# 摘 要

随着因特网应用和计算机技术的飞速发展，数据库逐渐成为信息系统的核心部分并广泛应用于企业、金融机构、政府及国防等各个领域。但是传统的关系数据库在面对日益增多的大量数据时暴露了很多问题，不能对其高效、实时的处理。对系统提出的新需求，传统的关系数据库并不能很好的解决。非关系型数据库是为弥补关系数据库的不足而产生的，OrientDB数据库就是非关系型数据库的一种。目前现有的非关系数据库对大数据量数据的处理能力很强但是在在很多关键领域，关系型数据库又是非关系型数据库无法代替的。如何结合关系型数据库和非关系型数据库的优点成为当前数据库领域的关键问题。

通过对分布式理论、关系数据库理论及非关系型数据库技术的学习和研究，本文基于Java语言实现了一个分布式的关系型数据库JSQL，JSQL是一个将关系数据库技术，非关系型数据库技术和分布式技术相结合的产物。JSQL采用常用的客服端-服务器架构，分为客服端和服务器端，服务器端又分为分布式管理节点和分布式数据库集群。分布式管理节点实现了数据库服务器的负载均衡和监控管理功能。分布式数据库集群节点主要包含五大模块，网络模块接收前端客服端的连接请求，对客服端进行认证和授权，并对连接进行管理，并实现基于Mysql的通信协议； Sql的解析和执行模块接收客户端的SQL请求，然后解析和执行数据库存储的调用，返回执行结果；审计模块是存储和分析所有对数据库的更改情况，向分布式管理节点提供审计数据；数据库引擎模块利用了非关系型Orientdb数据库引擎，实现可靠的数据存储；分布式模块利用hazlcast实现了数据库集群，本文提出了分布式多版本并发控制方法，用来解决分布式数据一致性问题。基于以上功能模块，ＪＳＱＬ具有高可用性，可扩展性，负载均衡等特性，同时从数据库底层考虑了数据库安全审计需求，加入了数据审计图形化界面显示审计结果。

论文对系统进行了功能和性能测试。功能测试结果表明，系统在功能上符合分布式数据库的基本要求，审计系统的功能也达到本论文的要求。论文通过对性能测试结果进行分析，认为系统的性能基本达到本论文的要求。但是本系统对复杂SQL语句的支持还不是很完善，最后提出了改进的方案。

**关键词**：分布式数据库，关系数据库，安全审计，Mysql，非关系型数据库

# ABSTRACT

With the rapid development of Internet application and computer technology, the database has gradually become the core of information system Points and widely used in enterprises, financial institutions, government and national defense and other fields. However, the traditional relational database is facing increasing When exposed to a large number of data a lot of problems, can not be efficient and real-time processing. The new needs of the system Seeking, the traditional relational database can not be solved very well. Non-relational database is to make up for the lack of relational database, the OrientDB database Is a non-relational database. At present, the existing non-relational database has great ability to handle large amounts of data But in many key areas, relational databases are non-relational databases that can not be replaced. How to combine the advantages of relational databases and non-relational databases has become the key issue in the current database area.

Through the study of distributed theory, relational database theory and non-relational database technology, This article based on the Java language to achieve a distributed relational database JSQL, JSQL is a relational database technology, The combination of non-relational database technology and distributed technology. JSQL common client-server - server architecture, divided into customer service and server-side, Server-side is divided into distributed management nodes and distributed database nodes. Distributed management nodes to achieve a database server load balancing and monitoring and management functions. Distributed database node contains five major modules, The network module accepts the connection request of the front-end application, authenticates and authorizes the client-end, and manages the connection, And realize the communication protocol based on Mysql, so easy to migrate to Mysql users to the system; Sql parsing and execution module to accept the client's SQL request, and then parse and execute the database stored call, Return the execution result The audit module is to store and analyze all changes to the database, the distributed management node to provide audit data; The database engine module makes use of the non-relational OrientDB database engine for reliable data storage; Distributed modules use hazlcast to implement a database cluster. Based on the above functional modules, JSQL has high availability, scalability, load balancing and other characteristics, while taking into account the database security audit

needs from the bottom of the database, Joined the data audit graphical interface shows the audit results.

The thesis has carried on the function and the performance test to the system. The result of function test shows that the system conforms to the basic requirements of distributed database functionally, and the function of auditing system also meets the requirements of this dissertation. Papers through the performance test results analysis, that the performance of the system basically meet the requirements of this paper. However, the system of complex SQL statement support is not perfect, and finally proposed an improved program.

**Keywords:** Distributed database, relational database, security audit, Mysql, Non-relational database

# 目 录

# 第一章 绪论

## 1.1 研究背景和研究意义

互联网诞生以来在全球迅速蔓延。 2017年我国工业和信息化部最新发布的通信业经济运行情况显示， 2月末，我国移动电话用户总数达到13.3亿户，移动互联网用户总数达到11.2亿户，使用手机上网的用户数接近10.6亿户。互联网用户数量还有很大的增长空间，特别是亚洲人口众多的发展中国家。同时，智能手机的革命发展通过移动互联网大大提升用户体验，使移动互联网迅速发展。伴随着互联网的发展出现了各种基于互联网的应用服务，从传统媒体门户网站到BBS以及近年来社会媒体的兴起，电子商务的巨大发展，还有各种各样的移动互联网应用程序。

随着互联网和互联网应用的发展，数据存储的需求不断增长。 IDC报告显示，预计到2020年全球数据总量将超过40ZB，这一数据量是2011年的22倍。在过去几年，全球的数据量以每年百分之58的速度增长，在未来这个速度会更快。如果按照现在存储容量每年百分之40的增长速度计算，到2020年需要存储的数据量甚至会大于存储设备的总容量。

未来是“大数据”时代，这么大的存储需求，给数据存储技术的发展带来了很大的压力。有很多应用基于互联网提供的各种服务正在进入井喷时代的发展，而这些应用，背景的很大一部分面临同样的问题：怎么样以尽可能廉价的方式实现大规模数据存储和查询。如搜索服务，需要存储页面而分析，微博等社交网络需求的实时用户来表达查询的观点，在线旅游需要玩家的信息和操作来存储和查询，银行需要用户帐户信息和用户用于实时存储和查询。这种应用所面临的挑战总结为大数据可用性和可靠性要求高，部分应用需要实时查询响应和高并发性和数据一致性要求。在这样的环境下，传统的关系数据库不能很好的满足现在数据库存储技术的要求，这种情况催生了 Nosql 数据库如雨后春笋般的出现。Nosql数据库与 RDBMS 相比，最大的特点是其符合 BASE 模型，并且抛弃了其 ACID 特性，具有高可扩展性、并发性等优点，更加适合大数据应用场景特别是非关系型数据的存储和管理。在过去几年，这种新兴的存储机制正在逐渐吞噬大数据的存储市场。 Nosql的一般解释为“non-relational”，不过“Not Only SQL” 也是被大众接受的主流说法之一。它是大数据时代以及 WEB2.0 时代的产物，也是解决大数据时代非结构化数据存储问题的重要手段。

Nosql 数据库具有 RDBMS 所不具备的优势，首先，扩展性好，基本上所有的 Nosql 数据库都摒弃了 RDBMS 的关系特性，数据之间很少有关联甚至无关联，这个特点在无形中使得 Nosql 在架构上扩展性更好；其次 Nosql 能够管理的数据量更大，性能更好，数据库的结构简单，一般根据特定场景从应用层来优化系统性能，再加之新型存储器件的辅助，使得在海量数据情况下，其性能同样表现更加优秀；再次，Nosql 的某些产品具有灵活的数据模型，例如 MongoDB ，用户可以随意修改字段，而这对于关系数据库来说是可望而不可及的；最后，Nosql 同样具有高可用性，在一个副本宕机之后，其他副本还可以提供服务，可以通过其他手段来同步新加入节点的数据一致性。这些特点都使得 Nosql变得越来越流行。虽然新型非关系型数据库虽然在大数据量的存储和处理上有很大的优势，但是关系数据库在很多关键领域又是无可替代的，所以如何结合关系型数据库和非关系型数据库的优点成为当前数据库领域的关键问题。

基于目前数据库应用的现状，迫切需要开发将分布式非关系数据技术和关系数据库技术相结合的分布式数据库来解决系统对大容量、可用性和可扩展性的需求。如何将关系数据库技术，非关系数据库技术和分布式技术的优点相集合来开发一个新的分布式数据库具有极现实的意义。相对于传统关系数据库，这种分布式数据库容量和可用性具有很大的优势，因此能够更好的应对大规模和超大规模的数据存储需求的应用。本文设计和实现的分布数据库系统(以下简称JSQL)就是一种将分布式非关系数据库技术和关系数据库技术相结合的系统，一旦JSQL开发成功以后，将会给系统和软件平台带来以下好处：

1.结合关系数据库和NoSQL的优点

JSQL分布式数据库系统是一种将非关系型数据技术和关系数据库技术相结合的系统，首先，它有Nosql系统的优点，能够处理非常大的数据，有很好性能和可扩展性。然后，JSQL作为关系数据库系统，能满足应用程序的开发要求，为应用程序提供SQL接口，提供复杂的关系操作和事务支持。

2.提高数据库系统资源使用效率

JSQL分布式数据库系统具有负载均衡功能，负载均衡在多节点之间按照一定的策略分发网络或计算处理负载, 提供了一种廉价而有效的方法来扩展服务器带宽, 增加吞吐量, 提高数据处理能力, 同时又可以避免单点故障。在JSQL中，分布式管理节点作为分布式数据库系统的负载均衡管理节点，负责路由消息到指定的分布式数据库节点，分布式管理节点采用动态局部性最少连接算法来实现分布式数据库节点的负载均衡，提高对分布式数据库节点的使用效率。

3.提高系统的访问效率

本文设计的分布式系统能为客户端选择最合适的分布式数据库节点，这样能够提高客户端的访问效率。系统为客服端选择最近的节点，同时也节约了系统的网络资源。

4.提高系统的可扩展性

本文设计的分布式数据库系统，其分布式数据库节点集群采用无主服务器设计，能动态的增加节点，而不需要用户任何的配置。在系统需要扩展时，只需要增加一个或几个分布式数据库节点就能够提高系统的整体性能；在系统不需要大容量存储时，可以减少一个或几个分布式数据库节点降低系统的整体容量。

5.提高系统可移植性

本文设计和实现的分布式数据库系统全套代码由JAVA语言实现，在Windows操作系统和Linux 操作系统之间，代码的可移植性较强，方面后续跨平台部署。

6.方便数据库的安全审计

数据的重要性越来越重要，如何保证数据库的安全性是现在急切需要解决的问题，如何防止用户随意篡改数据在系统的设计上作为一个重要的功能需求。JSQL从数据库底层实现了数据库的审计功能，提供比其他方法更高的性能。管理员通过管理客户端能够监视数据库系统的运行状态，能监控数据的更改情况。

## 1.2 数据库历史与现状

数据库技术的发展始于20世纪60年代。在没有数据库的情况下，使用计算机存储数据的用户（主要是财务科研单位）以操作系统中的文件的形式存储数据。随着业务的发展，应用程序变得越来越复杂，人们开始需要开发管理数据在通用软件，这促成了数据库的诞生。 20世纪60年代以来，人们探索数据模型实现数据库，后来开发出三大数据库模型：层次数据模型，网格数据模型，关系数据模型，后来也出现过对象数据模型。其中，可以使用关系数据模型严格的数学理论来描述数据库的组织和操作，具有简单灵活，数据库独立性高的特点特征。从20世纪70年代到90年代，关系数据库理论成熟并得到广泛应用，这个里程碑是1974年，IBM的圣荷西，加利福尼亚研究实验室的D.D. Chamberlin和Ray Boyce开发了SEQUEL结结构化查询语言，后来在1980年更名为SQL。 SQL是数据库中的标准数据查询语言，在1986年，由美国国家标准学会规范SQL，就这样成为了数据库系统的标准语言。 SQL包含三个部分：数据定义语言，数据操作语言，数据控制语言。 SQL是一种全面的通用关系数据语言，可以当它是一种高级的非程序语言，它允许用户在高级数据结构中工作。使用SQL 用户无需知道数据的具体存储方式。在SQL中一个简单的语句实现效果，使用其他编程语言需要很大一

部分程序才能实现。另外，SQL通过使用这种语言允许用户掌握这种语言就可以使用这种语言来操作任何一个标准数据库产品。到20世纪90年代，关系数据库标准几乎适用于任何数据存储需求的应用程序。

分布式数据库系统是数据库系统技术与网络技术的结合。分布式数据库系统（DDBS）的研究始于20世纪70年代。但是，分布式数据库的理论与应用成为一个热门话题，是20世纪90年代的事情。 90年代，互联网网络出现爆炸式增长，同时各种应用程序对存储的需求与日俱增。在这样的环境下，分布式数据库系统的研究成为了热门话题，人们探索分布式数据库系统理论和关键技术，并快速应用理论实践，开发了各种商业应用价值的数据库产品。 2002年，Eric Brewer提出了引导分布式数据库研究的重要理论，并且后来证明是正确的，这个理论就是CAP定理，CAP定理的核心观点是：在分布式计算系统中，不可能同时满足以下三点：一致性，可用性，分区容忍性。 CAP理论认为分布式数据库产品的设计必须介于三者之间，其中至少有两个可以满足，不可能同时满足三个。根据CAP理论的指导，有学者认为，基于传统的关系数据库构建分布式数据库，很难满足当前大型数据存储需求的各类互联网应用，如搜索引擎，社交网络等。由于传统的关系数据库非常重视数据一致性，在传统的关系数据库中，交易的四点必须保证要求：ACID（Atomicity，一致性，隔离，持久性）。有部分人认为，根据CAP理论，在满足强大的一致性之后，也希望获得互联网应用的高可用性性是非常困难的。

在二十世纪九十年代末和二十一世纪初，互联网应用大幅增长出现了一些人认为是革命性的分布式数据库概念：NoSQL（not only SQL）。 NoSQL和传统的关系数据库非常的不同，对于一个概念，NoSQL的显着特点是：非关系型，分布式，不提供ACID数据库设计模式。 NoSQL不支持复杂的关系操作，不提供对交易一致性的支持。由于摆脱了必须满足支持一致性的功能要求，根据NoSQL概念设计产品生成的高可扩展性高并发高可用性性支持得到了很大的改善，各种类似的开源或商业的NoSQL产品出现了，并且被大量和被各种互联网应用程序所使用。这些NoSQL产品被应用到新的产品，如博客，论坛，微博等其他社交服务。这些应用程序具有高度可扩展性，高可用性，并发性。高可用性这些功能很重要，因为这些应用压力主要来自大规模数据存储，大量并发访问及时响应， NoSQL拒绝一致性支持是合理的，因为像社交网络应用程序一样用户丢失数据库的后果不会太严重，毕竟，这不会导致用户遭受巨大的损失，如经济利益。

作为分布式数据库，NoSQL在过去两年爆发式增长，均受益于互联网高速发展， 也是各种应用到“大数据”的发展趋势，CAP理论再一次证明了它的正确性。毕竟，NoSQL放弃了一致性的支持，这些产品可以应用到请求的一致性不

高在应用上，随着互联网的快速发展和“大数据”时代的到来，那些需要关系操作，需要高级一致性并且面临大规模数据存储查询要求的应用，比如电子商务，这样的应用程序不能使用NoSQL产品。所以，目前的分布式数据库开发方向，根据CAP理论，一个是尽可能的减少一致性，提高可用性，另一个方向是需要尝试确保操作之间的关系，一致性的支持，同时利用分布式技术的优势，提供尽量高性能的服务。

下面对对目前市面上比较有代表性的分布式数据库系统进行简单的介绍。

1.Google Spanner

Spanner 是一个可扩展的、全球分布式的数据库，是在谷歌公司设计、开发和部署的。

在最高抽象层面，Spanner 就是一个数据库，把数据分片存储在许多 Paxos状态机上，这些机器位于遍布全球的数据中心内。复制技术可以用来服务于全球可用性和地理局部性。客户端会自动在副本之间进行失败恢复。随着数据的变化和服务器的变化，Spanner 会自动把数据进行重新分片，从而有效应对负载变化和处理失败。Spanner 被设计成可以扩展到几百万个机器节点，跨越成百上千个数据中心，具备几万亿数据库行的规模。

应用可以借助于 Spanner 来实现高可用性，通过在一个洲的内部和跨越不同的洲之间复制数据，保证即使面对大范围的自然灾害时数据依然可用。作为一个全球分布式数据库，Spanner 提供了几个有趣的特性:第一，在数据的副本配置方面，应用可以在一个很细的粒度上进行动态控制。应用可以详细规定，哪些数据中心包含哪些数据，数据距离用户有多远(控制用户读取数据的延迟)，不同数据副本之间距离有多远(控制写操作的延迟)，以及需要维护多少个副本(控制可用性和读操作性能)。数据也可以被动态和透明地在数据中心之间进行移动，从而平衡不同数据中心内资源的使用。第二， Spanner 有两个重要的特性，很难在一个分布式数据库上实现，即 Spanner 提供了读和写操作的外部一致性，以及在一个时间戳下面的跨越数据库的全球一致性的读操作。这些特性使得 Spanner 可以支持一致的备份、一致的 MapReduce 执行和原子模式变更，所有都是在全球范围内实现，即使存在正在处理中的事务也可以。

2.阿里巴巴OceanBase

OceanBase是一个支持海量数据的高性能分布式数据库系统，实现了数千亿条记录、数百TB数据上的跨行跨表事务，由淘宝核心系统研发部、运维、DBA、广告、应用研发等部门共同完成。在设计和实现OceanBase的时候暂时摒弃了不紧急的DBMS的功能，例如临时表，视图(view)，研发团队把有限的资源集中到关键点

上，当前 OceanBase主要解决数据更新一致性、高性能的跨表读事务、范围查询、join、数据全量及增量dump、批量数据导入。

目前OceanBase已经应用于淘宝收藏夹，用于存储淘宝用户收藏条目和具体的商品、店铺信息，每天支持4～5千万的更新操作。等待上线的应用还包括CTU、SNS等，每天更新超过20亿，更新数据量超过2.5TB，并会逐步在淘宝内部推广。

OceanBase设计和实现的时候暂时摒弃了不紧急的DBMS的功能，例如临时表，视图(view)，研发团队把有限的资源集中到关键点上，当前OceanBase主要解决数据更新一致性、高性能的跨表读事务、范围查询、join、数据全量及增量dump、批量数据导入。

3.微软SQL Azure

SQL Azure (旧称 SQL Server Data Services 或 SQL Services) 是由微软SQL Server 2008为主，建构在Windows Azure云操作系统之上，运行云计算 (Cloud Computing)的关系数据库服务 (Database as a Service)，是一种云存储(Cloud Storage)的实现，提供网络型的应用程序数据存储的服务。

SQL Azure提供了一个功能强大且为人熟知的存储结构，SQL Server 2008 90%的功能在SQL Azure中都得到了完美的支持。SQL Azure兼具云计算的好处，将数据存储基础结构托管，可以大大地降低企业在IT方面的资源的投入，根据具体需要的数据存储量和网络带宽支付进行支付，即我们常说的，即付即用!

数据在企业中扮演的角色越来越重要，确保数据安全，确保数据的高可用性，也是近年来企业不断追求的目标。SQL Azure完美地保证了高可用性(High Availability)和故障转移(Failover)，当应用SQL Azure进行数据的增删改的时候，SQL Azure会自动地将数据备份到若干节点(node)，以保证数据的高可用性;SQL Azure后台内置的集群机制完美的保证了自动故障转移，无需人工监守。

## 1.3 论文的主要工作

通过对分布式理论、关系数据库理论及非关系型数据库技术的学习和研究，论文基于Java语言设计和实现了一个分布式的关系型数据库JSQL ，JSQL结合了非关系型数据库OrientDB和关系型数据库的优点，同时实现了Mysql的通信协议，是一个兼容Mysqk通信协议的分布式数据库。在系统的实现过程中做的主要工作包括：

1.利用OrientDB非关系型存储引擎设计和实现了关系型数据库的本地存储系统。

2.实现了Mysql通信协议，使得可以通过Mysql客服端来连接JSQL分布式数据库，方便Mysql用户迁移到本数据库系统。

3.实现了对SQL语句的解析和执行，完成对关系数据库接口的支持。

4.实现了数据库的分布式架构，使得数据可以存储在多台计算机上面。

5.实现了系统的审计系统，使得系统的安全性得到增强。

6.实现了局部最少连接算法，完成了分布式数据库节点的动态负载均衡功能。

7.利用分布式多版本并发控制机制解决了分布式数据一致性问题。

8.数据库系统做完以后，对系统进行了各项软件测试，包括功能测试，性能测试等，以验证系统是否满足需求，达到了系统设计的目标。

## 1.4 本论文的结构安排

第一章作为本论文的绪论，首先介绍了论文的选题背景和本论文进行研究的意义。接着阐述了数据库的发展历史和现状，对当前几个流行的分布式数据库系统进行了简单的介绍。最后一节里介绍了论文的主要工作。

第二章，主要介绍了本论文相关的理论基础和相关的技术。这一掌首先对数据库系统方面的理论知识进行了说明，包括数据库系统和关系数据库系统的概念，简单的讨论了NoSQL数据库系统；然后介绍分布式数据库相关理论，对分布式系统中数据库分布式和数据复制技术进行了介绍。最后介绍了LBLCR动态均衡算法和分布式MVCC技术，这些技术为论文的实现提供了坚实的基础。

第三章，作为分布式系统JSQL的分析。本章首先给出了分布式数据库JSQL的实现目标，对系统进行需求分析。从系统用户的角度出发，对分布式内存数据库进行详细、切合实际的需求分析，并在需求分析的基础上，对分布式数据库的整体架构等进行模块划分，按照功能分解结构细化、模块化系统功能，并对各个模块进行再划分。然后本章对重要的模块进行了分析，包括对各种技术和框架的选择，详细说明了实现分布式数据库系统需要用到的各种技术。

第四章，是系统的设计，包括总体设计和模块划分，本章对系统的总体架构和各个模块的详细架构给出了阐述。系统主要包括分布式管理节点和分布式数据库节点。本章对分布式管理节点和分布式数据库节点的各个模块进行了详细的分析。

第五章，是关于系统的具体实现。在前面模块设计的基础上，给出每个模块的详细实现，重点对分布式管理节点和分布式数据库节点功能的实现进行阐述，对本系统中的关键算法进行了详细的流程分析。

第六章，论文对jsq分布式数据库进行测试，包括功能测试和性能测试。

第七章，论文总结了论文所做的工作，对本系统的优点和不足进行了说明。并对未来的工作进行了展望。

最后部分，是关于致谢和参考资料。

# 致 谢

时间过的真快，转眼间，2年多的研究生生活就快要结束了。回顾自己的研究生学习生涯，感慨万千。论文的完成，除了自己的努力以外，更离不开老师和学弟们的帮助，在论文成稿之际，衷心感谢给予自己悉心指导和热情帮助的各位老师和学弟们。

论文的完成，首先要感谢我的校内导师曹晟教授，在整个论文的的写作过程中，曹老师都给予了我很大的关心和帮助。从作者毕业论文的选题、写作一直到最终完成的过程中，曹老师都是在百忙的工作中以一贯认真负责的态度认真仔细阅读作者的论文，给予作者耐心的指导，使得论文能够顺利的完成。他严肃的科学态度，严谨的治学精神，以及精益求精的工作作风，深深地感染和激励着我。

在这一年多的时间里，我还要感谢我的学弟们。感谢你们陪我一起开发这个分布式数据库系统，我们一起学习，一个努力，才能让本系统顺利完成。任何一个系统都要靠团队合作，更何况分布式系统这个既具有挑战性的工程项目，没有你们的帮助，就不可能按时完成这个系统。对学弟们的帮助，在此表示非常的感谢。

最后，作者非常感谢负责评审论文的教师、专家和教授，感谢你们认真负责的阅读论文，感谢你们为论文提出的宝贵意见和建议。