摘要:

2017年7月，我参与某互联公司自主研发全国运营的网约车出行平台。该平台主要是为了解决客户在线预约网约车。包含用户叫车、自动派单、司机接单、行程结算、车辆管理等功能模块。我在该项目中担任系统架构师设计师一职，负责系统架构设计、性能优化、核心功能模块开发等工作。本文以该项目为例，主要论述NoSQL数据库技术在该项目的具体应用。通过使用键值数据库（Redis）做分布式内容缓存解决了项目的性能问题达到设计预期；通过键值数据库做分布式锁解决分布式事务状态不一致问题达到事务一致；通过键值数据库做高并发读写数据存储需求达到了克服业务数据库性能的目的。事实证明，使用这些技术手段使得项目整体能够克服业务传统数据库瓶颈和应对大规模高并发场景。最终项目得以顺利完成，取得预期目标，获得用户好评。

正文:

2017年7月，我参加了某互联公司自主研发全国运营的网约车出行项目。 我公司致力于网络出行市场服务是首批获得当地“网约车出行牌照”, 并且先后获得四川、河南、海南、山东等各地的合法运营牌照背景下。以构建一个合法、合规安全的网约车出行环境为使命。以解决用户打车难、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为目标，构建一个全国性的网约车出行平台。 网约车出行平台包括乘客端、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。

出行平台定位是一个全国性的互联网出行平台，如果用传统的关系型数据库会有很多难以克服的问题，于是我们决定采用NoSQL来解决大规模数据集合以及多种数据类型的挑战。NoSQL数据库的4大分类如下：1、键值（Key-Value）存储数据库，通用作内容缓存、处理高访问负载和日志系统。Key-Value的键值对数据模型底层使用hash table实现。具有查询速度快的优点。如：Redis、Oracle BDB。2、列存储数据库，通常用作分布式文件系统，以列簇形式存储。具有查找速度快、可拓展性强，容易分布式拓展的优点。如：HBase, Cassandra。3、文档型数据库，通常使用在Web应用中采用版本化文档数据模型以特定的半结构化文档格式（JSON）存储。如：MongoDb, CouchDB。4、图形数据库（Graph）通常使用在社交网络、推荐系统等采用图结构。如：Neo4J, InfoGrid。通过以上的分析和项目组内的讨论我们决定采用键值存储数据库（Redis）来作为内容缓存数据库，对频繁查询的数据内容做内容缓存提升查询性能。在使用过程中可以降低架构的复杂度、提升程序吞吐量、并且具有高水平拓展能力和低端硬件集群能力等优点。

下面是我在项目中通过使用键值数据库（Redis）做内容缓存解决了项目的性能问题达到设计预期；通过键值数据库做分布式锁解决分布式事务状态不一致问题达到事务一致；通过键值数据库做高并发读写数据存储需求解决传统业务数据库克服业务数据库瓶颈。

分布式内容缓存场景，

分布式锁场景，派单系统是出行平台的核心系统，需要在按照业务地区，设置的派单规则进行派单是一个比较复杂的规则系统，我们需要保障为每一个乘客匹配一个能够最快接乘客，而且司机评价最优的司机。在平台中我们的派单系统是一个多节点集群系统，而且派单实现是全异步派单，在派单的过程中，我们首先需要锁定订单，将创建成功的乘客订单上锁对其进行异步多轮派单将订单派给司机，如果N个乘客在相同的网格区域内发出N个相似的订单。那么就会存在在筛选司机的时候会同时筛选出周边满足条件的司机，那么这个时候我们在派单的时候需要对司机和乘客加锁。订单的派单时间是根据业务方可以自行设置为5-10分钟，处于乘客体验我们需要每隔5秒钟触发一轮派单，我们一开始使用的是数据库的乐观锁的机制，当订单在上千个同时派单时候，业务数据就被拖垮了，一轮派单也需要2,3分钟无法满足业务需求。最后我们通过司机乘客的缓存状态然后通过Redis分布式锁的方式来解决。最终能够实现3s同时上万订单的派发满足业务需求，而且一直稳定运行获得业务部门的好评。

高并发读写场景，在平台的车辆位置监控系统中，它是一个基于Socket的一个长连接做实时数据上报的位置信息，

经过全体成员的不懈努力。在2017年12月，先后在四川、重庆、河南、贵州、海南等城市全国开展内测。2018年2月，全国正式发布运营。上线1年多程序一直稳定可靠运行。无较大线上生产事故、查询缓存命中率在90%以上大大的减少了业务数据库的压力提升了系统的整体性能。在这段时间内NoSQL也进行过2次拓容而且性能都能够的能很好的提升。上线以来得到了多个地区交通部门的点名表扬和和上万用户的好评。也为我们后续的开发、迭代、运维奠定了一个良好的基础。为NoSQL数据库及其运用和性能优化积累了丰富的经验。

项目上线至今一直运行稳定，不论是司机端、乘客端、运营管理端，基本没有任何性能问题。但是有一次运维同事通过分析日志发现有少数Redis Key存储的内容超过100kb 过后会出现查询时间相对较长的情况。经过分析，这个是Redis本身的一个设计缺陷在这样的场景下会出现一个性能拐点。后面我们通过拆解存储设计来改善了这个问题。实践证明，项目能够顺利上线，并运行稳定，性能良好，与系统接入NoSQL数据库设计密不可分。我们对NoSQL数据库技术在项目的应用是一个持续的过程，我们接下来，还会继续不断完善NoSQL数据库技术在项目的应用，使整个出行平台更加稳定可靠。