论数据分片技术及其应用

2019 年 2 月，我所在的公司成功中标某省生态环境厅的信息化综合服务平台（包括污染源在线监控系统，空气质量指数（AQI）实时自动发布系统，重点排污单位监控系统，危险废物处置系统，网上审批系统等）建设项目，该平台主要解决污染企业、产废企业、处置企业业务申报和环境执法部门对相关企业的在线监督、业务审批和管理。我作为项目的架构师和技术负责人，主要完成了需求获取分析、系统架构分析和设计等工作。本文以此项目为基础，首先介绍了常用到的数据库分片技术，然后介绍了我们这个项目采用数据分片技术（一致性 Hash 分片和按数据范围分片相结合）以及 Ceph 技术架构及实施。经过全体团队成员一年的努力，平台开发完成并顺利上线，通过近一年的用户反馈，整个系统运行稳定，达到了预期的目标和要求。

目前，环境保护是每一个省市地方政府和人民群众都十分关心的问题，国家也对各级地方政府在环保方面提出了更高的要求，为了满足人民群众需求、方便企业办事，省生态环境厅决定开发生态环境厅信息化综合服务平台，平台包括多个系统，如空气质量指数（AQI 实时自动发布系统，我们通过手机 APP 等终端就可以实时了解空气质量;网上审批系统帮助企业方便快捷的办理各项业务申报;重点排污单位监控系统帮助环境监督和执法部门对各排污企业进行实时监督和管理，防止企业偷排偷倒。由于平台涉及自动发布和实时监控系统，每分每秒都会产生大量数据进行存储，而且平台涉及全省 11 个地级市上万家重点排污企业的实时监控，所以系统的可靠性、稳定性、及时响应成为需要高度关注的问题。经过讨论研究，底层的数据存储我们决定采用数据分片技术，将系统中的数据分布存储在不同的节点，提高系统运行的可靠性，提升系统数据处理速度，增强系统的可扩展性。下面结合该项目对数据分片技术进行介绍。

数据分片就是按照一定的规则，将数据集划分成相互独立正交的数据子集。然后将数据子集分布到不同的节点上。常见的数据分片算法包括 Hash 分片、一致性 Hash 分片和按照范围数据分片三种。

1.Hash 分片：哈希表（散列表）是最为常见的数据结构，根据记录（或者对象）的关键值将记录映射到表中的一个槽（slot），便于快速访问。最为简单的散列函数是 mod N（N 为表的大小）。即将 hash 值对 N 取余，余数即在表中的位置。hash 分片的道理是一样的，首先按照数据的某一特征（key）来计算哈希值，并将哈希值与系统中的节点建立映射关系，从而将哈希值不同的数据分布到不同的节点上。比如系统中存在 3 个存储结点（Node0、Node1、Node2），对 Hash 值模 6，得出的结果就是要存入缓存节点的序号。hash 分片映射关系简单，需要管理的元数据也非常少，只需要记录节点的数目以及 hash 方式就行了。但 hash 方式的缺点也非常明显：当增加或者减少存储节点时，之前建立的映射关系会失效。

2.一致性 Hash 分片：用一致性 Hash 算法可以很好地解决增加和删减节点时，映射关系失效的问题。一致性 Hash 算法将整个 Hash 值空间组织成一个虚拟的圆环，然后将存储节点的 IP 地址或者主机名做 Hash 取值后，放置在这个圆环上。当我们需要确定某一个 Key 需要存取到哪个节点上的时候，先对这个 Key 做同样的 Hash 取值，确定在环上的位置，然后按照顺时针方向在环上"行走"，遇到的第一个缓存节点就是要访问的节点。一致性 Hash 算法解决了增加删除节点时大量映射关系的问题，但是，一致性 hash 方式在增加节点的时候，只能分摊一个已存在节点的压力;同样，当某个节点故障时，该节点承担的所有访问都会被顺移到另一个节点上，会对后面这个节点造成很大压力。我们希望在增删节点的时候，已存在的所有节点都能参与响应，达到新的均衡状态。因此，在实际工程中，引入虚拟节点的概念。即不是将物理节点映射在 hash 环上，而是将虚拟节点映射到 hash 环上。虚拟节点

的数目远大于物理节点，因此一个物理节点需要负责多个虚拟节点的真实存储。操作数据的时候，先通过 hash 环找到对应的虚拟节点，再通过虚拟节点与物理节点的映射关系找到对应的物理节点。

3.按照范围数据分片。就是按照时间区间或 ID 区间来切分。比如按年份将数据分散存储到不同的库（表）中;因为有些系统使用的数据可能存在着冷热不均的特点，比如最近一年的数据使用频率高，之前年份的数据使用率低，将使用较少的历史数据迁移到其他库中，提高查询的效率。我们结合数据分析技术特点和用户需求，采用了一致性 Hash 分片和按照范围数据分片相结合的方式。由于污染源在线监控系统，空气质量指数（AQI）实时自动发布系统产生的数据量大，时间性强，我们采用按季度进行数据分片，每个季度的数据存储在一个表中。对于重点排污单位监控系统等采用一致性 Hash 分片技术，采用 Ceph 集群部署了 6 个 OSD 节点和 3 个 MON 节点，每个 OSD 节点配备 2 块 1TB SSD 磁盘，12 块 8TB SATA 磁盘，SSD 磁盘和 SATA 分别用于 Journal 日志和数据存储，因为 Ceph 使用日志来提高性能及保证数据一致性。使用 SSD 作为 OSD 的日志盘来提高集群性能，OSD 存储服务主要功能是存储数据、平衡数据、恢复数据以及与其它 oSD 间进行心跳检查等。每一个 OSD 进程都可称作是一个 OSD 节点，每个 OsD 节点监听不同的端口，每个 OSD 节点可以设置一个目录作为实际存储区域，一般对应一整块硬盘。MON 监控服务主要负责监控整个集群，每个 Ceph 集群中至少要有一个 MON 节点，维护集群的健康状态，维护展示集群状态的各种图表，如 OSD Map、Monitor Map、PG Map 和 CRUSH Map。采用一主两从的架构，总计提供的存储容量大约为 290TB。每个 OSD 节点使用 4 条万兆网络，每 2 条绑定，分别连接外部公共网络和内部网络。Ceph 采用 CRUSH 算法，数据存储到 Ceph 集群时，首先将数据先被分割成多个 4MB 的 Object（大小可以设置），每个对象对应有 Object id。因为 Object 的 size 很小，在一个大规模的集群中可能有几千万个对象，这样导致遍历寻址速度会非常缓慢的，并且如果将对象直接通过某种固定映射的哈希算法映射到 OsD 上，当某个 OSD 损坏时，对象无法自动迁移至其他 OSD 上面。为了解决这些问题，Ceph 引入了归置组的概念，即 PG。PG 类似于数据库中的索引，每个对象都会固定映射进一个 PG 中，所以当我们要寻找一个对象时，只需要先找到对象所属的 PG，然后遍历这个 PG 就可以了，无需遍历所有对象，提高了遍历的速度。而且在数据迁移时，也是以 PG 作为基本单位进行迁移，Ceph 不会直接操作对象。PG 再通过 CRUSH 计算，映射到 OSD 中。如果是三副本的，则每个 PG 都会映射到三个 OSD，保证了数据的冗余。

2020 年 1 月，整个项目顺利完成，通过近一年的用户使用，整个系统运行稳定，达到了预期的目标和要求，受到了用户的好评。但也存在些不足，比如在数据同步的时候，偶尔出现部分数据同步失败的提示，后来我们及时更新数据同步程序，解决了这个问题。这是以后在项目中需要注意的地方。但总的来说，项目是成功的，我也通过这个项目学习到了不少经验。今后我将继续加强学习，总结经验教训，提高自己的能力。