RAID 10具有优秀的读写性能。由于数据被分布在多个磁盘上，并且可以同时从多个磁盘读取或写入，因此RAID 10能够提供更高的并发性和吞吐量。相比之下，其他RAID级别如RAID 5在写入操作时需要进行奇偶校验计算，可能会对性能产生一定的影响。而微博单日发微博数一个亿左右，而且由于微博的用户群体庞大，浏览微博的次数也十分频繁，因此需要十分优秀的读写性能，RAID10相对符合条件
- 冗余性和容错能力：RAID 10提供了非常高的冗余性，能够容忍多个磁盘故障。因为数据被镜像在多个磁盘上，如果一个磁盘故障，系统可以从镜像副本中快速恢复数据。这种级别的冗余性使得RAID 10在面对故障时具有更好的数据保护能力。
- 容错恢复速度：当发生磁盘故障时，RAID 10的恢复速度比其他RAID级别更快。因为数据被镜像在其他磁盘上，系统可以直接从镜像副本中复制数据到新的磁盘，而无需执行奇偶校验计算或重建整个条带。这导致RAID 10在故障发生后的恢复时间较短，减少了对系统性能的影响。
- 灵活性：RAID 10在扩展和管理方面较为灵活。可以通过添加更多的镜像对来扩展存储容量和性能。此外，RAID 10也比较容易进行维护和管理，因为它不涉及奇偶校验和条带重建等复杂操作。

RAID10的主要缺点是相对较高的成本，因为它需要使用较多的磁盘来实现镜像和条带化。然而，对于对数据可靠性和性能要求较高的微博，这种冗余性和性能的权衡通常是合理的。综上所述，RAID10是一个在微博数据存储系统设计中值得考虑的RAID级别，尤其是在对数据可靠性、性能和容错恢复速度有较高要求的场景中。

6. 数据清洗和处理

6.1 数据去重

微博数据中可能存在重复的内容，如重复的微博消息或重复的用户信息。数据清洗可以识别并去除重复的数据，减少存储空间和提高数据查询效率。

6.2 数据格式化

微博数据可能包含不同的语言、缩写、拼写错误等。通过数据清洗，可以对数据进行标准化处理，提高数据的可读性和可搜索性。

6.3 数据清洗

- 噪声和异常数据处理：识别并处理噪声数据、异常值或不一致的数据。例如，去除错误的字符、处理缺失值或异常数据。
- 停用词过滤：识别并过滤掉无关或常见的停用词，例如虚词、标点符号等。
- 敏感信息处理：识别和保护用户敏感信息，例如对手机号码、身份证号码等进行脱敏处理。

6.4 数据验证和修复

- 数据完整性验证：验证数据的完整性，例如检查必要字段是否存在、数据是否缺失或错误。
- 数据纠正和修复：根据规则或模型对数据进行纠正和修复，例如纠正拼写错误、填补缺失值等。

7. 备份策略设计

7.1 数据分级

按照 80/20 原则，数据的重要性是有差异的。如果不加区分的恢复 100PB 级别的数据，无论是从成本上还是效率上都是无法承受的。微博从垂直与水平两个方向对数据的优先级进行拆分：

- 垂直层面：按业务的重要程度划分核心与非核心。核心业务的确认相对复杂，通常需要公司层面的领导拍板，但核心业务变更的频率非常低。微博上千个对外提供的 API，核心 API 只有十几个。
- 水平层面：数据的访问是有时效性的，在微博场景下表现更加明显，7 天内的数据访问量超过 98%。部分超过 1 年的数据被访问的吞吐基本维持在个位数甚至是零，简单的使用吞吐量作为数据的访问热力值，通过热力值对数据进行二次分级。

7.2 备份频率与类别选择

一共有如下三种备份方式：

- 完全备份：一次对数据进行完整的备份。当发生数据丢失的灾难情况时，完全备份无需依赖其他信息，即可实现100%数据恢复，其恢复时间最短且操作最方便；
- 增量备份：只有那些在上次完全备份或者增量备份后被修改了的文件才会被备份。优点是备份数据量小，需要的时间短，缺点是恢复的时候需要依赖之前的备份记录，出问题的风险较大；
- 差异备份：备份那些自从上次完全备份之后被修改过的文件。因此从差异备份中恢复数据的时间较短，因为只需要两份数据——最后一次完全备份和最后一次差异备份，缺点是每次备份需要的时间较长。

由于微博的庞大的数据，所以全部采用完全备份对于备份的空间和时间都是巨大的考验，难以实现，而微博自身的定位，又要求其发生故障时可以迅速地恢复数据。因此我们总体上采用完全备份与差异备份相结合的方式，保证数据的安全完整和恢复的高速。

根据数据分级的情况，对于热点的比较高的数据和核心业务数据的备份频率会较高，一般以天为单位进行一次完全备份，小时为单位进行差异备份，确保数据的安全。而随着热点下降和业务重要性下降，可逐步减少备份的频率：以周为单位进行完全备份，天为单位进行差异备份。

除此之外，在某些重大事件发生时,(具有很高的讨论度),微博数据的更新频率可能大大增加，而故障的可能性也会变大，应当提高备份频率，以小时为单位进行数据的备份。

7.3 备份标准化与自动化

所有接入数据恢复中心的服务，都需要提供相关API：用于生成数据的快照和提供自上一次快照以来产生的所有的 操作。服务提供 API 后，数据恢复中心就能自动备份数据，实现备份与业务的解耦合。相关业务只要上线，即可纳入数据恢复中心进行备份。

7.4 321备份

低级别的日志类数据一般采取单机离线冷备，重要数据则采用多副本热备，而影响公司命脉的核心数据通常采用 321 备份策略，即：

- 至少 3 个副本
- 2 个不同的存储介质
- 1 个 offsite

微博在数据备份上遵循 321 策略：

- 所有的数据备份都至少包含 2 个热备，2 个冷备；
- 在线热备数据存储在 SSD 设备，冷备数据存储在独立的 OSS 集群；
- 数据会在异地离线存储，通过专线进行数据同步与传输。

7.5 灾难恢复计划

建立灾难恢复计划，明确恢复数据的优先级和步骤。确定关键数据和系统的恢复时间目标（RTO）和恢复点目标（RPO），以确保数据能够在规定的时间内快速恢复。

因为微博其自身的高社交性和高流量，每分每秒都是极其重要的，需要在2个小时内恢复微薄的核心数据：用户数据，热门数据，核心业务数据等等，在24小时之内恢复微博所有数据，完成整个系统的恢复，并记录日志。