**大连理工大学本科毕业设计（论文）**

**TPC-H数据库性能测试工具设计与实现**

**Design and Implementation of Database Performance Testing Tool for TPC-H**

学 院（系）： 软件学院

专 业： 软件工程（日语强化）

学 生 姓 名： 关鹏

学 号： 201293076

指 导 教 师： 王洁

评 阅 教 师： 侯刚

完 成 日 期： 2016年 5月30日

大连理工大学

Dalian University of Technology

# 摘 要

国际非盈利性机构-事务处理性能委员会（Transaction Processing Performance Council，简称TPC）在过去二十多年时间内，发布了多个数据库评测标准，其中委员会所制定的TPC-H性能基准被学术界和工业界普遍采用，用来决策和评价数据库在技术方面应用的性能。该数据模型基于商业应用环境，对实际的数据库操作进行了真实的模拟。该标准同时严格定义数据模型构造，并根据实际用户操作模拟出核心的查询与更新操作，通过操作细节测试出数据库管理系统的性能指标。与此同时，该标准还对数据库的四个事务特性 ACID，也就是原子性，一致性，隔离性与持久性，进行了详细的测试说明和发布了详细的测试标准。

在软件的设计过程中，首先整体结构遵循标准，SUT（被测试系统）上配置数据库，测试工具通过远程进行连接，同时为了统一数据库测试接口，采用ODBC方式，因此能方便的支持所有数据库进行性能测试。其次在操作设计上，简化测试环境配置，方便用户快速搭建测试平台，使用上简单易懂。

软件开发完成后，拥有以下优势，首先对于选择数据库的用户而言，只有准确的数字指标才能给予用户最直观的感受，了解到不同数据库在同一模型下的表现，进而体现出性能的差异。其次数据库厂商虽然也可以根据标准模型进行系统优化，但只是定性并没有定量，所以设计数据库性能测试工具能够准确地提供量化指标并有利于数据库厂商之间的竞争，推动数据库产业发展，并由性能上的提升进而节约机器上的功耗。

**关键词：**数据库；性能测试；TPC-H；联机事务处理

**Design and Implementation of Database Performance Testing Tool for TPC-H**

# Abstract

In the last 20 years International non-profit organization, Transaction Processing Performance Council (namely TPC) has published many database evaluation criteria. The TPC-H performance benchmarks which the Commission developed has been commonly used by academics and industry for decision-making and performance evaluation of the technical aspects of database applications. The data model is based on the business application environment with universality and simplicity. Meanwhile real simulation of the actual database operation is carried out. The standard strictly defines the structure of data model and simulates the core of the query and update operation according to the actual operation of the user. Also the standard tests the performance index of database management system through the operational details. At the same time, a detailed description and detailed testing standards are presented included ACID properties of a transaction database, which is atomicity, consistency, isolation and durability,

In the software design process, the first overall structure follows a standard. A database is configured on the SUT (system under test). Testing tools are successfully connected through a remote connection. At the same time in order to unify the database test interface using ODBC way, it can easily support all database performance test. Secondly it which is user-friendly platform to quickly build test and is easy to understand simplifies the configuration of the test environment in operating design,

After the completion of software development, it has the following advantages, Firstly, only the exact numerical indicators will give users who choose the database the most intuitive feelings. Users will understand the performance of different databases in the same model, which in turn reflects the difference in performance. Secondly, although the database vendors can also be optimized according to the standard model of the system, but the result is only qualitative and not quantitative. So the design of database performance testing tool can be capable of providing accurate and quantitative indicators in favor of competition among database vendors, to promote the development of database industrial. Meanwhile the development of performance will save power lift on the machine.

**Key Words：**Database；Performance Testing；TPC-H；OLTP

目 录

[摘 要 I](#_Toc453267133)

[Abstract II](#_Toc453267134)

[1 绪论 1](#_Toc453267135)

[1.1 工程背景 1](#_Toc453267136)

[1.2 国内外研究概况 1](#_Toc453267137)

[1.3 主要研究工作 5](#_Toc453267138)

[1.4 论文组织结构 5](#_Toc453267139)

[2 项目相关技术 7](#_Toc453267140)

[2.1 数据库相关技术 7](#_Toc453267141)

[2.2 C#相关技术 8](#_Toc453267142)

[2.3 相关术语定义 9](#_Toc453267143)

[3 数据库测试系统需求分析 11](#_Toc453267144)

[3.1 功能性需求分析 11](#_Toc453267145)

[3.1.1 系统流程分析 11](#_Toc453267146)

[3.1.2 系统用例图 12](#_Toc453267147)

[3.1.3 系统结构分析 13](#_Toc453267148)

[3.2 非功能性需求分析 14](#_Toc453267149)

[4 数据库测试系统设计 15](#_Toc453267150)

[4.1 TPC-H基准分析 15](#_Toc453267151)

[4.1.1 TPC-H测试模型介绍 15](#_Toc453267152)

[4.1.2 数据库实体与关系 16](#_Toc453267153)

[4.1.3 TPC-H测试工具包 16](#_Toc453267154)

[4.1.4 TPC-H度量指标 19](#_Toc453267155)

[4.2 TPC-H Driver整体软件架构 20](#_Toc453267156)

[4.2.1 软件的体系架构 20](#_Toc453267157)

[4.2.2 软件的逻辑架构 20](#_Toc453267158)

[4.3 TPC-H Driver软件功能设计 22](#_Toc453267159)

[4.3.1 主控与日志 22](#_Toc453267160)

[4.3.2 数据库配置与数据装载测试 23](#_Toc453267161)

[4.3.3 Power与Throughput测试 24](#_Toc453267162)

[4.3.4 ACID测试 26](#_Toc453267163)

[4.4 本章小结 26](#_Toc453267164)

[5 数据库测试系统实现 27](#_Toc453267165)

[5.1 程序实现原则 27](#_Toc453267166)

[5.2 主控与日志实现 28](#_Toc453267167)

[5.3 数据库操作实现 30](#_Toc453267168)

[5.3.1 数据库连接配置 30](#_Toc453267169)

[5.3.2 数据生成 31](#_Toc453267170)

[5.3.3 数据加载 32](#_Toc453267171)

[5.4 性能测试实现 33](#_Toc453267172)

[5.4.1 Power测试实现 33](#_Toc453267173)

[5.4.2 Throughput测试实现 33](#_Toc453267174)

[5.5 ACID测试实现 34](#_Toc453267175)

[5.6 本章小结 36](#_Toc453267176)

[6 测试结果与分析 37](#_Toc453267177)

[6.1 测试环境 37](#_Toc453267178)

[6.2 测试执行流程 37](#_Toc453267179)

[6.2.1 数据库和ODBC配置 37](#_Toc453267180)

[6.2.2 测试数据库的创建 37](#_Toc453267181)

[6.2.3 执行性能测试 39](#_Toc453267182)

[6.2.4 执行ACID测试 39](#_Toc453267183)

[6.3 性能测试结果与分析 39](#_Toc453267184)

[6.4 ACID验证结果与分析 40](#_Toc453267185)

[6.5 本章小结 41](#_Toc453267186)

[结 论 42](#_Toc453267187)

[参 考 文 献 43](#_Toc453267188)

[致 谢 44](#_Toc453267189)

# 1 绪论

## 1.1 工程背景

互联网数据中心（Internet Data Center,简称IDC）的报告称，目前全球数据的持续增长速度在每年40%左右，如果按照该增长率进计算，如果 2016年全球数据总量在12ZB左右，那么2017年全球数据总量将达到16.8ZB，2018年将达到23.52ZB左右，截止到2020年，全球的数据总量将达到40ZB。

从目前中国公开数据量来看，2013年百度作为国内搜索引擎的霸主，所拥有的数据数量达到EB级别，而BAT中另外两家公司阿里巴巴和腾讯的数据总量也表示达到了百PB。此外，在其他各个行业，包括电信、医疗、金融、公共安全、交通、气象等所保存的数据量也都达到数十乃至上百PB级别。

在巨大数量信息存储的今天，数据库系统承载了绝大部分的信息系统。也正是因为数据库系统所拥有的数据集中管理、数据共享简洁、数据使用方便的原因，才被如此广泛的使用到各个行业中。而随着数据数量级的增大，企业逐渐意识到数据中存储的信息价值，更多的去挖掘和收集信息，扩大数据量，带来更大的利润，正是由于数据和信息的相关性，使得数据库系统成为了企业和行业中的核心。

在数据量不断扩大的今天，数据库性能就显得尤为重要，也一直是数据库系统研究的核心问题。在20世纪80年代，就有很多数据库性能测试标准被提出，但由于标准不够严密，数据库厂商提供的数据不一，没有形成公共认可的标准。后来由数十家会员公司在美国创建的非盈利组织TPC统一了测试标准，由于TPC成员主要由计算机硬件厂家组成，所以其制定的数据库测试标准、测试模型和发布的测试结果具有权威性。

TPC在20多年的时间里，推出了多套测试标准和程序，包括现在普遍适用的TPC-H、TPC-C和TPC-W，由于标准过久而被放弃的TPC-A、TPC-B、TPC-D和TPC-R，还有没被所有大型企业接受的TPC-S和TPC-E等等，其中提出的TPC-H的标准，描述了拥有一定数量级下的数据库，待测系统每小时对其查询完成的数量，该标准应用于OLAP环境，测试在特定查询下的待测系统支持能力，侧重于分析处理数据的能力。

## 1.2 国内外研究概况

在开源社区中，许多开源技术如Hadoop、Spark、Map/Reduce除了在云计算领域被使用和商业化之外，在大数据领域上同样得到了使用和发展，同时对于数据库的要求也在不断提高，对数据库技术发展起到了推进作用。研究表明，当前所谓的大数据技术主要有特点简称为3V，也就是Volume（海量）、Velocity（高速）和Variety（多样）。TPC的联合主席之一Raghunath Nambiar在当前大数据3V特点下，又提出了大数据还需要有Value（价值）和Veracity（精确）的这两个特性。如何对不同的数据库管理系统进行比较，进而便于进行大数据的部署，变成了当今重要的研究课题。

事务性能管理委员会在数据库管理系统方面是当今世界上最为著名的评测管理团队，对当今数据库评测做出了重要贡献。从20世纪80年代开始，该组织就开始发布各种数据库评测标准，迄今为止已经发布了很多类型的指标。如已经被废弃的TPC-A、TPC-B和正在使用的TPC-C、TPC-E和TPC-H等等，并且在整个数据库行业中得到了广泛的认可和应用。同时，在有关Apache的开源社区中，针对大数据领域中的开源软件Map/Reduce已经制定了多种测试指标和测试用例。由于对大数据测试基准的研究在国内起步比较晚，所以到现在为止还未建立起权威的数据库测试基准。针对此现状，目前由中国信息通信研究院领头，与中科院计算技术研究所以及国内外知名公司和机构合作，制定的大数据测试基准正在逐步测试中[1]。

为了方便企业选择数据库管理系统的测试基准，或者普通的数据测试基准，本文将结合前人的研究成果，以此为基础，更深入的讨论数据测试基准的各种必需要素。并在讨论后，对现有的数据测试基准进行讨论，并对TPC-H测试基准进行更深入的挖掘和探索。

对于企业来说，真正有效的测试基准需要很多要素，其中对于业务相关性，数据真实性，数据生成方式，数据压力设定，数据统计方式等有着严格的要求，所以制定一系列标准时非常复杂而又充满难度的事情，除了以上几点外，包括测试基准的抗压性，对于SQL语言的兼容性还有标准的平台移植性等等都是十分关键的因素。下面将就以上几点进行理解和分析。

（1） 自身业务相关性

对于企业来说，在选择数据库管理系统的测试标准时，第一步就要考虑标准的模型与自己业务的相符度，二者越相符那么相关性就越强，相关性越强测试指标则越有参考价值。

其中基准的模型主要描述了测试执行的场景，根据该场景即可判断企业是否与其一致或者类似，比如说游戏开发的数据库评测和图书馆管理系统的数据库评测相关性微乎其微，所以说与自身业务相符度低的测试标准被企业使用后，即便对于整体数据库测试的结果显示非常好用，适宜选择该数据库作为企业来用，但实际上并非如此。

（2） 保证数据真实可用

在数据测试前如果已经能够保证所选标准和自身业务的相符度，那么该标准也必须产生相应的真实场景数据来进行呼应，如果数据并非真实可用，那么业务即使再相符，同样没有参考价值，其实这里也是对测试标准提出了更多的要求，即数据的真实可用性和业务相符性，正是数据才能够让用户模拟出真是的应用场景，让模拟数据和真实数据相一致，更好的让用户得到参考和测试。

（3） 数据压力的设置具有可扩展性

对于不同的计算机，如普通笔记本和服务器，还有对于不同的数据库，如MySQL和ORACLE，都要能够产生让它们同样能导入的数据规模，同时该数据提供不同的比例因子来产生不同的大小，这样的话才能让用户更清晰的看到，在不同的数据量的压力下，到底哪个数据库布置在哪台电脑上能够产生最大的性能指标，因为不同用户的数据量需求也不同，比如小型企业的数据库就不需要用大量的数据来进行测试，这样反而与其应用不符，故而数据压力设置的可扩展性尤为重要。

（4） 测试指标选取易于理解

不同的测试标准会有不同的测试指标，指标的目的是为了给用户最直接的感受，让用户明白哪个数据库测试系统才是真的性能指标更高更好，所以指标的易于理解性尤为重要，以TPC-H为例，以查询语句的时间作为测试指标的一部分，便于用户去理解，因为查询时间越短，说明数据库访问越快，响应越快，那么该数据库的性能也就越好。

（5） 测试指标保证客观与公平

众所周知，在竞技比赛中，一个人不能既是运动员又是裁判员。测试基准好比竞技比赛中的裁判员，应该由中立的第三方机构制定。事实也证明，在各个领域最受欢迎的测试基准都是有第三方机构设计的。过去20多年的经历证明TPC系列基准是数据库领域最为广泛接受的基准。除此之外，第三方机构的审计也是保证评测结果的客观性与公正性的重要手段。

（6） 测试标准抗压能力强

所谓的抗压能力强，也就是不会很容易的崩溃掉，同时保证测试的公平性。比如在进行某项测试的过程中，在该测试标准下，每种数据库都轻易的崩溃掉，不能够体现真实的数据，或者某个数据库能够通过测试，但是其他的完全给不出来任何有效的测试值，那么这个测试标准都不能说是成功的，由此可见，抗压能力强对于公平的重要性。

（7） 对SQL语言的兼容性

SQL语言是数据库通用的语言，不同的数据库厂商之间虽然有差距，但是基本语法基本一致。这种语言主要用于对数据库中数据的添加、删除、修改和查询，管理关系数据库，编写存储过程，增加数据索引，提高数据效率等。SQL基本独立于数据库，电脑配置、计算机网络、操作系统情况等均不对SQL有任何影响，正因如此，使得其移植性十分强大，也正是SQL语言的可移植性才使得符合SQL标准如此的重要，同时SQL支持与其他语言嵌套，使得能够实现高级语言不能实现的功能，SQL从1986年开始进行统一规范，在当今几乎所有的数据库厂商均支持该规范，如ORACLE、MySQL、DB2等等，所以测试标准对该规范的支持，直接决定了测试的通用化，是测试能否正常被用于各种数据库的根本[2]。

（8） 测试的普遍性

普遍性主要指的是针对不同数据库的物理结构与逻辑结构，均能够进行普遍的测试并能够得到可用的指标，同时对于测试工具的具体实现不应该有所规定，而应该专注于测试规范基准，而其他的数据库管理系统，在测试过程中，只要遵循了该规范，那么就能够很容易的得到规范要求的结果，也就说明该测试具有了普遍性。

TPC-H的前身是TPC-D基准。TPC-H用遵照第三范式（3-NF）模拟零售商市场分析，其中共有8张数据表，8张表之间有着一定的数据关系，该数据关系又与比例因子有联系，在TPC-H的官方标准中有具体的描述，总体来说这8张表的数据量从1GB到3TB均可以生成，提供了8个级别的数据，其中REGION和NATION表的行数是固定不变的，而其他的6张表的行数因比例因子的参数而不同。

TPC-H的查询语句具有如下特征：

（1） 具有高度复杂性；

（2） 使用各种各样的访问模式；

（3） 查询语句带有随机的特性；

（4） 检查可用数据的大多数数据；

（5） 22条SQL各不相同；

（6） 每次查询的参数可变。

从官方标准中给出的模板来看，使用到SQL语言的地方都是符合SQL标准的，包括SELECT、UPDATE、INSERT、IN、EXISTS等等一系列数据库语句的操作，由此可见TPC-H测试基准符合以上所有的特性，是一种指标客观、普遍性强的测试基准。

我国高技术产业保持快速发展，依靠科技创新，取得了很多成绩，在信息技术服务、数字内容服务、电子商务和研发设计服务等高技术服务业方面尤为突出，在日新月异的发展下，市场也变得越来越饱和，竞争趋于白热化，对于用户的选择也变得困难起来，在数据库系统的选择上尤为突出，正是在这样的环境下，测试基准变得更加重要，为软件厂商提供了了解自身的平台，也为用户提供了了解数据库系统性能的渠道，TPC-H测试基准对于整个数据库行业有着不可替代的促进作用。

## 1.3 主要研究工作

本文对基于TPC-H标准的数据库测试软件的设计与实现进行详尽的阐述，在整个过程中构造了完整的测试系统，其中的测试功能简单实用，达到了基本的软件需求，完成了整个系统的设计目标，满足了系统的稳定性需求，在实现过程中修复了很多潜在的问题，使得整个软件更加完整。

整个基于TPC-H标准的数据库测试系统 TPC-H Driver采用C/S架构，包括待测数据库端和客户端，其中待测数据库端主要用于部署数据库，而客户端主要包含以下功能：

（1） 数据库配置：主要配置所要连接数据库的基本信息，包括数据库类型、用户名、端口号和密码等；

（2） 数据生成：调用DBGEN生成规定规模的数据量；

（3） 数据装载：将DBGEN生成的数据通过ODBC接口载入；

（4） Power测试：是随机执行22条查询流中的一条测试流和2条更新流中的一条测试流；

（5） Throughput测试：执行的是多条查询流和一条更新流对数据库的混合操作；

（6） ACID测试：测试原子性，一致性，隔离性与持久性；

（7） 结果显示和日志输出：显示测试结果和输出操作日志，方便用户阅读；

（8） 服务器主要工作是将测试结果交给客户端读出；

（9） TPC-H Driver遵守TPC-H的标准规范，操作简洁方便，界面清晰易懂，测试结果准确，能直观反应出被测系统的性能。

## 1.4 论文组织结构

共分为五章，每章内容如下：

第一章：从工程背景、国内外研究概况、主要研究工作和论文组织结构进行介绍，对工程的来历和发展现状做整体性的理解和把握。

第二章：介绍主要使用的项目相关技术，首先从数据库相关技术进行介绍，主要介绍了现在普遍被使用的数据库MySQL，其次对实现语言相关技术进行介绍，主要介绍了C#在使用中使用的技术，最后是对论文中相关术语定义进行解释，方便阅读和理解。

第三章：介绍整个系统设计方案，首先从TPC-H的标准进行分析，理清系统的结构，并对标准要求的模型和商业环境进行深入理解，对于基准中给出的数据库实体和关系有一个基本的认识，接下来获取官网的工具包，同时进行编译生成，清楚DBGEN和QGEN的使用方法。其次明晰TPC-H Driver的设计目标，对整体实现的功能思考到位。然后对软件的整体架构进行解释，主要从体系架构和逻辑架构两方面来看待。最后是软件的功能设计，细化了设计目标，并从主控与测试信息、数据库配置与数据装载、Power与Throughput测试和ACID测试四个模块来进行详细的阐述。

第四章：从设计中具体描述几个主要模块的实现，在描述前首先对程序实现原则进行了详细的阐述；之后对主控与日志模块的实现进行了详细的描述；数据库操作同样是程序实现的重要部分，在阐述该部分时，分别对数据库连接配置、数据生成以及数据加载进行了讲解；接下来对性能测试的实现进行了描述，对Power测试和Throughput测试的具体实现进行了理解；最后明确了ACID的测试实现。

第五章：对整个测试最后实现的结果进行了显示和分析。首先介绍了测试环境；其次对测试执行流程进行了详细的描述，主要包括数据库和ODBC配置、测试数据库的创建、执行性能测试、执行ACID测试四个步骤；之后对性能测试的结果进行分析；最后分析了ACID的测试结果。

# 2 项目相关技术

## 2.1 数据库相关技术

（1） MySQL的概述

MySQL在当今世界IT行业中是一个被普遍使用的关系型数据库管理系统，最开始开发由瑞典MySQL AB公司完成。后来在08年被Sun公司收购。09年ORACLE又收购了Sun公司。MySQL最开始基本没有人觉得它能够发展壮大。但是经过一段时间发展之后，MySQL现在被很多中小型企业青睐使用。其中最重要的原因就是MySQL拥有的速度快、体积小和成本低，其中尤以源码开发这一特点最吸引注意，很多网站为了快速开发和降低成本选择了MySQL作为后台数据库管理系统[3]。

（2） MySQL的特性

① 使用方便，提供了多种图形界面化操作平台，如MySQL Workbench和MySQL Administrator等，便于对数据的操作和使用。

② 除了本地连接数据库之外，还提供了大量的接口供编程人员使用，如微软使用的ODBC（开放数据库连接）接口、JAVA语言使用的JDBC（JAVA数据库连接）以及TCP/IP数据库连接等。

③ MySQL允许对算法就行优化，优化后的程序能够极大程度的提高查询速度和效率，由此可看出对于优化算法的支持。

④ 它自身作为数据库可以被使用在MVC模型中，独立作为后台的数据存储而存在，也可以使用库文件来被其他语言调用，提供数据库支持，而且其列名与表名与其他数据库具有通用性，其他数据库的数据也能够很容易的导入。

⑤ 能够和多种语言进行镶嵌使用，并且提供了其他语言调用的工具包和相应的函数，其中包括C、Java、C#、C++、Python和PHP等主流语言。

⑥ 为了提高CPU的利用效率，允许使用多线程调度，其中还包括远程多线程测试。

⑦ 支持Windows、Linux和Mac OS等主流操作系统，对于不同的操作系统搭建一样简便。

⑧ 数据搜索与优化功能提供很好的支持，能够进行大规模的数据查询，规模可上升至千万条数据。

⑨ 由C语言和C++编写，并且经过多种环境下的多种编译器测试，MySQL的系统移植性非常强大。

（3） MySQL的应用

MySQL作为关系型数据库，有着其缺点和优势，与ORACLE、SQL Server等数据库相比较而言，可以说小规模成为了一把双刃剑，一方面能够轻易地布置到服务器上，节省空间提高效率，另外一方面功能十分受限制，功能效率自然下降。但是以上的缺点并没有使得其热度减少，因为对于大多数企业来说，MySQL提供的功能已经完全满足了用户的需求，而且其精简的特点使得很多小型企业倍加推崇，另外一个受用户欢迎的也是它开源的特点，正因为开源才能够大大降低布置数据库的成本。 在Linux服务器上，当下被整个网络上使用的建站方式为LAMP，也就是使用Linux、Apache、MySQL和PHP搭建网站。具体含义为使用Linux作为网站的操作系统，Web服务器使用Apache，数据库管理系统使用MySQL，服务器脚本使用PHP语言。而且这四个都为开源产品，真正可以达到成本为零的网站搭建[4]。

（4） MySQL管理

首先MySQL可以通过命令行进行控制，在MySQL的二进制文件中提供了命令MySQL；其次，MySQL还提供了多种图形界面化工具，可以去MySQL官网上进行下载，其中包括MySQL Administrator、MySQL Query Browser和MySQL Workbench等；除了服务器上的管理之外，MySQL还有Web管理工具，其中phpMyAdmin就是最直接的代表，phpMyAdmin是由PHP写的用于管理MySQL的Web界面工具，让用户可以直接使用其进行管理数据库系统，包括增删改查，以及备份恢复等所有数据库操作。

## 2.2 C#相关技术

C#是由微软开发的一种面向对象的语言，具有精确简单、类型安全等特点。其中C#和.Net之间关系密切，.Net和C#之间类似于集合和元素之间的关系，按照微软总裁兼首席执行官的说法：.Net代表一种环境，它提供了良好的编程平台，其中.Net中最具有代表性的语言就是C#。

C#主要具有的特征有：

（1） 简洁的语法，C#在使用时比其他语言相对简洁易读；

（2） 完全地面向对象设计；

（3） 与Web紧密结合，提供网站设计技术；

（4） 完全的安全性与错误处理，通过.Net保证代码安全，对于错误处理也要更加完善；

（5） 版本处理技术；

（6） 灵活性和兼容性，不同版本之间无论是向上还是乡下兼容都做得非常好。

使用.Net框架主要的优点有：

（1） 一致的编程模式；

（2） 一次运行就会永远运行；

（3） 一次编写，一次构造，处处可以运行；

（4） 跨语言集成；

（5） 内存和资源的自动管理；

（6） 一致的异常处理；

（7） 发布；

（8） 安全。

常常有人喜欢将C#和Java进行比较，因为两种语言之间有很多共同点，比如同样都是面向对象的语言。但是二者在数据类型、继承、接口、内存管理和源代码编译等等地方存在着相当大的差别，也正是由于微软的强大，才使得后开发出来的C#能够和Java鼎足而立，流行起来，在Windows平台上，可以说C#的使用比Java拥有更多的好处和优点，从界面绘制、网络编程都不逊色。而且由于其与C++和Java相近的特点，开发人员可以很容易的理解和运用。

## 2.3 相关术语定义

（1） TPC：全称为Transaction Processing Performance Council，翻译成中文即事务处理性能委员会，是由数十家会员公司建立的一个非盈利性的组织，其官方网站地址为 http://www.tpc.org。其目的是制定一系列数据库制定商务应用基准程序的规范，在规范中给出性能指标和价格度量，同时对测试结果进行管理和发布[7]。在整个行业中具有科学性和权威性。TPC并不给出基准程序的代码，而是只给出基准程序的规范，所以说基准程序规范是公开的，并且可以从官方网站上进行下载标准和TPC提供的工具包。所有软硬件厂商可以根据下载的规范进行自己系统的测试，以图达到系统的最优状态，而用户也可以根据基准对厂商的数据指标提出相应的要求，所以为了测试结果的真实准确，测试者必须提交处一份完整的测试报告，同时交由TPC的工作人员审核，通过后即具有权威的效力。

（2） TPC-H：TPC-H基准是由TPC-D（TPC之前制定的标准）进化而来的。它的主要目的是用于评价整个数据库系统的特定查询的支持能力[8]，特别强调服务器在数据挖掘和数据分析处理的能力，数据库查询可以说是数据库必不可少的操作之一，也是决策支持的最主要应用之一，而查询一般分为两种类型，第一是预先知道的查询，如定期定时的工作报表[9]，第二种是无法提前知晓的查询，也就是动态查询，比如用户的临时操作。TPC-H使用每小时执行的查询数（TPC-H [QphH@size](mailto:QphH@size)）作为度量单位，H的含义即为每小时执行查询的平均次数，而size表示整个数据库的大小。同时在整个测试中还要满足ACID的事务特性，这也是由于其模拟的环境来决定的。在整个TPC-H标准中由10章正文和5个附录组成。

（3） 事务（Transaction）：事务是可以访问并可能对数据库进行一系列操作的程序执行单元。通常用高级语言，例如SQL、C++、Java等来进行书写，而一个事务也是一个最基本的数据库操作，拥有不可分割的特点，一个事务需要具有ACID特性。

（4） 原子性（Atomicity）：事务作为基本的数据库操作单元，它所执行的数据库操作要么全部执行，要么全部不执行，这种特性即为原子性。

（5） 一致性（Consistency）：在事务执行之前和执行之后数据库始终要保持一致性的状态。所谓一致性的状态，比如说当事务成功提交之后，会从原来的一致性状态变到现在的一致性状态，那么就说明该事务符合一致性，如果在运行事务时系统发生故障，导致该事务的一部分操作语句写入了数据库，这时数据库的内容处于不对的状态，也就是不一致的状态，那么说明该事务不符合一致性。

（6） 隔离性（Isolation）：隔离性指的是事务之间是互不干扰，相互分离的，也就是事务内部的数据操作不能被其他事务所干扰，事务之间彼此封锁消息，提供安全保证。

（7） 持久性（Durability）：持久性代表着当系统发生故障，必须保证已经提交的事务不能丢失他的效果，整个数据库系统要保证该事务对数据库数据的操作时永久的，也就是说要求数据库系统拥有恢复的性能。

（8） ODBC接口规范：英文为Open Database Connectivity，翻译成中文也就是开放数据库连接，它是微软公司开放服务结构中在数据库操作上的组成部分。它所建立的一套规范，在规范中提供了可以对数据库进行直接访问的API接口，通过这些接口就可以直接对数据库进行操作，进而避免了不同类型数据库操作差异大的麻烦，ODBC支持SQL语句，它对于统一接口有着重要的作用[7]。

# 3 数据库测试系统需求分析

## 3.1 功能性需求分析

### 3.1.1 系统流程分析

数据库测试系统整体的功能是，处理数据库连接，配置连接数据库的基本信息，连接数据库后对数据库进行一系列删除数据、建立数据表、建立主键外键索引、导入新生成数据，执行Power测试、Throughput测试、ACID测试、保存日志、查看CPU占用率和IO占用率等，最终通过表格来进行结果的展示，让用户能够直观清晰的理解数据含义。

整个执行测试的流程如图3.1所示。



图3.1 测试系统执行流程图

（1） 根据待测数据库所在服务器信息，使用ODBC连接数据库，配置数据库的基本信息，包括端口号、IP地址、用户名和密码等，可以通过远程进行连接，还可以多台机器进行连接；

（2） 连接成功后，通过工具观察是否数据库中存在残余的数据表，如果存在则先进行数据表数据删除、再删除数据表结构，删除完成后重新构建表结构，通过DBGEN产生数据，导入表中；

（3） 如果数据导入成功，那么就可以进行性能测试和ACID测试，如果导入失败，那么需要将失败这次的数据全部删除，重新进行建表导入数据；

（4） 性能测试主要是Power测试和Throughput测试，根据用户需求也可以进行单条查询语句的时间测试，包括RF1和RF2测试，最后会得到相关的测试结果，可以通过界面观看，并且可以保存日志，生成报告；

（5） ACID测试主要是进行原子性、一致性、隔离性和持久性测试，根据TPC-H的官方标准，对数据表中的数据进行判断，来测试数据库是否满足以上特性，进而得出结论，并且可以保存日志，生成报告；

（6） 在测试的同时可以实时监测CPU占用率和IO占用率，对数据库的变化进行实时监控，观察不同查询语句对数据库的影响。

### 3.1.2 系统用例图



图3.2 测试系统用例图

数据库测试系统的主要使用人员为测试工作人员，测试者根据实际的数据库情况来进行相应的操作，对于其操作和目的在系统上进行了不同模块的划分。

测试者主要进行的操作有连接数据库、配置数据库、执行性能测试、执行ACID测试和保存报告，整个系统的用例图如图3.2所示。

### 3.1.3 系统结构分析



图3.3 测试系统结构图

整个测试系统的结构图如图3.3所示。

（1） 连接数据库

主要操作为查看数据库信息和更改数据库信息，这里可以配置远程连接还是本机连接，如果是远程连接则需要输入相应的IP地址，本机则是127.0.0.1。

（2） 配置数据库

主要操作为通过ODBC删除旧数据、建立数据表、生成新数据和导入数据，支持MySQL数据库的操作。

（3） 性能测试

性能测试提供单条语句的执行测试时间，支持RF1和RF2更新流的测试，测试一为Power测试，测试二为Throughput测试，测试的同时可以查看日志和测试时间。

（4） ACID测试

主要包括原子性测试、一致性测试、隔离性测试、持久性测试，测试结束后能够看到完整的测试报告，并对测试报告进行分析。

（5） 保存报告

对测试结果进行保存，支持多种格式，并且可以查看CPU占用率和IO占用率波形图，对于测试过程中系统资源的利用进行详细分析。

## 3.2 非功能性需求分析

由于系统是C/S架构的，并且要求在Windows系统上执行，所以在需求分析上除了功能之外，还考虑了开发效率、经济成本、软件安全三个方面。

（1） 开发效率：由于软件在Windows系统上执行，所以选择了C#语言进行开发，效率要更高一些，另外虽然软件是C/S架构，但是去掉了Socket层，与三层结构比较而言，两层结构更直接，而且更符合TPC-H的要求标准，无论和C/S三层结构还是B/S结构比较，都减少了中间层的编程，提高开发效率。

（2） 经济成本：整个软件的大部分逻辑结构都在客户端，减少了服务器的压力，让服务器专注于进行数据库的操作，几乎所有的CPU和IO占用都是被数据库占用的，大大提高了数据库处理数据的效率，从经济成本上也进行了一定的节约。

（3） 软件安全：由于软件采用了两层架构，减少了可能出现的中间层错误，并且也排除了在非用户操作端和数据库端可能的人为数据操作，为整个数据库测试提供了公平公正的环境，整体软件安全得到了提高。

# 4 数据库测试系统设计

## 4.1 TPC-H基准分析

### 4.1.1 TPC-H测试模型介绍

TPC-H测试由一系列商业查询组成，这些查询代表在复杂商业应用中的实际数据库操作，但TPC-H标准并没有指定属于某个特定商业领域，而是适用于任何在全球范围内需要销售与管理的行业（如汽车租赁、食品销售等）。基于该模型的测试根本目的不在于待测系统复杂度的还原，而在于对提取待测系统最根本的特性，测试出系统的利用率和操作复杂度。所以在测试标准中的查询和更新操作的目的在于占用系统资源，且具有复杂、访问模式多样化、随机性等特性，并为相应系统查询出定价和销售、供货和需求管理、利润和收入管理等信息，如图4.1所示。

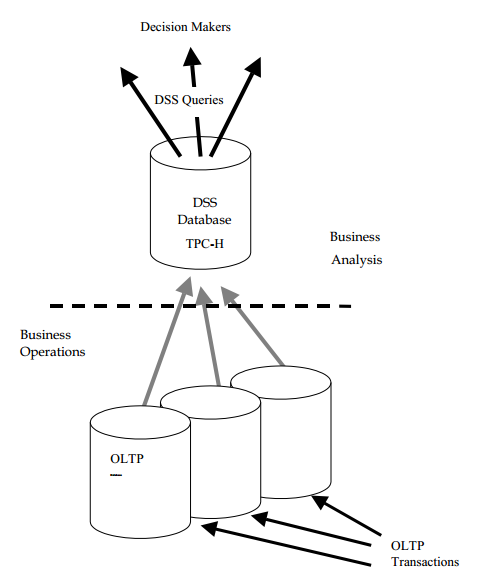


图4.1 TPC-H测试模型

### 4.1.2 数据库实体与关系

TPC-H数据库的组成被定义为由八个单独的表（基本表）组成，该实体-关系结构由TPC-H官方测试文档指定。这些表的列与列之间的关系如图4.2所示：



图4.2 数据库E-R图

### 4.1.3 TPC-H测试工具包

测试工具包可以生成DBGEN和QGEN两个可执行文件。DBGEN用于产生被加载的测试数据，通过工程生成可执行文件之后，使用命令行即可产生不同数量级和状态的数据。QGEN根据模板产生测试需要的22条查询语句，以及RF1和RF2两个更新流操作[10]。



图4.3 工具包结构

模块之间的关系如图4.3所示。

整个测试工具包主要可以分为以下几个模块：

（1） 用于产生数据表和更新数据的DBGEN可执行文件。

DBGEN工具的主要功能是根据比例因子用来产生8张数据表文件和更新流数据的，数据表文件是整个测试的起始部分，拥有合理的数据才能进行下一步的测试，所以在测试之前，需要用数据库语句或者特定的数据库工具将数据装载到数据库的表中，而更新流数据主要是在测试过程中用于对orders表和lineitem表进行数据更新和删除操作，模拟真实的数据库环境，在相应的行上面进行数据的修改。

（2） QGEN中给出的22条查询语句模板还有给出的更新函数。

查询语句模板是被QGEN来使用的，其中的语句都是通过TPC-H的官方标准来定义的，之后在某些地方会被QGEN替换掉，以产生真正的可执行语句，而基准中提供了一共22种查询，所以对于每种查询都需要一个特定的模板。更新函数的主要功能是在查询过程中对数据库进行数据的插入和删除操作，在TPC-H的官方文档中做了具体的阐述，文档中对于Power测试，在执行完整的测试之前必须要做插入操作，而执行整个测试之后还必须要做一个删除操作，而对于Throughput测试来说，要在测试的同时进行插入与删除操作，并且查询操作始终在运行中。

（3） 通过22条语句查询模板，来产生不同数据库可执行查询语句的QGEN可执行文件。

QGEN和DBGEN类似，都是使用C语言写的可执行程序，主要用来根据模板生成查询语句，用于TPC-H和TPC-R标准中，可以在Linux和Windows等多种平台使用，主要的操作是利用给定的查询语句模板，进行相关的数据替换，替换后即成为可以执行的查询语句，再将该查询语句使用工具进行数据库操作。

DBGEN和QGEN的命令行调用参数如下所示。

数据生成程序 DBGEN

DBGEN [-{vfFD}] [-O {fhmsv}][-T {pcsoPSOL}] [-s <scale>][-C <procs>][-S <step>]  
DBGEN [-v] [-O {dfhmr}] [-s <scale>] [-U <updates>] [-r <percent>]  
-b <s> -- load distributions for <s>  
-C <n> -- use <n> processes to generate data  
[Under DOS, must be used with -S]  
-D -- do database load in line  
-d <n> -- split deletes between <n> files  
-f -- force. Overwrite existing files  
-F -- generate flat files output  
-i <n> -- split inserts between <n> files  
-n <s> -- inline load into database <s>  
-r <n> -- updates refresh (n/100)% of the data set  
-s <n> -- set Scale Factor (SF) to <n>  
-S <n> -- build the <n>the step of the data/update set

数据生成程序 QGEN  
Options：  
-a -- use ANSI semantics.  
-b <str> -- load distributions from <str>  
-d -- use default substitution values.  
-h -- print this usage summary.  
-i <str> -- use the contents of file <str> to begin a query.  
-l <str> -- log parameters to <str>.  
-n <str> -- connect to database <str>.  
-N -- use default rowcounts and ignore :n directive.  
-o <str> -- set the output file base path to <str>.  
-p <n> -- use the query permutation for stream <n>  
-r <n> -- seed the random number generator with <n>  
-s <n> -- base substitutions on an SF of <n>  
共22个查询模板，以1.sql为例：  
sql: select count(\*) from lineitem where l\_orderdate < ':1';  
QGEN 1: select count(\*) from lineitem where l\_orderdate < '1997-01-01';  
批量产生22个查询：  
QGEN.exe -c -s %d -p %d

### 4.1.4 TPC-H度量指标

TPC-H的度量指标全部与测试执行查询时间相关，相关的时间主要有以下三个：装载数据时间，每个查询执行时间和更新操作时间[11]。通过对以上时间的测量可以计算出：数据装载时间、Power @ Size，Throughput @ Size，QphH @ Size和￥/QphH @ Size，以下讲对这几个指标的具体含义进行解释说明。

（1） 装载数据时间

在整个装载数据的过程中，大部分时间都要统计其中，其中人为的手动操作不能计入总的装载数据时间中，而像建表、添加主键、添加外键、删除表数据、导入表数据、更新表数据、删除表数据和建立索引等，这些时间必须进行计算，也就是最后的装载数据时间。

（2） 查询和更新时间

在Power测试和Throughput测试中所有查询和更新时间必须被测量和记录[12]，每个查询的时间统计方法是这样的，从开始执行查询语句开始，到获得查询结果的最后的字符，此为一次查询的时间，而更新操作RF1和RF2定义类似，从提交开始到完成数据库操作的总时间。

（3） Power @ Size

该数据是Power测试的指标，在官方文档给出的定义为：总查询和更新时间的几何平均数，几何平均数再取倒数即为该指标，计算公式如下：

（4.1）

公式中的参数含义为：数据大小Size，决定数据规模的比例因子SF；第i个查询时间QI(i,0)；RFj更新时间RI(j,0)。

（4） Throughput @ Size

该数据是Throughput测试的指标，在官方文档给出的定义为：所有查询执行的总时间的平均值，然后再对平均值取倒数，计算公式如下[13]：

（4.2）

公式中的参数含义为：数据大小Size；查询流个数S；查询和更新总执行时间Ts为；决定数据规模的比例因子SF。

（5） QphH @ Size

该数据是每小时查询数(Query-per-Hour)，由Power测试和Throughput测试的两个结果组合而成，计算公式如下[14]：

（4.3）

该指标反应了待测数据库系统在某种数据规模下数据处理的综合能力。

（6） Price-per-QphH @ Size

该数据表示性能价格比，它是最后不同数据库系统比较的指标，综合评价了某个系统的花销与其性能的关系，计算公式如下：

（4.4）

公式中的参数含义为：系统运行总费用￥，该费用包括所有购置设备费用，安装软件费用，维护和技术支持费用等。

## 4.2 TPC-H Driver整体软件架构

### 4.2.1 软件的体系架构

下图所示，TPC-H Driver的体系架构符合标准中提供的测试结构，其中包括两种结构，第一种是查询执行和数据库访问都在同一台主机上，即驱动器在待测系统上，由待测系统提供硬件和软件支持；第二种结构为C/S架构，执行查询操作的驱动器（Driver）在客户端，数据库访问在服务器端，二者通过网络进行连接。

其中需要开发者实现的部分主要有以下几点：

（1） 查询执行驱动器：实现查询和更新操作，以达到Power、Throughput测试的目的；

（2） 数据库逻辑：实现存储过程、查询语句和更新语句的生成等；

（3） 报告与测试信息：将测试结果生成报告并在测试过程中显示测试信息。

整个结构如图4.4所示。

### 4.2.2 软件的逻辑架构

如下图介绍了软件的整体逻辑结构，该结构严格按照TPC-H规定的架构来设计，并且为便于用户使用扩展了很多新的功能。

下面是各个模块的功能简介：

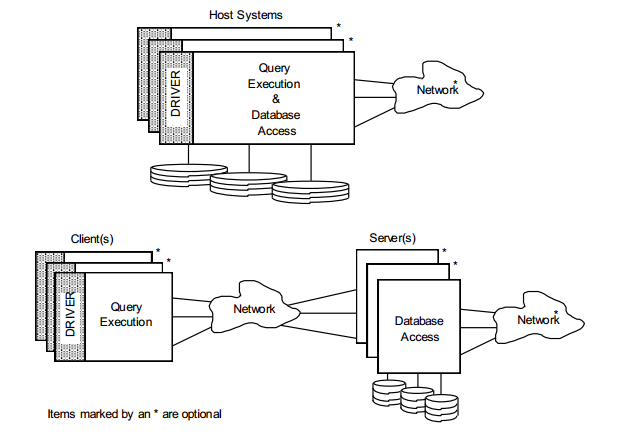


图4.4 软件体系架构

（1） 主控模块

主控模块中提供了所有其他功能的入口，测试的开始与结束都由该模块来控制，响应用户请求并提供用户接口，驱动相应的模块完成相应的功能。

（2） 数据库配置模块

配置待测系统的数据库参数，为数据库连接做好准备。

（3） 生成数据模块

生成数据模块根据DBGEN的路径和比例因子生成负载数据。

（4） 加载数据模块

加载数据模块提供了清除表、创建表、加载数据、添加主键和添加外键的功能。

（5） Power测试模块

进行Power测试同时在测试结束后返回测试结果。

（6） Throughput测试模块

进行Throughput测试同时在测试结束后返回测试结果。

（7） ACID测试模块

进行原子性，一致性，隔离性与持久性等一共10个测试。

（8） 日志和报告模块

主要功能为显示测试过程中的测试信息和保存测试结果的报告。

整个软件的逻辑架构如图4.5所示。



图4.5 软件逻辑架构

## 4.3 TPC-H Driver软件功能设计

### 4.3.1 主控与日志

主控模块的是所有其他功能的入口，初始化测试环境与用户进行交互都是该模块的作用。当用户进行不同的操作请求时能够及时响应，并给予正确的回复，主要的操作有新建测试、数据库配置、数据处理、请求测试等。

日志模块主要包括测试日志和生成报告，测试日志是测试信息的收集器，负责将测试信息采集过来并反馈给用户，其中测试信息包括：数据库连接情况、数据删除和创建情况、测试情况以及失败后的提示信息等等。生成报告主要针对测试后报告的内容进行显示和存储。

TPC-H Driver通过构建实时监控模型来实现该模块，通过构造一个信息存储单元来保存用户请求数据和测试返回数据，并通过一定的时间间隔来显示测试结果，其模型如图4.6所示。

从图4.6可看出，excel模块功能比较简单，主要用于实现对数据库进行查询后返回的结果保存为EXCEL格式的文件。

Msg与Log两个模块整体实现了软件的日记记录功能，其中Msg模块功能主要是定义消息类型，通过Log模块将生成消息类型记录到日志中。



图4.6 主控模块与日志流程

主窗体模块功能是控制这个TPC-H测试流程，也是用户能够接触使用的模块，用户通过该界面的一系列操作就可以实现对数据库的一系列测试，该模块在测试过程中通过调用各个功能类文件中的方法实现保存测试过程中产生的数据以及日志记录功能。

### 4.3.2 数据库配置与数据装载测试

数据库连接可以通过远程或者在同一台机器上部署来完成，配置信息通过主控模块利用ODBC接口连接，统一数据库连接方式。

在连接成功后，首先需要按照比例因子通过命令行调用DBGEN，进而生成装载数据的八张表，数据装载之前，需要选择数据库中的一个schema来对其进行操作，选中后需要清空里面的数据和表内容，重新根据TPC-H标准构建表结构，添加主键和外键并导入刚生成的数据，以上操作需要按照顺序执行。

从图4.7可看出，该模块是软件运行的基础，其中定义了各个功能模块使用的SQL语句，包括数据库建表，装载数据，查询等一整套程序使用的语句。并且针对不同的数据库，可以通过定义一整套不同的数据库语言的方法，来扩充该软件以支持的不同版本的数据库。

该模块包含对数据进行基本操作的基本功能，其中包括对数据进行基本操作的各种静态方法，包括连接数据库，建表，删除数据库等操作，测试过程中装载数据阶段通过对该功能文件中方法的调用实现数据库建立、数据加载、添加主键和外键等一系列前期准备操作。



图4.7 数据库操作流程图

除此之外，该模块功能还包括按照TPC-H标准进行设计的Power测试阶段需要调用的功能方法以及一对更新流操作。

### 4.3.3 Power与Throughput测试

TPC-H标准测试就22条查询语句进行，每个查询语句严格定义，严格遵循SQL语法，用户无法修改。标准中定义的查询语句复杂程度较高，超过实际应用的场景，一个查询语句通常执行时间可以达到几十秒甚至十几小时，所以如果数据库规模较大的话，那么22条查询语句执行一遍的时间甚至需要十几个小时。

Power测试是无序执行22条查询语句和一对更新流操作，Throughput测试要求开启多个查询流，每个查询流执行22条查询语句和一对更新流操作。

Power测试在刚刚装载完数据后，数据库处于初始状态，将22个查询语句执行一遍，同时伴随着一对更新操作的执行；最后进行Throughput测试，也是最复杂和对SUT系统压力最大的测试，在有多个查询流执行的同时，还有一对更新流[15]。

2个更新操作RF1和RF2是为了更真实的模拟商业环境，它们在查询语句执行过程中并发执行。其中RF1向Order和Lineitem表中插入占原行数0.1%的新行；RF2则从Order和Lineitem表中删除占原行数0.1%等量的数据。这两个操作模拟了数据新旧数据动态变化的过程。



图4.8 测试流程图

如图4.8所示，Power测试模块，显示的是Power测试过程中，每条查询语句运行的查询结果，同时显示每条查询语句运行的时间以及整个power测试过程消耗的总时间，同时可以实现对查询结果的保存。Throughput测试模块，显示的是Throughput测试过程中，每个查询流中查询语句的开始时间与结束时间，并且在后台进行保存，用于最终生成测试报告。该模块中定义了Throughput测试阶段每个工作流执行的一整套操作需要调用的方法，在进行Throughput测试的过程中，按照TPC-H标准中的规定，通过开启新的线程来执行该文件中定义的一整套方法来实现，并且记录保存下每个工作流的运行时间用于生成最终的报告文件。

### 4.3.4 ACID测试

ACID测试主要是针对数据库的原子性、一致性、隔离性以及持久性进行测试。ACID特性对DBMS处理事务的能力有较大的影响，同时也是保证数据库正确运行的保证，如果不能保证ACID特性，则DBMS可能会显示出虚假的高性能指标。

该模块中按照TPC-H文件规范中的要求，主要功能用于实现ACID测试阶段的每个属性的测试流程以及最终条件的判断。

其中包括ACID测试准备阶段需要建立HISTORY表等准备操作以及测试阶段中原子性、一致性、隔离性以及持久性测试中流程的实现以及测试结果是否正确的判定过程。

## 4.4 本章小结

本章侧重于简述数据库测试的整体布局。

首先分析TPC-H的官方标准，适用于多数销售与管理行业的商业模型，从实体与关系中能够更具体的看出来数据模型的建立过程，并对TPC-H标准下所提供的官方工具DBGEN进行了目录分析和功能介绍。

其次描述了软件设计的整体目标，并介绍了最主要的测试指标，对软件的整体功能有一个全局的把握。

接下来从目标展开分别介绍了软件的体系架构和逻辑架构，对标准和实现进行进一步的探讨。

最后从逻辑架构入手，在软件概要设计中进行了逐个模块的分析，介绍了主控与测试信息模块、数据库配置与数据装载模块、Power与Throughput测试模块和ACID测试模块。

# 5 数据库测试系统实现

## 5.1 程序实现原则

TPC-H Driver对于多种数据库的支持，在测试性能和ACID上必须考虑测试兼容性的问题，在程序内部需要对多种数据库进行测试和差异处理，这就对执行测试的数据库语句和进行数据库连接提出了要求，针对以上问题，对于整个软件的实现提出了如下原则：

（1） 软件设计符合TPC-H标准

软件的设计首先从整体上要满足界面友好，便于用户理解和操作的需求，与此同时测试出数据库的性能真实值，也就是说符合TPC-H标准，但由于软硬件差异，在实现过程中并不能完全适应标准要求，所以要更多的通过软件编程方式接近标准，同时增加测试系统的真实性和标准型。

（2） 软件运行在Windows平台

由于需求中要求软件运行在Windows平台上，所以虽然 JAVA 的跨平台能力要比C#强，但在Windows编程上还是C#优势更加明显，故而从实现角度最终采用C#语言，并使用ODBC接口。同时由于要求测试软件能够对多种数据库进行测试，所以要能够兼容 32位操作系统和64位操作系统、不应使用数据长度可变的数据类型如长整型long。

（3） 用户交互友好

在拥有标准和数据库的情况下，完全可以不通过软件来进行测试，之所以使用测试软件是为了避免用户操作繁琐和不当导致测试结果不准确，对于用户来说进行测试简化，所以与用户交互友好是软件实现的基本要求，在软件实现的过程中无论从数据库连接，性能测试还是ACID测试都力求用户根据手册能够很快使用，操作简单，并且获得可信结果。

（4） 测试系统可扩展性强

从测试多种数据库以及增加测试功能的角度，需要软件拥有较强的可扩展性，保证软件在实现过程中不会因为扩展性低而产生大量无意义的工作，可扩展性强同时可以加快编程效率，为后续软件维护做好准备。

（5） 测试结果阅读便利

测试结果对于用户来说是直接体现数据库性能的地方，所以对于结果的处理至关重要，针对结果首先要做的是让用户能够简单清晰地阅读到，其次就是对结果进行一定的分析，让用户更清晰地知道结果所代表的含义，最后就是某次的测试结果应该能够反复观察和阅读，便于用户对于多次测试结果的比较和分析。

## 5.2 主控与日志实现

主控模块是TPC-H Driver的核心控制单元，在用户交互上具有重要意义，在实现上采用了面向对象的方法，整个类图如图5.1所示，其中MainFrame为整个测试软件的入口界面，为用户提供了交互接口，同时里面包含了4个TabPage控件，提供与其他4个界面的交互。TabPageConfiguration为待测数据库连接界面，提供了许多配置选项，包括数据库类型、用户名、密码、服务器、端口号以及选择待测库名称。TabPageOperation为操作数据界面，主要针对数据库进行操作，使用DBGEN程序生成装载数据，存储在本地文件中，再用DBGEN生成刷新数据，其中RF1为插入ORDERS和LINEITEM数据，RF2为删除ORDERS和LINEITEM数据，同时用QGEN程序，调用SQL模板转换成真正可执行的SQL，同时还要完成建表、添加主键、外键、导入数据等功能。TabPageFunctiontest为性能测试界面，主要功能包括进行Power测试、Throughput测试、RF1和RF2更新流以及保存报告、显示测试结果的功能。TabPageACIDtest为ACID测试界面，主要进行2个原子性测试、1个一致性测试、6个隔离性测试和1个持久性测试。



图5.1 主控模块类图

主控模块是所有其他模块的枢纽，负责响应用户的操作请求，建立用户自定义的测试环境，并根据用户操作销毁测试环境，还有建立多线程操作，主要作用是防止主线程假死，干扰用户操作，导致体验变差。根据不同的用户任务，主控模块启动的有以下线程：主线程，启动线程，测试监控线程，数据库连接线程，生成数据线程，数据库操作线程，Power测试线程，Throughput测试线程，ACID测试线程，保存报告线程。

图5.2 线程启动图

其中主线程的含义为主控模块最先启动的线程，也是通过该线程完成测试环境的初始化，测试UI的绘制，并且响应用户的操作请求。启动线程由主线程启动，用来启动其他线程。测试监控线程用于监视测试是否完成，进而判断是否给予用户相应的测试信息。数据库连接线程用于读取配置信息，连接数据库不干扰主线程。生成数据线程读取DBGEN路径和比例因子或者更新流数目，通过新启动进程来完成数据生成。数据库操作线程是读取数据文件路径，通过ODBC执行事先写好的数据库脚本，其中加载数据的操作需要较长时间，所以需要数据库操作线程来执行，否则主线程会出现假死现象。Power测试线程是在进行Power测试时启动的，其中包括两部分，分别是执行Power测试和读取测试结果到TabPageFunctiontest性能测试界面。Throughput测试线程是在进行Throughput测试时启动的，其中包括两部分，分别是执行Throughput测试和读取测试结果到TabPageFunctiontest性能测试界面。ACID测试线程包括10个线程，是在进行不同测试时分别启动的，主要工作是执行ACID测试和返回测试结果到TabPageACIDtest测试界面。保存报告线程用来保存各种测试结果，根据输入的不同保存的内容也不相同。图5.2描述了主控模块对于线程的处理。

在测试软件启动后会启动主线程，主线程向其他界面发送使能信号，其他界面则根据使能新号进行显示，启动线程工作，之后会根据用户的输入来进行相应的处理。

测试监控线程是监视测试与用户交互的重要线程，其处理流程需要进行说明，保存测试信息的单元SaveStatus在该线程启动时进行初始化，并且该单元由所有测试有关线程和监控线程共有。测试相关线程测试结束后会更新SaveStatus，监控线程以1秒1次的时间频率对SaveStatus进行读取，一旦读取到新更新的内容即将其显示到主控界面相应位置中，并再次将信息单元清空。

## 5.3 数据库操作实现

数据库操作分为三个部分，分别是数据库连接配置、数据生成、数据加载。

### 5.3.1 数据库连接配置

数据库连接配置指的是需要用户在测试过程中需要用户自定义的信息，配置过程在主控模块启动之后，由用户主观进行操作。TPC-H Driver的参数配置既可以通过用户手动填写，也可以通过xml文件解析来实现。TPC-H Driver启动时配置数据全部被清空，配置信息同样为空，为了简便用户操作，提高软件效率，提供了xml文件读取的方式，初始化全部的测试参数。会读取或创建指定的配置文件，初始化各测试参数。除了支持读取xml文件，同样支持保存xml文件，方便之后读取本次的配置信息。

<tpch>

<cbxType\_S>MySQL</cbxType\_S>

<txbxServer\_S>localhost</txbxServer\_S>

<txbxPort\_S>3306</txbxPort\_S>

<txbxUser\_S>root</txbxUser\_S>

<txbxPassword\_S>12345678</txbxPassword\_S>

<txbxDbgen>D:\TPCH\dbgen.exe</txbxDbgen>

</tpch>

### 5.3.2 数据生成

数据生成分为两个部分，生成负载数据和更新数据。虽然都是调用DBGEN产生表数据，但是调用参数不同，各有一个生成线程，生成数据在DBGEN所在的目录下面。

其中DBGEN产生的负载数据8张表有以下特点：

在TPC-H标准中SF表示scale factor，即规模因子的含义。如果数据库大小为1G，则SF=1。

（1） SUPPLIER数据表：共计SF\*10000行；

（2） PART数据表：共计SF\*200000行；

（3） PARTSUPP数据表：由于PART数据表中的1行数据对应PARTSUPP数据表中的4行数据。所以共计SF\*200000\*4行；

（4） CUSTOMER数据表：共计SF\*150000行；

（5） ORDERS数据表：由于CUSTOMER数据表中的1行数据对应ORDERS数据表中的4行数据。所以共计SF\*150000\*4行。由于ORDERS数据表和LINEITEM数据表中只有一部分数据，存在大量空白，在32行里只有前8行有数据，后24行是没有数据的，数据占用率只有25%。这两张表这样设计是为了更新流RF1插入数据；

（6） LINEITEM数据表：对于ORDERS数据表中的每一行，在LINEITEM数据表中有1到7行与之对应。所以共计SF\*1500000\*4\*RANDOM\*7行；

（7） NATION数据表：共计25行；

（8） REGION数据表：共计5行。

使用DBGEN产生的更新数据有两种，分别为RF1和RF2。

RF1是新订单更新函数，该函数的功能为创建新订单，并将订单插入到数据库中。所以RF1会向ORDERS数据表和 LINEITEM数据表中插入数据，数据大小与SF有关。该函数的定义如下：

LOOP (SF \* 1500) TIMES

INSERT a new row into the ORDERS table

LOOP RANDOM(1, 7) TIMES

INSERT a new row into the LINEITEM table

END LOOP

END LOOP

更新数据由DBGEN用-U和-s选项来生成。

RF1更新函数的实现用SQL表示出来如下：

INSERT INTO LINEITEM (SELECT \* from TMP\_LINEITEM$SET\_NUM)

INSERT INTO ORDERS (SELECT \* from TMP\_ORDERS$SET\_NUM)

RF2是旧订单更新函数，该函数的功能为从数据库中查询到旧订单并将其删除。所以和RF1操作表相同，从ORDERS和LINEITEM数据表中删除旧订单所在行。该函数定义如下：

LOOP (SF \* 1500) TIMES

DELETE FROM ORDERS WHERE O\_ORDERKEY = [value]

DELETE FROM LINEITEM WHERE L\_ORDERKEY = [value]

END LOOP

更新数据由DBGEN用-U选项来生成。

RF2更新函数的实现用SQL表示出来如下：

DELETE FROM LINEITEM WHERE L\_ORDERKEY IN (SELECT \* FROM TMP\_ORDERKEY$SET\_NUM)

DELETE FROM ORDERS WHERE O\_ORDERKEY IN (SELECT \* FROM t TMP\_ORDERKEY$SET\_NUM)

### 5.3.3 数据加载

数据装载主要指的是五个部分，删除表、创建表、添加主键、添加外键和加载表数据，目标就是实现测试库的初始化。

搭建过程主要包括建表、添加主键、添加外键和导入数据，这些操作都可以在客户端驱动器TPC-H Driver下通过ODBC接口操作完成，下面用实际数据库语句进行解释，以Nation表为例。

（1） 测试库建表

建表要注意与标准中字段名、数据类型、约束条件保持一致。

CREATE TABLE NATION

( N\_NATIONKEY INTEGER NOT NULL,

N\_NAME CHAR(25) NOT NULL,

N\_REGIONKEY INTEGER NOT NULL,

N\_COMMENT VARCHAR(152));

（2） 测试库添加主键

ALTER TABLE NATION

ADD PRIMARY KEY (N\_NATIONKEY),

ADD KEY (N\_REGIONKEY);

（3） 测试库添加外键

ALTER TABLE NATION

ADD FOREIGN KEY NATION\_FK1 (N\_REGIONKEY) references REGION (R\_REGIONKEY);

（4） 导入数据

导入数据即为DBGEN根据比例因子生成的表，以在命令行生成1G规模的数据量为例。

>E:\dbgen\dbgen -fF -s 1

之后即可将刚生成的数据导入已经建立好表的测试库中，如果数据量较大时间也会相应延长。

"load data local infile \"E:/dbgen/nation.tbl\" into table part fields terminated by \"|\" lines terminated by \"\r\n\"";

## 5.4 性能测试实现

### 5.4.1 Power测试实现

Power 测试的步骤：

（1） 从点击Power测试开始记录开始时间t1；

（2） 在Power测试前执行RF1，记录RF1开始时间t2；

（3） 执行RF1插入操作；

（4） RF1执行结束后，记录结束时间t3；

（5） 开始执行全部22条查询；

（6） 执行查询结束后，执行RF2删除操作，记录RF2开始时间t4；

（7） 执行RF2删除操作；

（8） RF2执行结束后，记录结束时间t5；

（9） 全部操作执行结束，记录Power测试结束时间t6。

Power测试将以上时间记录后，即可通过减法计算得到Power测试总时间，RF1执行时间，RF2执行时间，之后要将每一条查询和更新语句所用的时间来进行乘积运算。如果查询时间或者插入删除时间没达到1秒，那么就视为0秒来进行计算，测试结果为0，该结果无意义不被计入。所以为了避免执行时间不到1秒的情况，数据量的选择尤为重要，建议数据量大一些以免以上情况出现。

### 5.4.2 Throughput测试实现

Throughput 测试的步骤：

（1） 从点击Throughput测试开始记录开始时间t1；

（2） Throughput将启动多个查询流；

（3） 同时还有很多并行的更新流操作被启动；

（4） Throughput测试结束，记录测试结束时间t2。

Throughput 测试结果的计算方法使用t2-t1即可，不需要考虑每条语句的查询时间，所以对于数据量的要求并不大，不过建议还是选取一定数据量来进行测试。

## 5.5 ACID测试实现

其中包括ACID测试准备阶段需要建立HISTORY表等准备操作以及测试阶段中原子性、一致性、隔离性以及持久性测试中流程的实现以及测试结果是否正确的判定过程。

未测试ACID，需要事务初始准备方法

create procedure ACID\_Transaction(IN O\_KEY int,IN L\_KEY int,IN delta int)

begin

declare ototal decimal(15,2);

declare quantity decimal(15,2);

declare extprice decimal(15,2);

declare pkey int;

declare skey int;

declare tax decimal(15,2);

declare disc decimal(15,2);

declare rprice decimal(15,2);

declare cost decimal(15,2);

declare new\_extprice decimal(15,2);

declare new\_ototal decimal(15,2);

select O\_TOTALPRICE into ototal from ORDERS where O\_ORDERKEY = O\_KEY;

select l\_quantity,l\_extendedprice,l\_partkey,l\_suppkey,l\_tax,l\_discount into quantity, extprice,pkey,skey,tax,disc from lineitem where L\_ORDERKEY = O\_KEY and L\_LINENUMBER = L\_KEY;

set ototal =ototal-truncate(truncate(extprice\*(1-disc),2)\*(1+tax),2);

set rprice =truncate(extprice/quantity,2);

set cost=truncate(rprice\*delta,2);

set new\_extprice =extprice+cost;

set new\_ototal=truncate(new\_extprice\*(1-disc),2);

set new\_ototal=truncate(new\_ototal\*(1+tax),2);

set new\_ototal=ototal+new\_ototal;

update lineitem set L\_EXTENDEDPRICE = new\_extprice,L\_QUANTITY = quantity + delta where L\_ORDERKEY = o\_key and L\_LINENUMBER = l\_key;

update orders set O\_TOTALPRICE =new\_ototal where o\_orderkey=O\_KEY;

insert into history values(pkey,skey,O\_KEY,L\_KEY,delta,now());

end;

其中十个ACID测试的定义如下：

（1） 原子性测试

测试1：Atomicity Test 1(COMMIT)执行ACID事务，COMMIT并确保ORDERS，LINEITEM，HISTORY相关行的正确性。

测试2：Atomicity Test 2(ROLLBACK)执行ACID事务，ROLLBACK并确保ORDERS，LINEITEM，HISTORY相关行没有改变。

（2） 一致性测试

对于order和对应的lineitem，若满足以下两个变量条件：

DATE= trunc(trunc(L\_EXTENDEDPRICE\*(1-L\_DISCOUNT)2)\* (1+L\_TAX)；

O\_TOTALPRICE=SUM(DATE,2))；

那么可以认为该数据库满足一致性。首先确保ORDERS,LINEITEM的一致性提交至少100个ACID事务再次确保ORDERS,LINEITEM的一致性。

（3） 隔离性测试

测试1（读-写-提交）：执行ACID事务Txn1，在COMMIT前暂停，使用相同O\_KEY 执行ACID查询Txn2，确保Txn2没有读取到Txn1的更新结果（level2，不等待txn1），完成Txn1的执行。

测试2（读-写-回滚）：执行ACID事务Txn1，在COMMIT前暂停，使用相同O\_KEY 执行ACID查询Txn2，确保Txn2没有读取到Txn1的更新结果（level2不等待txn1），回滚Txn1。

测试3（写-写-提交）：执行ACID事务Txn1，在COMMIT前暂停，使用相同O\_KEY 执行ACID事务Txn2，Txn2应该等待（level1），完成Txn1。检查结果是否符合预期：Txn2在Txn1基础上继续执行。

测试4（写-写-回滚）：执行ACID事务Txn1，在COMMIT前暂停，使用相同O\_KEY 执行ACID事务Txn2，Txn2应该等待（level1），回滚Txn1。检查结果是否符合预期：Txn2的结果不受Txn1影响。

测试5（更新事务不影响并行查询）：执行ACID事务Txn1，在COMMIT前暂停，执行可并行的事务Txn2，读取PARTSUPP数据表，确保Txn2先执行完毕。完成Txn1。

测试6（查询不影响更新事务）：执行Q1查询Txn1，完成前执行ACID事务Txn2，Txn1依然执行时再并行执行Q1查询Txn3。确保Txn2在Txn3之前完成，并确保正确变更。

（4） 持久性测试

首先确保ORDERS，LINEITEM的一致性。并行Query Stream Number + 1个session，每个session提交200个ACID事务。在session大概完成100个ACID事务完成后，用户触发一个系统错误。重启SUT进入恢复过程。检查HISTORY中的数据与事务提交情况是否一致，重新检查ORDERS，LINEITEM的一致性。

## 5.6 本章小结

本章主要对整个数据库测试系统的实现进行了详细的介绍。

首先从程序实现原则开始，对于软件的要求提出了详细的分析，包括符合TPC-H标准、运行在Windows平台、用户交互友好、测试系统可扩展性强和测试结果阅读便利。

接下来对整个主控与日志模块的实现进行了详尽的描述，对整个系统的线程启动进行了介绍，并对日志实现做除了具体分析。

在之后对数据库操作实现进行阐述，首先是数据库连接配置，介绍了xml文件解析，之后是数据生成，对DBGEN进行调用，并介绍了八张表的情况和彼此之间的关系，还有RF1和RF2的操作流程。还有数据加载，包括了删除表、创建表、添加主键、添加外键和加载表数据，并以Nation表为例列出了SQL语句。

下一部分是性能测试的实现，详细描述了Power测试和Throughput测试的步骤。

最后是ACID验证测试实现，对于事务准备和ACID测试的标准进行了详细讲解。

# 6 测试结果与分析

## 6.1 测试环境

处理器：CPU Intel Core i5（建议酷睿i5以上）；

内存：4G（建议4G以上）；

网卡：100M以上支持TCP/IP协议的网卡（建议使用千兆网卡）；

硬盘：512G固态硬盘；

操作系统：Windows7 64位操作系统；

.Net 运行时环境：Microsoft .Net Framework 4.5以上。

## 6.2 测试执行流程

TPC-H在标准中描述了测试所需要的八张表，包括主键和外键等重要信息。在测试进行时，针对不同的数据库，建立数据表以及导入数据语句有一定区别，本节以MySQL为例将具体讲解待测系统测试执行流程。

TPC-H Driver的主要测试流程如下：

（1） 数据库和ODBC配置；

（2） 测试数据库的创建；

（3） 运行客户端驱动器，执行性能测试；

（4） 运行客户端驱动器，执行ACID测试。

### 6.2.1 数据库和ODBC配置

（1） 安装数据库并配置好用户名、密码等信息，并允许远程连接，同时启动数据库；

（2） 安装数据库对应的ODBC驱动并建立ODBC数据源；

（3） 创建测试库TPC-H，之后建立数据表、导入数据都在此测试库中进行。

### 6.2.2 测试数据库的创建

首先在连接数据库阶段选择数据库类型，配置连接信息等连接条件，数据库远程连接要求客户端和待测端连接同一网段，在同一个局域网下，使用待测系统的IP地址、用户名和数据库密码进行数据库连接，通过TPC-H Driver连接待测系统数据库，如图6.2所示。

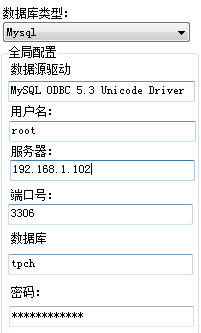


图6.2 初始化界面

数据装载阶段包括生成数据（可以自定义比例因子），创建数据库以及快速加载数据操作，加载数据库成功后，前期准备工作结束，进入测试阶段，如图6.3所示。

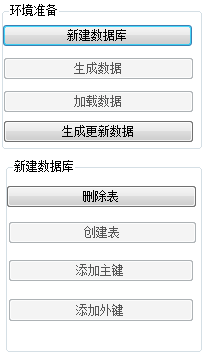


图6.3 数据装载界面

### 6.2.3 执行性能测试

性能测试包括更新流、Power和Throughput测试，测试数据以及过程保存到日志文件中，其中Power测试结果如图6.4所示。

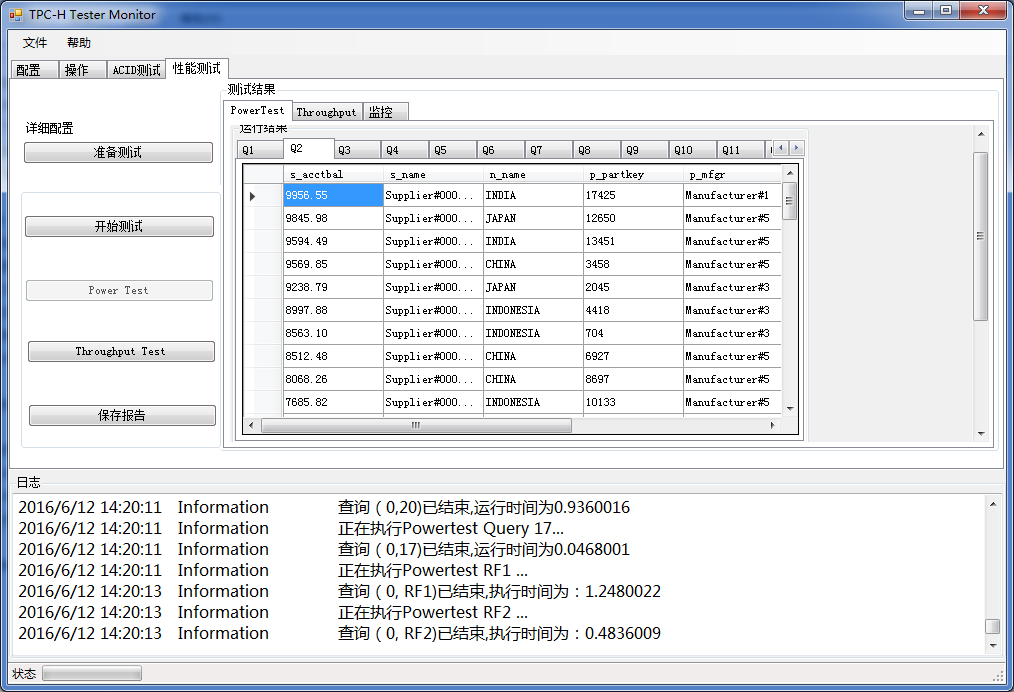


图6.4 Power测试结果

Throughput测试结果如图6.5所示。

### 6.2.4 执行ACID测试

ACID测试阶段主要针对数据原子性，一致性，隔离性和持久性进行测试，测试过程中的相关信息会记录在日志当中，运行客户端驱动器，执行ACID共计十个测试，测试过程中如果该数据库符合标准将会提示测试成功。根据整套测试，生成符合TPC-H标准的测试报告，其中可以选择excel和pdf不同的生成形式，以供用户阅读。

## 6.3 性能测试结果与分析

MySQL的测试比例因子SF大小为1，也就是1G大小的数据，Stream为5，测试结果Power Score为3912.61秒，Throughput Score为5919.04秒，QphH为490.77。

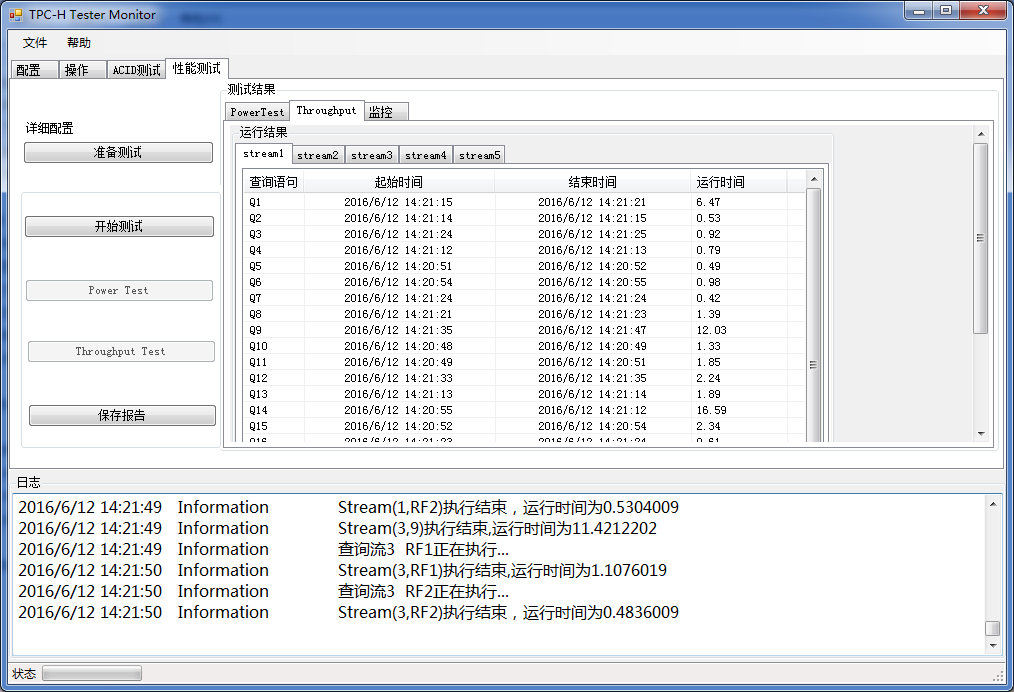


图6.5 Throughput测试结果

从表6.1可以分析出来，第14条测试也就是Q14在查询过程中使用的时间最长，大概占用了测试时间的一大半。

除了Q14之外，Q9、Q10、Q13、Q18、RF1等耗时也较长，其他语句耗时较短。

测试过程中，MySQL内存的使用并没有超过1G，说明内存不是影响测试的因素。

对于在机械硬盘和固态硬盘上测试，发现固态硬盘上性能提升明显，并且磁盘I/O访问一直非常频繁，说明磁盘I/O是TPC-H的主要的影响因素。

## 6.4 ACID验证结果与分析

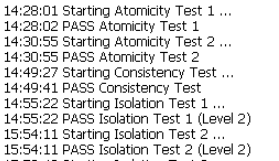


图6.6 ACID测试信息

从MySQL的数据库测试来看，只要数据库正常工作，ACID测试都能够通过，在进行隔离性测试4、5、6的时候，会出现无法通过的情况，主要是由于MySQL事务隔离级别默认为第三级别Repeatable Read（可重读），该数据库的事务隔离级别不足，如果改成第四级别Serializable（可串行化）即可顺利通过事务测试。

表6.1 MySQL性能测试结果表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Query | Power | Stream1 | Stream2 | Stream3 | Stream4 | Stream5 |
| Q1  Q2  Q3  Q4  Q5  Q6  Q7  Q8  Q9  Q10  Q11  Q12  Q13  Q14  Q15  Q16  Q17  Q18  Q19  Q20  Q21  Q22  RF1  RF2 | 24.40  29.74  113.23  9.29  73.75  5.50  41.38  145.05  364.84  304.12  7.99  11.78  260.82  1814.74  31.65  4.36  2.41  279.59  4.63  61.32  24.85  0.60  268.42  28.16 | 34.86  52.78  149.99  13.04  134.44  7.32  61.49  272.64  591.20  524.52  16.22  19.83  496.67  1669.82  15.47  9.43  3.28  456.29  4.83  89.73  37.46  1.13  1306.46  25.67 | 42.19  54.09  150.00  13.05  131.28  7.23  61.16  265.14  575.83  560.18  14.09  19.47  466.92  1669.82  16.09  12.55  2.97  456.29  4.83  82.25  52.95  1.06  1292.08  36.23 | 35.28  52.14  163.80  18.08  129.83  7.50  56.44  267.24  495.38  566.10  10.25  20.26  470.98  1669.82  12.98  11.60  3.74  494.49  4.83  86.69  50.75  1.49  1294.73  36．10 | 43.48  52.35  163.11  13.05  125.34  6.41  59.83  240.32  596.48  530.86  15.06  18.20  509.74  1669.82  15.48  9.68  2.91  443.20  4.83  91.11  48.11  1.11  1273.35  41.90 | 32.89  41.34  160.65  18.77  133.98  6.57  54.27  275.30  498.35  548.07  12.50  16.92  474.03  1669.82  18.12  11.18  3.47  507.23  7.74  78.19  55.02  1.48  1267.52  22.82 |

## 6.5 本章小结

本章首先从测试环境入手，提供了测试的最低标准，保证数据库测试能够成功执行，其中对处理器、内存、网卡以及.Net运行环境提出了明确的要求。接下来描述了整个测试执行的流程，从测试数据库的创建到运行客户端驱动器、执行性能测试和ACID测试，并展示了整体的UI设计，对软件的整体有更真实的描述。然后又对性能测试结果进行了展示和分析，说明某系统下数据库在性能上的强弱。最后对ACID验证结果进行了分析，表明为什么数据库ACID测试通过和不通过的原因。

# 结 论

本文在对TPC-H基准学习和研究之后，根据官方提供的工具设计与实现了自动化的测试工具，并且通过软件测试，完成了对MySQL的测试。验证了工具的功能性和稳定性，其中整个软件的主要工作和成果总结如下：

（1） 从整个数据库测试的标准开始进行探索，对目前国内外的研究成果进行了初步的了解，对数据库测试的背景有了基本的认识，并着重就TPC-H标准进行了研究，对其规范和模型应用进行深入了解，并且对其商业应用环境、事务模型和测试指标进行了描述，对其提供的DBGEN和QGEN工具进行了编译生成，并弄清楚使用的方法和参数调用方式；

（2） 参考论文，对于测试工具的需求进行分析，并且对于整体设计有所了解，在整个测试系统的框架上进行了初步的研究，并结合论文和自己的理解，对系统模块进行了合理的分类和整体设计，最终完成了稳定性强，测试准确地简单实用软件系统；

（3） 对设计中要实现的功能和模块进行了真实的实现，并且对于存在的困难和疑问进行了逐步解决，并实现了主控与日志、数据库操作、性能测试和ACID测试四大模块，实时监控测试信息，建立了准确地信息反馈渠道，解决了四个难题，其一为主控模块与其他模块沟通问题，其二为测试软件对数据库操作问题，包括建立表结构和加载数据等，其三为性能测试调用官方工具，生成测试语句问题，其四为ACID共10个测试的定义理解和实现步骤解决的问题；

（4） 进行了大规模的软件测试，修复了存在的bug，并且将操作简单化，便于用户对于软件的使用，同时对于测试出的数据进行了详尽的分析，得出准确地数据库测试结果，能够真正的为用户选择提供可靠地信息，同时有利于整个社会资源的合理利用，节约社会资源也是该软件的优势。

虽然测试软件的实现取得了相应的成果，但是由于技术操作和系统测试不完全的原因，系统中还存在着诸多需要修改的地方，有待继续加工与完善。

（1） 数据库测试系统在ACID测试验证上个别时候存在问题，验证不够完全，需要更细致的阅读TPC-H基准，进行更细致的测试以排除潜在的测试问题；

（2） 测试系统在本地测试上完成度较好，通过远程连接会存在一定问题，并且对于长时间测试，系统的崩溃处理和硬件异常恢复没有做相应的处理。

对于不足之处，建议之后的研究人员可以继续遵循TPC-H基准，从ACID验证、数据库连接、功能扩展以及系统修复等多个角度进行软件的完善，进而增加软件的完整性和稳定性。

参 考 文 献

[1] 谭磊.解析大数据基准测试——TPC-H or TPC-DS[EB/OL].[2015,05,14].

http://www.chinacloud.cn/show.aspx?id=20146&cid=17

[2] Lo E, Cheng N, Lin W W K, et al. MyBenchmark: generating databases for query workloads [J]. Vldb Journal, 2014, 23(6):895-913.

[3] 王良，栗跃鹏，杨政. TPC-H 自动测试工具 TPCHDriver 的研究[J].计算机工程与应用,2007,43(3):77-80.

[4] 康强强，江舟，金澈清，周傲英. 一个自动化测试工具的设计与实现[J].计算机研究与发展,2013,50(Suppl.):394-398.

[5] 高宁宁. 面向国产安全数据库的测试体系的构建[D].北京:中国科学院大学.2014.

[6] 曾妍. 基于Eclipse RCP的DB2 for i数据库测试系统的设计与实现[D].长春:吉林大学,2009.

[7] 王天云. 软件开发中的数据库测试技术研究[D].长春:长春工业大学,2013.

[8] 司冠男，许静. 基于TPC-C的XML数据库测试方案[D].天津:南开大学,2010.

[9] 周鹏. 数据库系统性能和可靠性评测技术的研究与设计[D].哈尔滨:哈尔滨工业大学,2012.

[10] 孙辉. MySQL查询优化的研究和改进[D].武汉:华中科技大学,2005.

[11] 张金莲. 数据库决策支持评测系统的扩展和InnoDB存储管理分析[D].长春:吉林大学,2002.

[12] [Diana Marculescu](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Diana%20Marculescu.QT.&newsearch=true), Sebastian Herbert. Analysis of dynamic voltage/frequency scaling in chip-multiprocessors [J].Proceedings of the International Symposium on Low Power Electronics & Design, 2007:38-43.

[13] [Jianyong Zhang](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.Jianyong%20Zhang.QT.&newsearch=true), [A. Sivasubramaniam](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.A.%20Sivasubramaniam.QT.&newsearch=true), [M. Kandemir](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=%22Authors%22:.QT.M.%20Kandemir.QT.&newsearch=true). Interplay of energy and performance for disk arrays running transaction processing workloads [J]. [Performance Analysis of Systems and Software](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8467), 2003:123-132.

[14] 司美琴. [DB2统计信息测试数据自动生成系统的设计与实现](http://epub.cnki.net/kns/detail/detail.aspx?QueryID=3&CurRec=28&recid=&FileName=2007164544.nh&DbName=CMFD2007&DbCode=CMFD&pr=)[D].北京:北京邮电大学,2007.

[15] 薛萍. 多级安全数据库测试技术研究与测试工具设计[D].青岛:中国海洋大学,2011.

# 致 谢

在即将结束本科的学习之际，首先要感谢毕业设计这段时间指导我的老师，真诚感谢老师在这半年中对我的指导，从项目伊始让我参与到整个团队之中，除了专业知识外，还学习到了很多在日常学习中得不到的知识，团队合作、认真细致的工作态度都是我学习到的，老师刻苦、严肃的工作态度和品德是我学习的目标和榜样。除此之外，还要感谢老师在我研究和写作过程中的耐心指导，提出了很多宝贵意见，再论文完成之际，向老师表示深深的感谢！

同时还要感谢项目总负责老师，老师作为整个队伍的负责人，他认真的工作态度和一丝不苟的工作精神深深的打动着我，激励着我，在即将毕业之际，对我们毕业生更加关怀备至，营造出了很好的实验室氛围，让每个人都觉得工作起来很舒服，并且及时的督促我们进行相应的研究工作和论文工作，营造了良好的工作环境和团队氛围，感谢老师一直以来的付出。

还要感谢与我一起学习的沈佳凯、冯玉杰、寇庆祥、赵阳、孙嘉鑫、钱小宇等同学，在整个项目完成的过程中，我们一起工作，一起努力，我得到了他们的支持和帮助，并且共同完成了很多工作，在这段日子里，极大地丰富了我大学最后一年的生活，这段日子将会成为我生命中宝贵的财富，也是最美好的回忆。

还要衷心感谢父母在我大学四年中给予的支持和鼓励，虽然身在异地，但是他们一直以来都关心我，支持我让我感受到家的温暖，愿你们健康平安，开心快乐。

同时感谢陪我度过大学时光的每一个人，因为你们我的大学才会如此绚丽多彩，如此难忘，感谢每一个出现在我生命中的人，谢谢你们。

最后感谢各位评委的细心审阅，感谢他们的宝贵意见。