摘要:

2017年7月，我参加了某互联网公司的网约车出行平台的开发，系统架构师设计师一职，该平台主要是将司机乘客资源匹配解决用户出行问题，提供了专车、快车、出租车等多种出行方式。本文结合作者实践，以出行平台为例，讨论NoSQL数据库技术以及其应用，NoSQL数据库泛指非关系型数据库。按照数据模型可以分为4类：键值（Key-Value）存储数据库、列存储数据库、文档存型数据库、图（Graph）数据库。主要是为了弥补，关系型数据库：性能低、容量有限、和运维成不低的缺点。在平台中我们选择键值数据库，来做内容缓存。具体在用户登录注册、司机位置上报、订单执行过程等场景增加键值数据库同步，降低业务数据库的读写压力，提升平台性能。使平台达上线预期性能目标，稳定运行的一年多时间。无较大生产故障得到用户的一致好评。是对我工作的认可和肯定。

正文:

2017年7月，我参加了某互联网公司的网约车出行平台的开发，该平台主要是为了改善目前：黑车乱象、乘客出行安全无法得到保障背景下。为了构建一个安全、规范的网约车环境。解决用户打车难、提升用户的出行幸福感，拉动了产业链的发展，构建了一个完整的出行生态链的目标。而构建的一个网约车出行平台。平台包括乘客端、司机端、后台管理系统三部分组成。乘客端供乘客查询车辆、发布订单、支付车费、评论司机；司机端供司机车辆信息认证、出车接单、乘客接送、车费提现等；管理系统主要是提供系统报表查询、规则配置、乘客管理、司机管理、分公司管理、账务管理等。本平台提供了专车、快车、出租车业务，乘客可以通过自身的需求来选择不同类型业务出行，司机需要上传自身拥有的运营车辆等证照信息到平台审核，只有当平台审核通过后才能正常的再平台上合法运营。在该项目中本人担任系统架构师设计师一职，负责项目的架构设计以及软件开发的部分工作。

在NoSQL的技术选型的过程中我们分析了常见的4种数据库，1、键值（Key-Value）存储数据库，使用场景在内容缓存，主要用于处理大量数据库的高访问负载也用于一些日志系统等等。数据模型包括：Key指向Value的键值对，通常通过hash table来实现。优点查询速度快。缺点数据无结构化，通常只被当做字符串或二进制数据，如：Redis、Oracle BDB。2、列存储数据库，使用场景在分布式文件系统，以列簇形式存储，将同一列数据存储在一起。查找速度快、可拓展性强，更容易分布式拓展。如：HBase, Cassandra。3、文档型数据库。使用场景在Web应用中。它是一种键值存储相类似，该类型的数据模型是版本化的文档，半结构化的文档以特定的格式存储。比如JSON。如：MongoDb, CouchDB。4、图形数据库（Graph）使用场景在社交网络、推荐系统等专注构件关系的图谱。采用图结构，利用图解狗相关算法，比如最短路径寻址，N度关系查找等。如：Neo4J, InfoGrid。在本平台中要是通过内容缓存来提升平台性能，所以我们选择了键值数据库Redis数据库。

在平台中使用键值数据库可以降低程序的复杂度，提高了核心系统的吞吐量，高水平拓展和低端硬件集群，避免了昂贵的对象-关系映射。键值数据库Redis采用内存数据结构存储，作用数据库的，缓存和消息代理，它支持数据库如: 字符串、散列、列表，集合，带有范围查询、地理空间位置索引，可以设置不同级别的数据持久化，支持自动分区等特征。支持master-slave集群方式。键值数据库采用hash table对比关系型数据库索引采用B+树的方式来实现，查询性能更高。并且键值数据库数据库Redis数据库也是内存数据库，它抛弃了磁盘数据的管理方式，基于全部数据都在内存中重新设计了结构体系，并且在数据缓存、快速算法，并行操作，方面做了选哪个赢得改进，所以处理速度要比传统数据库的数据处理速度快很多。

通过键值数据库做内容缓存来降低程序的复杂度。如我们在用户中心。司机的信息分散在各个数据表中，在执行一次查询指定司机信息查询的时候需要在业务数据库先查询司机的基本信息，再去关联司机的车辆信息，最后关联司机的订单信息，这个查询的关联过程会显得非常的复杂，通过做通过冗余的方式也不好设计，基于这样的情况我们可以在第一次创建司机信息的时候同步构建一个司机对象到键值数据库中，包含：基本信息、车辆信息、订单信息存储到同一个键值数据的Key中做一个清晰的定义。可以使得查询的时候可以直接通过司机的手机号直接查询司机的所有信息，也不改变已有的业务数据库设计，又可以减少了业务系统对业务数据库的依赖，由此实我们的降低架构的复杂度的目标。

通过键值数据库做内容缓存来提升吞吐量。对于业务复杂，高并发和大流量的系统，内容缓存是重要的组成部分，提升性能的主要方式之一就是内容缓存，它可以挡掉大部分的数据库访问冲击，防止业务系统崩溃。平台中我们将用户信息、订单信息等。在同步存储业务数据的同时同步存储键到值数据库做内容缓存，可以在相关的业务节点快速的获取这些信息。比如网关可以直接访问键值数据库来获取用户登录状态以及访问权限来来抵挡后端服务的流量冲击，防止爬虫，机器人等恶意攻击。还有在位置中心需要将位置信息做实时上报，由于业务数据库的资源非常宝贵，我们将位置司机车辆实时位置信息存储到键值数据库中，能够做到高频的读写查询，据统计比存储到关系型数据库中速度要要快10倍以上。

通过键值数据库节约服务器的运维成本。如我们在以前需要高配置的小型机来支持业务运行，如果遇到拓容，有需要去采购更加高规格、高配置得服务器来解决，由于这个环节涉及到资金问题，且对于替换下来的服务器资源也只能闲置，使得每次都需要走很长的审批流程使得拓容的周期拉得非常的长。并且平台的运维和管理成本非常的高。现在采用键值数据库服务器的它本身具有高水平拓展和低端硬件集群的特征。我们可以使用之前淘汰、闲置的服务器来做键值数据库服务器集群，这样避免了资源的浪费，节约了公司的运维成本，而且键值数据库的安装维护也极其简单，得到了运维部门的极大支持，也得到了上级领导的认可和支持。

以上的架构设计使得平台达到了预期性能目标，发挥了NoSQL数据库的避免不必要的复杂性、吞吐量、高水平拓展能力和低端硬件集群，避免高贵的对象-关系映射等优势。首先这些设计，对于平台的整体提升的性能非常大。使得平台更加的灵活，更容易拓展，更加高可用，提升查询效率，提升了平台承载体量，提升了公司核心产品竞争力。其次为公司节约了硬件资源和运维成本，为公司节约了运营成本。在过程中我们做了几次相关的技术分享，总结了在开发过程中遇到的问题和分享了一些优秀的解决方案。从而提升了开发人员的技术水平，同时也提升了团队的凝聚力。每个开发成员应对分布式系统的设计有了新的认识。

当然，NoSQL自身也有数据模型和查询语言没有经过数学验证，不支持ACID特征，功能简单，没有统一的查询模型等缺点。再项目中我们需要开发构建去维护业务数据库的数据一致性。所以我们应该继续优化NoSQL的使用设计，做到更进一步，争取能够设计出更多高质量，高性能的软件产品。