科研管理信息系统中数据库的设计与实现

摘 要

数据库技术出现于 20 世纪 60 年代末，是计算机科学技术中发展最快的领域之一。数据库系统已在当代社会生活中获得了广泛的应用，渗透到了工农业生产、商业、行政管理、科学研究、教育、工程技术和国防军事等各行各业，而且已围绕数据库技术形成了一个巨大的软件产业，即数据库管理系统和各类工具软件的开发和经营。

北京化工大学科研管理信息系统（以下简称本系统）是为了适应网络环境下的科研管理任务而开发设计的一套软件，它以网络为媒体提供对各种科研信息的处理功能。数据库系统作为本系统的一部分，它起着方便数据管理、为数据的完整性和安全性提供保障以及实现数据的共享等作用，是系统中必不可少的重要部分。本文是针对本系统数据库的设计与实现来进行的。

首先，本文系统讨论了数据库的产生和发展趋势，对管理信息系统作了简单的介绍。在对各种数据库系统的性能作了初步测试的基础上，选定了本系统数据库适合的数据库系统。并且探讨了数据库的开发应用模式。

其次，本文通过对数据库开发过程的讨论，讨论了本系统数据库的设计开发过程。重点讨论了数据库设计实现过程中的如下几个阶段，需求分析、概念结构设计、逻辑结构设计和物理结构设计。

最后，本文对本系统数据库的安全和优化作了讨论。一方面，通过对数据库安全方面的讨论，论述了本系统数据库的安全措施。另一方面，通过对数据库优化的定义和方法论述，讨论了本系统数据库的优化方法。

关键字：数据库技术，管理信息系统，设计开发，数据库安全，数据库优化

THE DESIGN AND IMPLEMENTATION OF SCIENTIFIC RESEARCH MANAGEMENT INFORMATION SYSTEM'S DATABASE

ABSTRACT

Database technology that has been promoted since 1960's is one of the fastest developing domains in computer science technology. Database system has been applied widely in many scopes of current society life such as industry and agriculture production, commerce, administration, science research, education, engineering, national defense and military affairs and so no. Futhermore a large software industry about it has come into being, which is the development and management of database management system and all kinds of tool software.

Scientific research management information system of BUCT(for short: this system) is a software served to materialized the management of scientific research through internet. It provides the processing function of all sorts of scientific research information through the media of internet. Database system is an important part in this system. It can make data-management expediently, can ensure integrality and security the data and can implement data-share and so on. It is an absolutely necessarily part in this system. This paper aims at studying the design and implementation of this system's database.

Firstly, this paper systematically introduces the development of database technologies, fellowed by the brief introduction of management information system. Choose the proper database system for this system based on primary testings of different kinds of database system. In the first

the paper also discusses the empolder mode of database system.

Secondly, this paper discusses the empolder process of this system through discusses the empolder process of database system. The paper focuses on several phases of the empolder process, such as requirement-analysis, concept-structure-design, logic-structure-design and physics-structure-design.

Thirdly, this paper discusses the security and optimization of database system. On one hand, the paper discusses the security-measure of this system through discusses database- security. On the other hand, the paper discusses the optimization-method of this system through discusses the definition and method of database-optimization.

KEY WORDS: database technologies, management information system, design and exploitation, database security, database optimization

北京化工大学位论文原创性声明

本人郑重声明： 所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：林东杰 日期：06.5.16

关于论文使用授权的说明

学位论文作者完全了解北京化工大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京化工大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。

保密论文注释：本学位论文属于保密范围，在 2 年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

作者签名：林东杰 日期：06.5.16

导师签名：[illegible] 日期：06.5.16

第一章 绪论

1.1 课题来源与背景

本课题来源于实际项目——北京化工大学科研管理信息系统（以下简称北化科研系统）数据库的设计与实现。北化科研系统是在原有的多个 C/S 管理系统的基础上开发的以 Oracle 数据库为中心，以 J2EE 技术为平台，B/S 结构的科研管理信息系统[1]，为我校的科研管理工作的自动化提供了强有力的支持，目前，本项目已基本开发成功。数据库作为科研管理信息系统的重要组成部分，起着实现数据信息的存贮管理、完成数据集成和共享的作用。我参与了该项目的开发与实现，本论文主要研究内容是以该系统数据库的设计与实现、安全和优化为背景来论述。

1.2 数据库系统概述

本论文研究的主要内容是数据库的设计与开发，下面从数据库的基本概念及其产生与发展等方面来对数据库系统进行简单论述。

1.2.1 数据库的常用术语和基本概念

1、数据[2]（Data）

数据是数据库中存储的基本对象。数据在大多数人头脑中的第一个反应就是数字。其中数字只是最简单的一种数据，是数据的一种传统和狭义的理解。广义的理解，数据的种类很多，文字、图形、图像、声音、学生的档案记录等，这些都是数据。

数据可以定义如下：描述事物的符号记录称为数据。描述事物的符号可以是数字，也可以是文字、图形、图像、声音、语言等，数据有多种表现形式，它们都可以经过数字化后存入计算机。

2、数据库[2]（DataBase，简称 DB）

数据库，顾名思义，是存放数据的仓库。只不过这个仓库是在计算机存储设备上，而且数据是按一定格式存放的。

所谓数据库是长期储存在计算机内的、有组织的、可共享的数据集合，数据

所谓数据库是长期储存在计算机内的、有组织的、可共享的数据集合，数据

库中的数据按一定的数据模型组织、描述和储存，具有较小的冗余度、较高的数据独立性和易扩展性，并可为各种用户共享。

3、数据库管理系统（DataBase Management System，简称 DBMS）

数据库管理系统是位于用户与操作系统之间的一层数据管理软件，是数据库系统的重要组成部分。它的主要功能包括以下几个方面：

（1）数据定义功能，DBMS 提供数据定义语言（Data Definition Language，简称 DDL），用户通过它可以方便地对数据库中的数据对象进行定义。

（2）数据操纵功能，DBMS 还提供数据操纵语言（Data Manipulation Language，简称 DML），用户可以使用 DML 操作数据实现对数据库的基本操作，如查询、插入、删除和修改等。

（3）数据库的运行管理，数据库的建立、运用和维护是由数据库管理系统统一管理、统一控制，以保证数据的安全性、完整性、多用户对数据的并发使用及发生故障后的系统恢复。

（4）数据库的建立和维护功能，它包括数据库初始数据的输入、转换功能，数据库的转储、恢复功能，数据库的重组织功能和性能监视、分析功能等。这些功能通常是由一些实用程序完成的。

4、数据库系统（DataBase System，简称 DBS）

数据库系统是指在计算机系统中引入数据库后的系统，一般由数据库、数据库管理系统（及其开发工具）、应用系统、数据库管理员和用户构成。应当指出的是，数据库的建立、使用和维护等工作只靠一个 DBMS 远远不够，还要有专门的人员来完成。这些人被称为数据库管理员（DataBase Administrator，简称 DBA）。

数据库系统可以用图 1-1 表示。

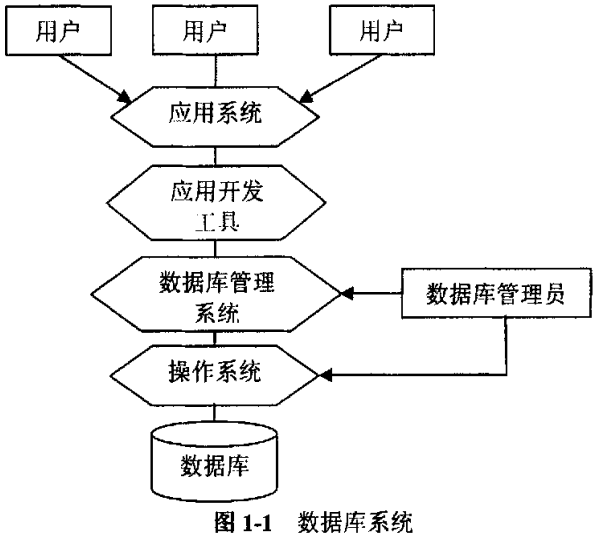


图 1-1 数据库系统

Fig.1-1 the database system

数据库系统在整个计算机系统中的地位如图 1-2 所示。

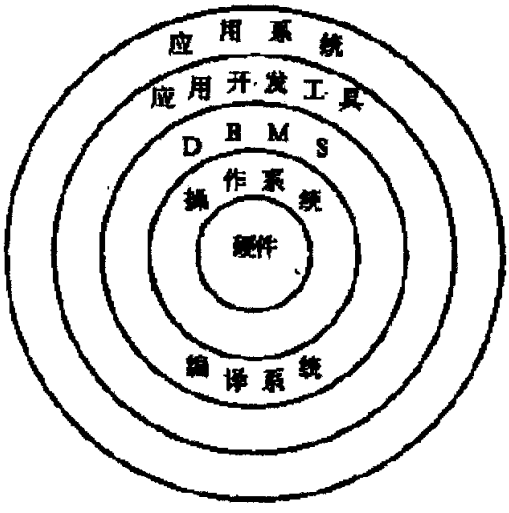


图 1-2 数据库在计算机系统中的地位
Fig.1-2 the status of database in computer sytem

1.2.2 数据库技术的产生和发展

数据库技术是应数据管理任务的需要而产生的。随着科学技术的发展，人们掌握和处理的信息越来越多，要想充分的开发与利用这些信息资源，就必须对大量的信息进行识别、存储、处理与传递[3]。以计算机为基础的数据库技术，由于具有信息存储量大、处理和传输速度快、逻辑推理严密、重复性高、能有效合理地存储各种信息、能为有关应用准确快速地提供有用信息等特点，很快成为了信息处理的强有力的工具。

人们借助计算机进行数据处理是近三十年的事情。研制计算机的初衷是利用它进行复杂的科学计算[4]。随着计算机技术的发展，其应用远远地超出了这个范围。在应用需求的推动下，在计算机硬件、软件发展的基础上，数据库管理技术经历了人工管理、文件系统、数据库系统三个阶段。这三个阶段的特点及其比较如表 1-1 所示。

表 1-1 数据管理三个阶段的比较
Table 1-1 the comparison of the data management's three phases

比较内容		人工管理阶段	文件系统阶段	数据库系统阶段
背景	应用背景	科学计算	科学计算、管理	大规模管理
	硬件背景	无直接存取存储设备	磁盘、磁鼓	大容量磁盘
	软件背景	没有操纵系统	有文件系统	有数据库管理系统
	处理方式	批处理	联机实时处理、批处理	联机实时处理、分布处理、批处理
特点	数据的管理者	用户（程序员）	文件系统	数据库管理系统
	数据面向的对象	某一应用程序	某一应用	现实世界
	数据的共享程度	无共享，冗余度极大	共享性差，冗余度大	共享性高，冗余度小
	数据的独立性	不独立，完全依赖于程序	独立性差	高度的物理独立性，一定的逻辑独立性
	数据的结构化	无结构	记录内有结构，整体无结构	整体结构化，用数据模型描述
	数据控制能力	应用程序自己控制	应用程序自己控制	由数据库管理系统提供数据的安全性、完整性、并发控制和恢复能力

1、人工管理阶段

20 世纪 50 年代中期以前，计算机主要用于科学计算。当时的硬件状况是，外存只有纸带、卡片、磁带，没有磁盘等直接存取的存储设备；软件状况是，没有操作系统，没有管理数据的软件；数据处理方式是批处理。人工管理数据具有如下特点：

（1）数据不保存

由于当时计算机主要用于科学计算，一般不需要将数据长期保存，只是在计算某一课题时将数据输入，用完就撤走。不仅对用户数据如此处置，对系统软件有时也是这样。

（2）应用程序管理数据

数据需要由应用程序自己管理，没有相应的软件系统负责数据的管理工作。应用程序中不仅要规定数据的逻辑结构，而且要设计物理结构，包括存储结构、存

取方法、输入方式等。因此程序员负担很重。

（3）数据不共享

数据是面向应用的，一组数据只能对应一个程序。当多个应用程序涉及某些相同的数据时，由于必须各自定义，无法互相利用、互相参照，因此程序与程序之间有大量的冗余数据。

（4）数据不具有独立性

数据的逻辑结构或物理结构发生变化后，必须对应用程序做相应的修改，这进一步加重了程序员的负担。

2、文件系统阶段

20 世纪 50 年代后期到 60 年代中期，这时硬件方面已有了磁盘、磁鼓等直接存取存储设备；软件方面，操作系统中已经有了专门的数据管理软件，一般称为文件系统；处理方式不仅有了批处理，而且能够联机实时处理。用文件系统管理数据具有如下特点：

（1）数据可以长期保存

由于计算机大量用于数据处理，数据需要长期保留在外存上反复进行查询、修改、插入和删除等操作。

（2）由文件系统管理数据

由专门的软件即文件系统进行数据管理，文件系统把数据组织成相互独立的数据文件，利用“按文件名访问，按记录进行存取”的管理技术，可以对文件进行修改、插入和删除的操作。文件系统实现了记录内的结构性，但整体无结构。程序和数据之间由文件系统提供存取方法进行转换，使应用程序与数据之间有了一定的独立性，程序员可以不必过多地考虑物理细节，将精力集中于算法。而且数据在存储上的改变不一定反映在程序上，大大节省了维护程序的工作量。但是，文件系统仍存在以下缺点。

（3）数据共享性差，冗余度大

在文件系统中，一个文件基本上对应于一个应用程序，即文件仍然是面向应用的。当不同的应用程序具有部分相同的数据时，也必须建立各自的文件，而不能共享相同的数据，因此数据的冗余度大，浪费存储空间。同时由于相同数据的重复存储、各自管理，容易造成数据的不一致性，给数据的修改和维护带来了困难。

（4）数据独立性差

文件系统中的文件是为某一特定应用服务的，文件的逻辑结构对应用程序来说是优化的，因此要想对现有的数据在增加新的应用会很困难，系统不容易扩充。一旦数据的逻辑结构改变，必须修改应用程序，修改文件结构的定义。应用程序的改变，例如应用程序改用不同的高级语言等，也将引起文件的数据结构的改变。因此数据与程序之间仍缺乏独立性。可见，文件系统仍然是一个不具有弹性的无

结构的数据集合，即文件之间是孤立的，不能反映现实世界事物之间的联系。

3、数据库系统阶段

20 世纪 60 年代后期以来，计算机用于管理的规模越来越大，应用越来越广泛，数据量急剧增长，同时多种应用、多种语言互相覆盖地共享数据集合的要求越来越强烈。

这时硬件已有大容量磁盘，硬件价格下降；软件则价格上升，为编制和维护系统软件及应用程序所需的成本相对增加；在处理方式上，联机实时处理要求更多，并开始提出和考虑分布处理。在这种背景下，以文件系统作为数据管理手段已经不能满足应用的需求，于是为解决多用户、多应用共享数据的要求，使数据为尽可能多的应用服务，数据库技术便应用而生，出现了统一管理数据的专门软件系统——数据库管理系统。

用数据库系统来管理数据比文件系统具有明显的优点，从文件系统到数据库系统，标志着数据管理技术的飞跃。

1.2.3 数据库系统的特点

与人工管理和文件系统相比，数据库系统的特点主要有以下几个方面：

1、数据结构化

数据库系统实现整体数据的结构化，是数据库的主要特征之一，也是数据库系统与文件系统的本质区别。

在数据库系统中，数据不再针对某一应用，而是面向全组织，具有整体的结构化。不仅数据是结构化的，而且存取数据的方式也很灵活，可以存取数据库中的某一个数据项、一组数据项、一个记录或一组记录。而在文件系统中，数据的最小存取单位是记录，粒度不能细到数据项。

2、数据的共享性高，冗余度低，易扩充

数据库系统从整体角度看待和描述数据，不再面向整个应用而是面向整个系统，因此数据可以被多个用户、多个应用共享使用。数据共享可以大大减少数据冗余，节约存储空间。数据共享还能够避免数据之间的不相容性与不一致性。

所谓数据的不一致性是指同一数据不同拷贝的值不一样。采用人工管理或文件管理时，由于数据被重复存储，当不同的应用使用和修改不同的拷贝时就很容易造成数据的不一致。在数据库中数据共享，减少了由于数据冗余造成的不一致现象。

由于数据面向整个系统，是有结构的数据，不仅可以被多个应用共享使用，而且容易增加新的应用，这就使得数据库系统弹性大，易于扩充，可以适应各种

用户的要求。可以取整体数据的各种子集用于不同的应用系统，当应用需求改变或增加时，只要重新选取不同的子集或加上一部分数据便可以满足新的需求。

3、数据独立性高

数据独立性是数据库领域中一个常用术语，包括数据的物理独立性和数据的逻辑独立性。

物理独立性是指用户的应用程序与存储在磁盘上的数据库中的数据是相互独立的。也就是说，数据在磁盘上的数据库中怎样存储是由 DBMS 管理的，用户程序不需要了解，应用程序要处理的只是数据的逻辑结构，这样当数据的物理存储改变了，应用程序不用改变。

逻辑独立性是指用户的应用程序与数据库的逻辑结构相互独立的，也就是说，数据的逻辑结构改变了，用户程序也可以不变。

数据与程序的独立，把数据的定义从程序中分离出去，加上数据的存取又有 DBMS 负责，从而简化了应用程序的编制，大大减少了应用程序的维护和修改。

4、数据由 DBMS 统一管理和控制

数据库的共享是并发的（Concurrency）共享，即多个用户可以同时存取数据库中的数据甚至可以同时存取数据库中的同一个数据。为此，DBMS 还必须提供以下方面的数据控制功能：

（1）数据的安全性（Security）保护

数据的安全性是指保护数据以防止不合法的使用造成的数据的泄密和破坏。使每个用户只能按规定，对某些数据以某些方式进行使用和处理。

（2）数据的完整性（Integrity）检查

数据的完整性指数据的正确性、有效性和相容性。完整性检查将数据控制在有效的范围内，或保证数据之间满足一定的关系。

（3）并发（Concurrency）控制

当多个用户的并发进程同时存取、修改数据库时，可能会发生相互干扰而得到错误的结果或使得数据块的完整性遭到破坏，因此必须对多用户的并发操作加以控制和协调。

（4）数据库恢复（Recovery）

计算机系统的硬件故障、软件故障、操作员的失误以及故意的破坏也会影响数据库中数据的正确性，甚至造成数据库部分或全部数据的丢失。DBMS 必须具有将数据库从错误状态恢复到某一已知的正确状态（已成为完整状态或一致状态）的功能，这就是数据库的恢复功能。

数据库管理阶段应用程序与数据之间的对应关系如图 1-3 所示。

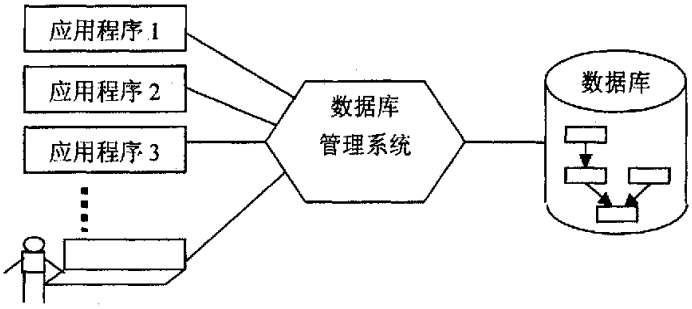


图 1-3 数据库管理阶段应用程序与数据之间的对应关系

Fig.1-3 the relation of application and data in database management phase

综上所述，数据库是长期存储在计算机内有组织的大量的共享的数据集合。它可以供各种用户共享，具有最小冗余度和较高的数据独立性。DBMS 在数据库建立、运用和维护时对数据库进行统一控制，以保证数据的完整性、安全性，并在多用户同时使用数据库时进行并发控制，在发生故障后对系统进行恢复。

数据库系统的出现使信息系统从以加工数据的程序为中心转向围绕共享的数据库为中心的新阶段。这样既便于数据的集中管理，又有利于应用程序的研制和维护，提高了数据的利用率和相容性，提高了决策的可靠性。

1.3 数据库技术领域的发展趋势

1.3.1 国际数据库研究界动态

数据库技术从诞生到现在，在不到半个世纪的时间里，形成了坚实的理论基础、成熟的商业产品和广泛的应用领域，吸引了越来越多的研究者加入，使得数据库成为一个研究者众多且被广泛关注的研究领域，随着信息管理内容的不断扩展和新技术的层出不穷，数据库技术面临着前所未有的挑战。面对新的数据形式，人们提出了丰富多样的数据模型（层次模型、网状模型、关系模型、面向对象模型、半结构化模型等），同时也提出了众多新的数据库技术（XML 数据管理、数据流管理、Web 数据集成、数据挖掘等）。

每隔几年，国际上一些资深的数据库专家就会聚集一堂，探讨数据库的研究现状、存在的问题和未来需要关注的新的技术焦点，其中包括：1989 年在 Laguna Beach，Calif.[5]，1990 年和 1995 年在 Palo Alto，Calif.[6-7]，“Lagunita”，1996 年在 Cambridge，Mass.[8]，1998 年在 Asilomar，Calif.[9]和 2003 年在 Lowell，Mass.[10]举行的研讨会。与会的学者集中讨论了信息的存储、组织、管理和访问等问题。这些问题受新型应用、技术趋势、相关领域的协同工作和领域本身的技术变革所

驱动，信息的本质和来源在不断变化，每个人都意识到 Internet、Web、自然科学和电子商务是信息和信息处理的巨大源泉。同时，另一个巨大的信息源即将到来，即廉价的微型传感器技术使得大部分的物体可以实时上报它们的位置和状态。这类信息能支持对移动对象的状态和位置的监视等应用。

伴随新的制约与机会，传感信息的处理将会引发许多新环境下的极有趣味的数据库问题。在应用领域，Internet 是目前主要的驱动力，特别是在支持“跨企业”的应用上。在历史上，应用都是企业内部的，可以在一个行政领域内进行完善的制定和优化。但是现在，大部分企业感兴趣的是如何与供应商和客户进行更密切的交流，以便提供更好的客户支持。这类应用从根本上说是跨企业的，需要安全和信息集成的有力工具。由此产生的新问题需要数据库研究人员去解决。

越来越重要的另一个应用领域是自然科学，特别是物理科学、生物科学、保健科学和工程领域，这些领域产生了大量复杂的数据集，需要比现在的数据库产品更高级的数据库的支持。这些领域同样也需要信息集成机制的支持。除此之外，它们也需要对数据分析器产生的数据管道进行管理，需要对有序数据进行存储和查询（如时间序列、图像分析、网络计算和地理信息），需要世界范围内数据网格的集成。

除了在信息管理领域遇到的这些挑战之外，在传统的 DBMS 相关的问题上，诸如数据模型、访问方法、查询处理代数、并发控制、恢复、查询语言和 DBMS 的用户界面等主题也面临着巨大的变化。

另一个推动数据库研究发展的动力是相关技术的成熟。比如，在过去的几十年里，数据挖掘技术已经成为数据库系统重要的一个组成部分。Web 搜索引擎导致了信息检索的商品化，并需要和传统的数据库查询技术集成。

许多人工智能领域的研究成果也和数据库技术融合起来，这些新的技术使得我们可以处理语言、自然语言，进行不确定性推理和机器学习等。

1.3.2 数据库主流技术发展趋势

目前，数据库的主流技术包括以下几个方面：

1、信息集成

信息系统集成技术已经历了 20 多年的发展，已经提出了很多信息集成的体系结构和实现方案，然而这些方法所研究的主要集成对象是传统的异构数据库系统。随着 Internet 的飞速发展，网络迅速成为一种重要的信息传播和交换的手段，尤其是在 Web 上，有着极其丰富的数据来源。如何获取 Web 上的有用数据并加以综合利用，即构建 Web 信息集成系统，成为一个引起广泛的研究领域。信息集成系统

的方法可以分为[11]：数据仓库方法和 Wrapper/Mediator 方法。

2、数据流管理

测量和监控复杂的动态的现象，如远程通信、Web 应用、金融事务、大气情况等，产生了大量、不间断的数据流。数据流处理对数据库、系统、算法、网络和其他计算机科学领域的技术挑战已经开始显露，这是数据库界一个活跃的研究领域，包括新的流操作、SQL 扩展、查询优化方法、操作调度（operator scheduling）技术等[8]。

3、传感器数据库技术

新的传感器数据库系统需要考虑大量的传感器设备的存在，以及它们的移动和分散性。因此，新的传感器数据库系统需要解决一些新的问题。主要包括：

（1） 传感器数据的表示和传感器查询的表示；

（2） 在传感器节点上处理查询分片：传感器资源的有限性，要求必须有效地处理各个节点上的查询；

（3） 分布查询分片：产生和传输传感器数据都需要花费代价，必须考虑单个节点的查询效率和网络传输代价的平衡；

（4） 适应网络调节的改变；

（5） 处理站点失败和传输失败的情况；

（6） 传感器数据库系统：必须利用系统中的所有传感器，而且可以像传统数据库那样方便、简洁地管理传感器数据库中的数据等。

4、XML 数据管理

目前大量的 XML 数据以文本文档的方式存储，难以支持复杂高效的查询。用传统数据库存储 XML 数据的问题在于模式映射带来的效率下降和语义丢失。XML 数据是半结构化的，不像关系数据库那样是严格的结构化数据，这样就给 XML 数据库中的存储系统带来更大的灵活性，同时，也带来了更大的挑战。恰当的记录划分和簇聚，能够减少 I/O 次数，提高查询效率；反之，不恰当的划分和簇聚，则会降低查询效率。研究不同存储粒度对查询的支持也是 XML 存储面临的一个关键性问题[12]。

5、网格数据管理

简单地讲，网格是把整个网络整合成一个虚拟的巨大的超级计算环境，实现计算资源、存储资源、数据资源、信息资源和专家资源的全面共享。目的是解决多机构虚拟组织中的资源共享和协同工作问题。按照应用层次的不同可以把网格分为三种：计算网格，提供高性能计算机系统的共享存取；数据网格，提供数据库和文件系统的共享存取；信息服务网格则支持应用软件和信息资源的共享存取。

由于得到政府和企业的大力支持，近年来，网格技术在国内外都得到飞速发展。网格研究呈现了实验室研究和实际应用紧密结合的局面，网格技术的应用从

单纯的科学计算领域扩展到企业计算领域，并推动了相关的产业化进程。

6、DBMS 的自适应管理

随着 RDBMS 复杂性增强以及新功能的增加，使对数据库管理人员的技术需求和熟练数据库管理人员的薪水支付都在大幅度增长，导致企业人力成本支出也在迅速增加。随着关系数据库规模和复杂性的增加，系统调整和管理的复杂性相应增加，现在，一个 DBA 必须了解磁盘分区，并行查询执行，线程池和用户定义的数据类型。基于上述原因，数据库系统自调优和自管理工具的需求增加，对数据库自调优和自管理的研究也逐渐成为热点。

7、移动数据管理

目前，蜂窝通信、无线局域网以及卫星数据服务等技术的迅速发展，使得人们可以随时随地访问信息的愿望成为可能。在不久的将来，越来越多的人将会拥有一台掌上型或笔记本电脑，或者个人数字助理（PDA）甚至智能手机，这些移动计算机都将装配无线联网设备，从而能够与固定网络甚至其他的移动计算机相联。用户不再需要固定地联接在某一个网络中不变，而是可以携带移动计算机自由地移动，这样的计算环境，称之为移动计算（mobile computing）。

研究移动计算环境中的数据管理技术，已成为目前分布式数据库研究的一个新的方向，即移动数据库技术。

8、微小型数据库技术

数据库技术一直随着计算的发展而不断进步，随着移动计算时代的到来，嵌入式操作系统对微小型数据库系统的需求为数据库技术开辟了新的发展空间。微小型数据库系统是一个只需很小的内存来支持的数据库系统内核。内存限制是决定微小型数据库系统特征的重要因素。根据占用内存的大小可以分为：超微 DBMS（pico-DBMS）、微小 DBMS（micro-DBMS）和嵌入式 DBMS 三种。

微小型数据库技术[13-15]目前已经从研究领域逐步走向应用领域，各种微小型数据库产品纷纷涌现，尤其是对移动数据处理和管理需求的不断提高，紧密结合各种智能设备的嵌入式移动数据库技术得到了学术界、工业界、军事领域和民用部门等各方面的重视部不断实用化。

9、数据库用户界面

一直以来，一个普遍的悲哀是数据库学术界在用户界面方面做得工作太少了。目前，计算机已经有足够的能力在桌面上运行很复杂的可视化系统。然而，对于一个 DBMS 给定的信息类型，如何使它在可视化上达到最优还不清楚。20 世纪 80 年代是，提出了少数优秀的可视化系统，尤其是 QBE 和 Visi Cal c。但至今仍没有更优秀的系统出现，因此迫切需要在这方面有所创新[10]。

数据库界面的研究在我国一直未引起足够的重视，因此缺乏适合我国用户的数据库界面。开展数据库中文自然语言界面的研究十分有意义。中文自然语言查

询系统 NChiql[16-17]在这方面做了有益的尝试。特别在今天，计算机的汉语语音识别已初步达到实用的阶段，中文语言查询界面若与语音识别配套，前景十分诱人。

1.4 北化科研系统的数据库系统

本论文主要是以北化科研系统的数据库为背景来进行数据库设计和实现的研究，下面首先简单介绍管理信息系统，然后论述了北化科研系统产生的背景，最后对北化科研系统的数据库系统的重要性进行了论述。

1.4.1 管理信息系统简介

1、管理信息系统的定义

管理信息系统 MIS（Management Information System）是从 20 世纪 60 年代发展起来的。管理信息系统[18]是一个以人为主导、利用计算机硬件、软件、网络通信设备以及其他办公设备，进行信息的收集、传输、加工、储存、更新和维护，以企业战略竞优、提高效益和效率为目的，支持企业高层决策、中层控制、基层运作的集成化人机系统。通过管理信息系统可以实测企业生产经营活动过程中的实际运行情况，并能利用历史数据对未来进行预测，从全局出发辅助管理人员做出科学决策。

管理信息系统是随着现代科学技术的发展和现代化管理的客观需要而形成的。管理科学、系统科学、信息科学、计算机科学和现代通信技术都对管理系统的形成和发展起到了巨大的促进和推动作用[19]。管理信息系统作为现代化管理的重要手段和标志，已经成为管理活动中必不可少的一个组成部分。

2、管理信息系统的功能和特点

根据管理信息系统的定义，可以看出管理信息系统具有以下的基本功能：

（1） 数据处理功能：管理信息系统能对各种形式的原始数据进行收集、整理及保存，以便向管理者及时、全面、准确地提供所需要的各类信息；

（2） 预测功能：根据一定的数学方法和预测模型，可以利用历史的数据对未来进行预测；

（3） 计划功能：对各种具体工作能合理地计划和安排，对不同的管理层次提出不同的要求，提供不同的信息，以提高管理工作效率；

（4） 控制功能：对整个系统的各个部门及各个环节的运行情况进行监测，可以及时发现问题进行纠正；

（5） 决策功能：在系统中利用运筹学的方法和技术，可以为最佳决策提供科学依据，以便合理地利用各种资源，提高效益。

管理信息系统的基本特点，概括起来可以有以下几个方面：

（1） 管理信息系统是一个人机系统：利用计算机强大的处理和存储能力，这既是管理现代化的客观要求，也是管理信息系统的基本特点。但是，人在利用计算机的同时，必须考虑到各自的特长，使管理信息系统成为一个人机协调的系统；

（2） 管理信息系统是一个人机一体化的集成系统：管理信息系统是以系统思想为指导进行设计和建立的。因此保证了整个系统的统一和协调，使得系统中的数据具有一致性和共享性。现代的网络技术和数据库技术是实现管理信息系统一体化的重要技术基础；

（3） 数据库的应用：具有集中统一规划的数据库是现代管理信息系统的重要特点；它标志着管理信息系统真正实现了数据的集中统一，使数据成为各种用户共享的资源；

（4） 数据模型的应用：在管理信息系统中利用计算机的计算功能和判断能力来分析数据，进行预测和辅助决策，是管理信息系统的又一显著特点，通过应用数学模型可以为管理人员进行最佳决策提供必要的信息。

3、管理信息系统的应用现状

由于管理信息系统的实用性和它给社会带来的巨大经济效益，所以从它问世以来，一直受到各国的普遍重视。随着科学技术特别是计算机科学和现代通信技术的迅速发展，管理信息系统得到了极为快速的发展。

在理论上，管理信息系统的知识体系已经形成，并且有了一套完整的解决问题的方法和程序。目前，在一些高等院校已建立管理信息系统专业，也有一些院校将管理信息系统课程作为管理类、财经类和计算机专业的课程。从这一点上，也可以看出它在理论上已经走上成熟。

在实际应用中，由于新技术革命的兴起和微型计算机的迅速普及，以计算机为主要信息处理手段的管理信息系统在各行各业中得到了广泛的应用。例如金融、通信等行业，都越来越多的借助管理信息系统，使其管理迈上了一个新台阶。

4、管理信息系统的发展方向

随着科学技术的发展和管理水平的提高，无论在深度上还是在广度上，管理信息系统都有着广阔的发展前景。

在理论上，管理信息系统正朝着以下几个方向发展：

（1） 管理信息系统研究方法的进一步系统化的研究；

（2） 管理信息系统研制工具的进一步改善和完善的研究；

（3） 管理信息系统模型的进一步研究；

（4） 管理信息系统对人、组织和社会影响的进一步研究。

在实际应用上，概括起来是向着高层次和普及性两个方向发展。从普及性方面来说，随着微型计算机功能的增强和广泛应用，以微型计算机为主要信息处理工具的、功能完善的管理信息系统将越来越多地受到各级管理者的欢迎，并且要逐渐地向着用于支持个人独立思考和决策活动的方向发展。

从管理信息系统应用的高层次上来讲，以大型计算机为中央处理机的分布式管理信息系统也将是一个重要的发展方向，利用微型计算机和大型主机联网，可以为管理者提供更加广泛的信息和决策支持。另外，各种类型的专家系统、人工智能系统、决策支持系统也将得到进一步的研究和发展，并逐步走进应用领域。

从管理信息系统的体系结构看，传统的管理信息系统一般采用两层 C/S 结构，这种结构集中了大中型系统及文件服务器的优点，并有良好的系统开放性和可扩展性，它一般应用于局域网。但是，随着信息的全球化，区域的界限已经被打破，人们已经越来越不满足于只在一个小的区域内共享信息，尤其是近年来的电子商务在 Internet 的兴起，已成为一种强大的驱动力，迫使 C/S 模式从局域网（LAN）向广域网（WAN）延伸。如今，Internet 已经成为全球最大的网络互连环境，在 Internet 的环境下实现数据的 B/S 计算模式正是目前的流行趋势。

1.4.2 北化科研系统产生的背景

科研工作的好坏体现了一所高校的科技创新能力和学术研究水平。因此，对学校科研工作的管理是整个高校管理的核心和基础。随着科学研究的不断发展和科研水平的不断提高，高校科研管理部门的责任日趋加重，科研课题的申报和立项逐年增加，科研项目、经费和成果的管理及大量科研信息的搜集、储存与分析统计等工作任务越来越重。为使科研管理工作更加规范化、科学化、现代化，使科研管理人员从繁杂的事务性、重复性的手工劳动中解放出来，提高办事效率，使科研管理水平上层次，建立一个高效的、实用的科研信息管理系统势在必行。

目前国内高校开发的科研信息系统管理软件种类较多，但都是自成体系、通用性较差。因此，根据北京化工大学这一具有典型的以工为主兼经管文法综合型大学的科研管理模式，自主设计开发满足新形势需求的科研信息管理系统对于提高本校科研管理水平，加强本校科研管理工作具有十分重要的现实意义。

北京化工大学科技处负责管理着北京化工大学整个学校科研工作，包括两个方面，一个是高校的日常的科研管理工作，如项目合同的入库维护管理，经费的到款、入账、转账、结算和退费管理（包括各个阶段的经费报表单打印），论文和成果的入库和维护管理以及科研信息的发布。另一方面是日常的查询工作及阶段

性的统计工作。对于系统中的数据，要求可以在平常通过各个角度进行多方位的查询，然后列表显示。对于经费中比较重要的数据，要求能够做到每个月统计，然后导出 excel 或者打印。每年特定的时间，也要整理出数据，然后作为科研业绩衡量的标准。例如：在每年 9-11 月，要求项目负责人对结算的经费进行分配，然后在该年年末，对全校的科研人员的所有科研信息，包括项目合同，经费，论文和成果，作一个全面的、准确地、详细的统计，并且根据一定的规则，兑算出一个分值，而这个值是衡量老师这一年来科研业绩的最重要的依据。这样的业绩考核，为我校的教职工的职称评定、岗位考核及科学决策提供重要的依据。同时，在每年的特定时间，可以根据做好的报表模板，生成教育部要求的年统计报表。如果以上的工作全部用手工完成，那么至少要好几个人录入和维护统计这些数据，而且数据的准确性和一致性上会出现问题。

北京化工大学原有的科研管理信息系统是传统的计算机管理模式，这种系统在过去几年里为完善科研管理工作、提高工作效率和方便用户做出了一定的贡献。然而，随着信息网络技术的进步，人们信息意识的增强以及信息操作技能的普及和提高，传统的科研管理系统的不足显得越来越突出，主要表现在以下几点：

（1） 用户无法通过网络进行科研信息查询，只能到科研管理部门查询，同时科研管理部门的信息也无法及时地送到用户手中；

（2） 随着科研工作的发展和管理工作的完善，传统系统中诸如项目和经费管理的模块已远不能满足当前工作的要求；

（3） 信息流向单一，不能同时对数据进行综合性处理；

（4） 传统的管理系统可维护性和可扩充性较差。

针对原有系统的不足，在原有的单机版信息管理系统的基础上，设计并开发了一套基于 J2EE 技术的网络版科研管理信息系统，从而克服了原有系统运行环境、数据库等的异构性，提供了一个全面支持网络应用、数据集中统一管理和易扩展的集中统一的平台。该系统能够实现高校的科研机构、科研人员、科研项目、科研经费、科研著作论文、成果奖励专利等提供实用、先进化的网络信息管理，为高校的职称评定、岗位聘任、评先选优及科学决策提供重要的依据，并支持统计查询以及教育部的年度报表功能，为我校的科研管理提供了极大的方便。新的北化科研系统实现了以下几点基本目标：

（1） 信息标准化：代码信息的录入遵循国家标准代码、教育部标准等规范；

（2） 业务管理规范化：将科研相关业务进行分类整理，提供规范的科研管理业务流程；

（3） 用户接口人性化：从用户角度出发，提供给用户方便、易于使用的界面接口；

（4） 资源共享最大化：通过网络实现资源利用和共享的最大化。

新系统的特点和优势体现在以下几个方面：

（1） 标准化和规范性：系统信息代码采用国家教育部最新制定的相关标准；

（2） 全面支持网络应用：网络应用的优势在于可以设置多种角色，不同角色具有不同程度的使用权限；

（3） 数据集中管理：系统可以对数据进行集中管理，有效保证数据的一致性和易维护性，而且数据集中的另一个优势就是方便数据挖掘与资源共享；

（4） 易扩展性：系统在规划设计时，充分考虑到以后学校科研管理工作中可能存在的具体情况，系统采用模块化和组建化的设计模式，可以方便地对系统进行升级扩展；

（5） 易使用性：采用 B/S 多层结构，通过 WWW 浏览器就能完成系统提供的所有业务操作，实现了 anywhere， anytime 的使用方式，提高了系统的实用性能。

（6） 系统安全性：通过 PKI 等技术来保证数据的认证性、机密性、完整性和不可否认性；采用灵活的基于角色的访问控制模型，实现灵活的访问控制体系；强大的日志和审计功能，便于日后核查，及早发现安全隐患。

1.4.3 北化科研系统的数据库系统

目前，数据库已经成为现代信息系统的不可分离的重要组成部分。具有数百万甚至数十亿字节信息的数据库已经普遍存在于科学技术、工业、农业、商业、服务业和政府部门的信息系统。20 世纪 80 年代后不仅在大型机上，在多数微机上也配置了 DBMS，数据库技术倒到广泛的应用和普及。

同样，数据库也是北化科研系统必不可少的重要部分。在北化科研系统中，数据库系统和一部分的 JavaBean 构成了 MVC（Model-View-Controller）架构中的 M 部分，在科研管理信息系统中起着至关重要的作用。

数据库系统对科研管理信息系统的数据进行集中管理，有效保证数据的一致性和易维护性，数据集中的另一个优势就是方便数据挖掘与资源共享。数据挖掘即管理人员可对数据库中的信息进行综合性的处理，以实现科技统计工作和为科学决策提供依据；资源共享是使科研管理信息系统的用户之间可以共享系统的信息资源。通过组织良好的数据库系统，使得系统可以方便、迅速、准确的为用户提供服务。

1.5 本论文完成的主要工作

本论文主要完成以下工作：

第二章主要是对北化科研系统所涉及到数据库的相关内容做出比较和总结。通过对各种数据库系统介绍、数据库性能对比决定适合于本系统的数据库；介绍了现在流行的数据库应用开发模式。在此基础上对北化科研系统数据库所涉及到的相关内容进行了总结。

第三章主要是针对北化科研的数据库的设计。首先论述了数据库设计的重要性，然后从数据库设计开发过程的几个方面来介绍科研管理信息系统的数据库设计。着重从需求分析、数据库概念结构设计、数据库逻辑设计和数据库物理设计等方面对北化科研系统的设计进行了论述。

第四章主要是针对北化科研系统的数据库的安全。由信息安全的现状到数据库安全的现状，分析了数据库安全受到的威胁，以及数据库安全的特点，从 Oracle 数据库系统的自身安全入手，论述了北化科研系统的数据库的安全机制。

第五章主要是针对数据库的优化。由数据库优化的概念入手，分析了数据库优化的必要性；从数据库优化的方法出发，论述了北化科研系统的数据库的优化。

第二章 科研管理信息系统中涉及到数据库的相关内容

选择不同的数据库系统对于系统运行的效率和稳定性有很多的影响。下面先简单介绍各种数据库并通过测试对比各种数据库的性能，然后再对数据库的应用开发模式进行简单介绍。

2.1 各种数据库系统简介

2.1.1 SqlServer2000 数据库系统

Sqlserver2000[20]是微软提供的一种涵盖个人用户，工作组用户，企业级用户的全线全功能数据库产品，该数据库产品继承了微软产品的一贯特点，比如使用、安装简单；较少高级设置；功能强大、工具丰富；高端使用较少，中低端使用丰富；极度依赖于微软平台等。

2.1.2 Oracle 数据库系统

Oracle 数据库[21]是以高级结构化查询语言(SQL)为基础的大型关系数据库，通俗地讲它是用方便逻辑管理的语言操纵大量有规律数据的集合。是目前最流行的客户/服务器（Client/Server）体系结构的数据库之一，由于 Oracle 数据库的物理设计比较繁琐，且对使用性能影响较大，所以这里给一个相对来说比较详细的说明。

1、Oracle 数据库系统的特点

（1） Oracle7.X 以来引入了共享 SQL 和多线索服务器体系结构。这减少了 Oracle 的资源占用，并增强了 Oracle 的能力，使之在低档软硬件平台上用较少的资源就可以支持更多的用户，而在高档平台上可以支持成百上千个用户；

（2） 提供了基于角色(ROLE)分工的安全保密管理。在数据库管理功能、完整性检查、安全性、一致性方面都有良好的表现；

（3） 支持大量多媒体数据，如二进制图形、声音、动画以及多维数据结构等；

（4） 提供了与第三代高级语言的接口软件 PRO*系列，能在 C,C++等主语言中嵌入 SQL 语句及过程化(PL/SQL)语句，对数据库中的数据进行操纵。

加上它有许多优秀的前台开发工具如 POWER BUILD、SQL*FORMS、VISUAL BASIC 等，可以快速开发生成基于客户端 PC 平台的应用程序，并具有良好的移植性；

（5） 提供了新的分布式数据库能力。可通过网络较方便地读写远端数据库里的数据，并有对称复制的技术。

2、存储结构

（1） 物理结构：Oracle 数据库在物理上是存储于硬盘的各种文件。它是活动的，可扩充的，随着数据的添加和应用程序的增大而变化；

（2） 逻辑结构：Oracle 数据库在逻辑上是由许多表空间构成。主要分为系统表空间和非系统表空间。非系统表空间内存储着各项应用的数据、索引、程序等相关信息。

2.1.3 MySql 数据库系统

MySql 数据库是一个使用 C,C++开发的，开放源代码的，快速的（某种程度上是最快的，后面的测试将会证明这一点），多线程的，稳定的数据库。它不完全兼容 ANSI SQL-92。

由于它的开放性，它的安全性比 Oracle 稍差。同时多用户支持能力逊色于 Oracle，SqlServer2000，DB2，Informix 等大型数据库。

2.1.4 Access 系统

Access 系统是微软提供的桌面数据库系统。安装，使用都比较简单，其功能也相对简单。稳定性较差，在多用户，大数据量的情况下很容易崩溃。

2.2 各种数据库系统性能初步测试

由于公认 Access 系统在面对大数据量，多用户的状态下，性能稳定性欠佳，且没有 Access 的 JDBC 驱动，而 ODBC 桥并不支持 JDBC2.0 的许多 API,故本次测试没有将 Access 系统列为测试对象。但在某些测试时候还是针对 Access 做了对比测试。同时，本次测试仅仅考虑相对速度，所以在硬件配置的好坏，对测试结果

没有特别重要的意义，所以这里就不列出了。

2.2.1 多用户系统性能的初步测试

首先创建一个 20 个字段的表，同时通过程序插入 10000 条数据。之后通过 Java 多线程，模拟 50 个客户，从基数为 10 的 Sql 操作规则中随机抽取一条操作规则对对象数据库进行 DML 操作。同时启动另一个线程，针对表中的一条记录执行特定的 Sql 操作(update,select)，计算操作时间。（测试程序见附录 1）

测试结果如表 2-1 所示：

表 2-1 多用户性能测试结果

Table 2-1 the result of performance test with multi-user

数据库	时间	有无抛出异常的线程
Sqlserver	1560ms	无
Oracle	4457ms	无
Mysql	1220ms	无
Access	无法测试	4 个线程抛出异常

2.2.2 单用户系统性能的初步测试

首先创建一个 20 个字段的表，同时通过程序插入 1000 条数据，记录插入时间。（测试程序见附录 1）

测试结果如表 2-2 所示：

表 2-2 单用户系统性能测试结果

Table 2-1 the result of performance test with single-user

数据库	时间	有无抛出异常的线程
Sqlserver	12760ms	无
Oracle	24457ms	无
Mysql	2920ms	无

2.2.3 北化科研系统数据库的选择

由测试结果可以看出 Oracle（测试中系统配置较低，不能充分体现出 Oracle 数据库系统的性能）和 MySql 的性能和稳定性相对而言更高，出于学校科技处给

我们提供正版的 Oracle9i，并且 Oracle 的开发和支持工具相对而言更多，管理更方便，所以，最后，我们决定数据库使用 Oracle 数据库。

2.3 数据库的应用开发模式

2.3.1 客户机/服务器（Client/Server）体系

所谓数据库应用程序，就是能够从数据库管理系统 DBMS 中获得并操作数据的程序。它是目前最流行的计算机程序之一，其应用范围几乎覆盖了所有领域。从简单的数据输入、数据浏览和数据批处理程序到复杂的客户/服务器应用程序都属于数据库应用程序的范畴。不仅如此，各种类型的应用程序都有可能使用数据库，例如在如今非常流行的 QICQ 聊天软件，还有联众游戏里的个人资料保存都使用了数据库技术。为了建立交互站点，需要使用数据库存储来自访问者的信息。还有工业控制方面也需要对相关数据库进行访问和控制。

客户机/服务器（Client/Server）应用程序结构，是为了解决费用与性能的问题而提出来的。它是中和了集中式结构和文件服务器结构的优缺点发展而成的。人们主要使用过三种系统结构：集中式、文件服务器和客户机/服务器结构。

1、集中式结构

集中式结构也称为哑终端/主机（Dumb Terminals/Mainframe）模式。在客户机和服务器上传递的是终端的按键信息和服务器返回的字符。应用程序和数据都驻留在服务器上，通信中几乎不存在瓶颈，即使主机和终端相隔很远。应用程序的开发和维护也是集中式的，为控制和安全提供重要的措施。系统的管理如备份和数据维护也是集中处理的。同时虽然这种集中式处理大大提高了服务器能力，但是费用很高。

2、文件服务器结构

20 世纪 80 年代，个人电脑开始兴起，它们都带有一定的计算能力。为了充分利用比较便宜的资源，提出了文件服务器的概念。所有的应用程序都在客户机上执行，文件服务器只提供文件服务。文件服务器就像很多用户计算机共享的硬盘，它通过网络与各计算机相连。在应用程序需要寻找数据库中的某个特定的记录时，它需要从文件服务器中检索一组物理块，直到应用程序在数据流中找到所需要的记录后，再把它传输给用户。

这样每次请求的有效性（请求的数据与返回的数据之比）显著依赖于应用程

序的合理性，如果采用好的索引策略，熟练的程序员可以写成文件服务器下支持数以万计的记录的应用程序。如果用户不了解如何操作索引以提高性能的话，将会返回一大批数据，而只有很小的一部分是有用的，造成系统资源的浪费，可能使网络陷于通信阻塞中。

3、客户机/服务器（Client/Server）体系

客户机/服务器（Client/Server）体系的出现，正好融合了上述两种模式的优点：大型机的绝对功能和集中管理以及文件服务器的低费用和良好的处理平衡。同时也尽可能的避免了它们的缺点。

Client/Server 体系的特点是，把应用程序分给客户机和服务器运行，在客户机和服务器上的应用程序协调工作以完成特定的任务。客户机/服务器结构需要两个实体来完成一个进程。客户机向服务器发出请求，服务器为客户机提供完成这个请求的服务。一个 SQL 的查询过程是这样进行的：客户机的应用程序发出一个 SQL 查询请求，服务器处理这个查询，并把查询的结果返回给客户。

Client/Server 结构的典型可以描述成图 2-1 所示：

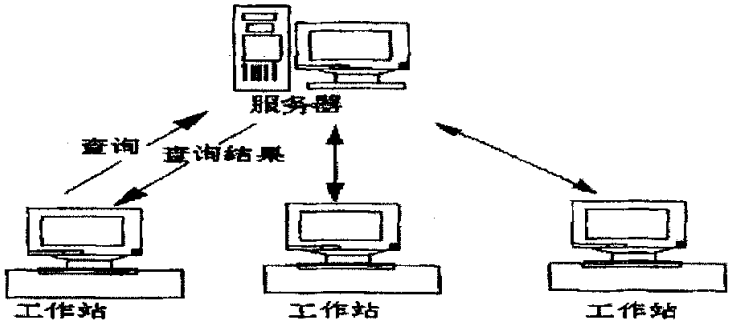


图 2-1 客户机/服务器结构示意图
Fig.2-1 the structure of Client/Server

一个成功的 Client/Server 环境可带来的好处如下：

（1） 地理上的分散，使有限的资源得到最大限度和合理的应用。

（2） 可以在不同的应用程序之间通信。使用不同通信协议的两客户机应用程序之间不能直接通信，但可通过同时支持两种协议的服务器进行通信。

（3） 可以使应用程序的各个部分独立开发，每个部分可以修改、替换却不影响其他部分。

（4） 响应速度取决于服务器，因此只要服务器功能强大而且速度极快，并且如果服务器中数据库引擎使用了缓冲机制时，所有用户均可受益。

（5） 数据完整性和安全性。数据库在服务器上，用户不必担心硬件故障或掉电。另外，数据库管理员能够更好地规定访问权限和数据操作权限。

（6） 减轻网络负担。服务器进程只把响应结构集通过网络传回来而非大的文

件 I/O 块，因此网络负担大大减轻。

这些优点使得 Client/Server 程序设计在 20 世纪 90 年代被许多企业和程序开发人员采用。

但是，Client/Server 体系也存在着不足之处：

（1） 维护较麻烦，需要同时维护服务器端和客户机端程序。

（2） 对于不同的客户机平台需要开发不同的应用程序。

（3） 如果服务器数据库数据的结构改变了，则必须对客户机端一一作相应的修改。

随着 Internet/Intranet 的发展，作为对 Client/Server 体系的扩充和改进，浏览器/服务器体系（Browser/Server）得到了广泛的应用。

2.4.2 浏览器/服务器（Browser/Server）体系

早期的 Web 服务只能处理简单的、静态的 HTML 文本，不利于实现动态的信息发布和管理。要管理动态的信息，必须使 WWW 和数据库技术结合起来，即在浏览器一端采用交互式 Web 页面，服务器端连接数据库服务器，构成浏览器/服务器（Browser/Server）模型。B/S 模式是目前在 Internet/Intranet 网络平台上最流行的运行模式。其结构如图 2-2 所示。

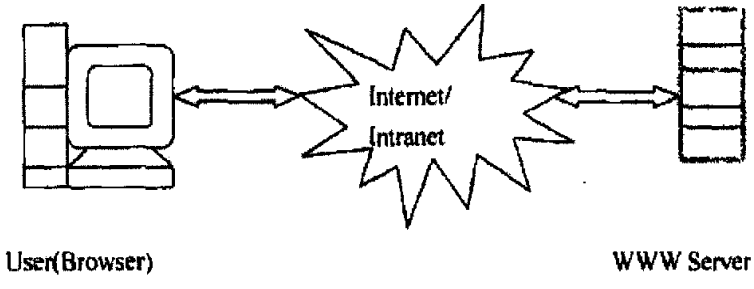


图 2-2 浏览器/服务器体系结构简单示意图
Fig.2-2 the simple structure of Browser/Server

它和 C/S 模式有很多相似之处，但浏览器是通用的用户界面，不需要在浏览器端安装用户应用程序，服务器负责提供用户需要的信息，但要访问网络数据库中的信息还需要进行某种扩展。信息系统采用此种模式可以使数据处理、内部信息（Intranet）的浏览和外部信息（Internet）的浏览界面完全一致，方便用户使用。同时，由于浏览器端不安装用户应用程序，可大大降低运行维护费用。表 1-1 是 C/S 和 B/S 之间的比较。

表 2-1 B/S 与 C/S 结构实现 Intranet 的比较
Table 2-1 the comparison of implement with B/S or C/S

比较内容	Client/Server 结构	Browser/Server 结构
对客户端 硬件配置要求	根据系统规模 需要较高的硬件配置	需要一般的硬件配置 没有特殊要求
要专门安 装客户端软件	需要安装 专门的客户端软件	客户端只需要浏览器 不需要安装专门客户端软件
系统部署代价	部署代价 与信息点的多少成正比	与信息点的多少无关， 部署代价比较小
系统可维护性	系统维护复杂	良好的系统维护性， 代价很小
实现功能 的复杂程度	能够根据用户要求， 定置复杂的应用	可以定置大型复杂的 系统应用，成本较高
系统功能 的可扩展性	一般系统定置好， 扩展性较差	具有良好的系统扩展性， 可随用户需求增加新的功能
系统使用 的难易程度	一般要经过专门 的培训才能使用	不需要专门的培训
数据控制的灵活性	对操纵数据的控制灵活	对操纵数据的控制不太灵活
与 Internet 的集成	与 Internet 较难集成	与 Internet 的集成平滑， 代价为零
未来技术 的发展趋势	不是未来技术发展的主流	是未来技术发展的主流

2.4.3 两层结构和三层结构

两层 Client/Server 结构由客户机和服务器组成，服务器是指运行在网络中的一台或多台计算机上的数据库系统，所以又称为数据库服务器。数据库服务器为客户提供数据查询、更新、索引、优化以及安全控制等。客户通过数据库接口，向服务器提交 SQL 请求，以得到服务器的服务。

而在三层 Client/Server 系统中，整个系统有三个部分组成：

（1） 客户机：安装应用程序，负责与用户的交互和与应用服务器的交互。

（2） 应用服务器：负责处理应用逻辑，接受客户端应用程序的请求。根据应用逻辑将此请求转化为数据库请求后，与数据库服务器交互。最后将从数据库服务器中得到的结果传送给客户机端应用程序。

（3） 数据库服务器：负责根据应用服务器发来的请求进行数据库操作，并将数据库操作的结果传送给应用服务器。

三层应用软件的用户界面与应用逻辑分别位于客户机和应用服务器上，应用逻辑为所有用户共享，应用逻辑可以划分为多个模块，而且每个模块可以同时响

应多个客户机端应用程序的请求。这样，“胖”客户机就变成了“瘦”客户机，开发和维护工作就向服务器方转移，因此较为方便。

同样地，Browser/Server 也可以有两层结构和三层结构。两层结构由浏览器和服务器组成，而三层结构由三部分组成：

（1） 安装了浏览器的客户机。

（2） 中间层服务器：由 Web 服务器、中间层的应用程序和数据库接口程序组成。

（3） 数据库服务器：管理和维护数据库，接受来自中间层的应用程序的 SQL 请求，并将结果返回给中间层。

B/S 三层结构如图 2-3 所示：

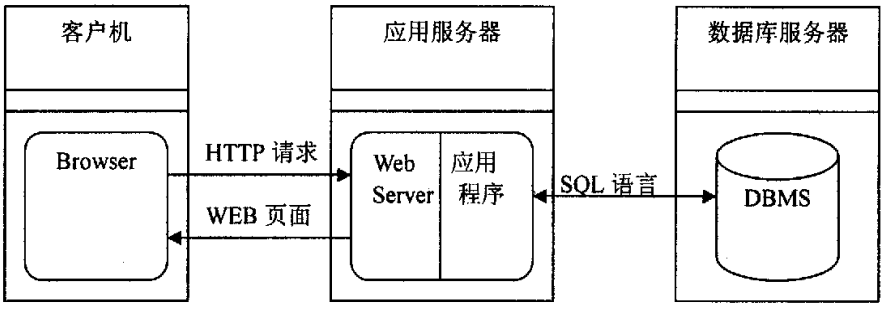


图 2-3 B/S 三层结构图

Fig.2-3 the three structure of B/S

2.5 小结

北化科研系统采用了 B/S 结构的应用开发模式，其数据库部分利用软件工程化思想和方法总体上采用结构化生命周期法进行系统规划和数据库开发。涉及到的数据库相关内容总结如下：

1、数据库的逻辑设计采用 SqlServer2000 系统(主要是使用其数据分析工具)；

2、物理设计采用 Oracle 数据库系统；

3、不建议用户采用 Access 数据库，不明确支持；

4、对于用户量较小，数据量较小的用户推荐使用 MySql 数据库；

5、所有 sql 操作都符合 ANSI SQL-92 标准。

第三章　科研管理信息系统数据库的设计

数据库设计是建立数据库及其应用系统的技术，使信息系统开发和建设中的核心技术，具体说，数据库设计是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能够有效地存储数据，满足各种用户的应用要求（信息要求和处理要求）[22]。

本章首先对数据库设计进行简单论述，然后介绍数据库设计的一般步骤，并在其中论述北化科研系统数据库设计的各个步骤进行论述。

3.1 数据库设计概述

数据库（Database）是数据管理的最新技术，具有数据结构化、最低冗余度、较高的程序与数据独立性、易于扩充、易于编制应用程序等优点。所谓数据库是指长期存储在计算机内的、有组织的、可共享的数据集合[23]。数据库中的数据按一定的数据模型组织、描述和储存，具有较小的冗余度、较高的数据独立性和易扩展性，并可为各种用户共享。数据库的应用已经越来越广泛了，不仅大型计算机及中小型计算机，甚至微型计算机都用先进的数据库技术来保证数据的整体性、完整性和共享性。

数据库设计（Database Design）是将业务对象转换为表和视图等数据库对象的过程[24]。数据库设计是数据库应用系统开发过程中的首要的和基本的内容。数据库是信息系统的核心和基础。它把信息系统中的大量数据按照一定的模型组织起来，提供存储、维护、检索数据的功能，使信息系统可以方便、及时、准确的从数据库中获取所需的信息。一个信息系统的各个部分能否紧密的结合在一起以及如何结合，关键在数据库。因此必须对数据库进行合理设计。

3.1.1 数据库与信息系统

从使用者角度看，信息系统是提供信息、辅助人们对环境进行控制和进行决策的系统。数据库是信息系统的核心和基础。它把信息系统中大量的数据按一定的模型组织起来，提供存储、维护、检索数据的功能，使信息系统可以方便、及时、准确地从数据库中获得所需的信息。一个信息系统的各个部分能否紧密地结合在一起以及如何结合，关键在数据库。因此只有对数据库精心合理的逻辑设计和有效的物理设计才能开发出完善而高效的信息系统。数据库设计是信息系统开发和

建设的重要组成部分。

3.1.2 数据库设计的特点

数据库设计既是一项涉及多学科的综合性技术，又是一项庞大的工程项目。有人讲“三分技术，七分管理，十二分基础数据”是数据库建设的基本规律，这是有一定道理的。技术与管理的界面（称之为“干件”）十分重要。数据库建设是硬件、软件和干件的结合。这是数据库设计的特点之一。这里主要讨论软件设计的技术。

数据库设计应该和应用系统相结合，也就是说，整个设计过程中要把结构（数据）设计和行为（处理）设计密切结合起来。这是数据库设计的特点之二。

3.1.3 数据库设计方法

由于信息结构复杂，应用环境多样，在相当长的一段时间内数据库设计主要采用手工试凑法。使用这种方法与设计人员的经验和水平有直接关系，数据库设计成为一种技艺而不是工程技术，缺乏科学理论和工程方法的支持，工程的质量难以保证，常常是数据库运行一段时间后又不同程度的发现各种问题，增加了系统维护的代价。经过人们的不断探索，提出了各种数据库设计方法，这些方法运用软件工程的思想和方法，提出了各种设计准则和规程，都属于规范设计法。

规范设计法中比较著名的有新奥尔良（New Orleans）方法。它将数据库设计分为四个阶段：需求分析（分析用户需求）、概念设计（信息分析和定义）、逻辑设计（设计实现）和物理设计（物理数据库设计）。其后，S.B.Yao 等又将数据库设计分为五个步骤。又有 L.R.Palmer 等主张把数据库设计当成一步接一步的过程，并采用一些辅助手段实现每一过程。

基于 E-R 模型的数据库设计方法，基于 3NF（第三范式）的设计方法，基于抽象语法规范的设计方法等，是在数据库设计的不同阶段上支持实现的具体技术和方法。

规范设计法从本质上看仍然是手工设计方法，其基本思想是过程迭代和逐步求精。

数据库工作者和数据库厂商一直在研究和开发数据库设计工具，现在数据库设计工具已经实用化和产品化。例如，Design 2000 和 PowerDesigner 分别是

ORACLE 公司和 SYBASE 公司推出的数据库设计工具软件。这些工具软件可以自动地或辅助设计人员完成数据库设计过程中的很多任务。人们已经越来越认识到自动数据库设计工具的重要性。特别是大型数据库的设计需要自动设计工具的支持。人们也日益认识到数据库设计和应用设计应该同时进行。目前许多计算机辅助软件工程（Computer Aided Software Engineering，简称 CASE）工具已经开始强调这两个方面。

3.2 北化科研系统的数据库设计

明确北化科研系统的功能和结构，是进行系统数据库设计的前提。论文本节首先对北化科研系统的功能和结构进行了简单的介绍，然后分析了数据库设计的基本步骤，并结合数据库设计基本步骤对北化科研系统的数据库设计各个步骤进行了论述。

3.2.1 北化科研系统功能和结构概述

1、北化科研系统的功能

北化科研系统需要满足两个方面的功能：一是支持高校的科研管理工作，管理人员可以通过系统对科研机构、科研人员、科研项目、科研经费和科研成果等进行综合管理以及对教职工的年终考评和职称评审提供重要的数据。二是支持日常的统计工作及教育部要求的年报统计工作。从功能上划分，可以分为管理员端和客户端。管理员端为我校科技处科研管理人员使用，也就是通常意义上的内网，在这个内网中，根据科技处办公的电脑的 IP 地址限制访问权限，基本上只有科技处的几台机器可以访问内网，这样可以在安全性上作第一级的保障。内网中的功能是最强的，最全的，基本上客户端的功能也包含在内网之内。内网需要维护基础数据、业务操作、报表的管理、数据查询、可视化操作、权限管理、业绩统计、报表打印、数据导入和导出，基本上涵盖了整个系统所有可用的功能。当然，在内网中，也有配置权限组，只有处于最高权限组的用户才有权力使用所有的功能，而横向或者是纵向的管理员都只有相应模块的管理权限。客户端是对整个 Internet 开放的，由于其开放性，所以对系统的安全性要求更高。同时，具有权限的用户（学校的教职员工）也尽量只能看到自己的信息，而且对自己的信息的操作权限也有严格的限制，这样可以最大程度上保证数据的安全性和系统的安全性。各个模块具有相对独立性，同时也相互关联。系统基本功能模块结构如图 3-1 所示：

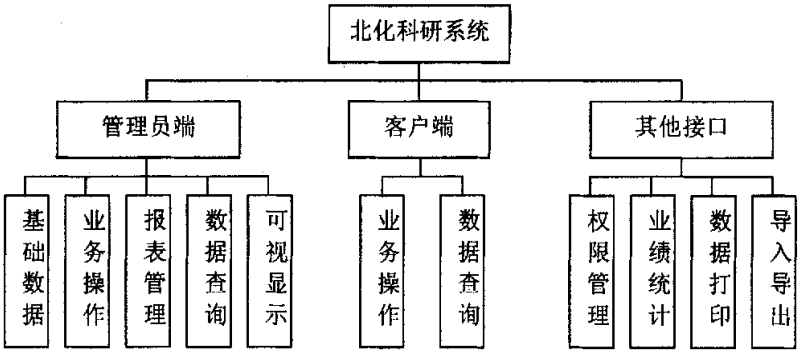


图 3-1 北化科研系统功能模块图
Fig.3-1 the function models of the system

2、北化科研系统的逻辑结构

为满足系统的功能需求，方便用户和系统管理，北化科研系统采用四层结构，分别是客户端、表现层、业务层和数据层，系统的逻辑结构如图 3-2 所示：

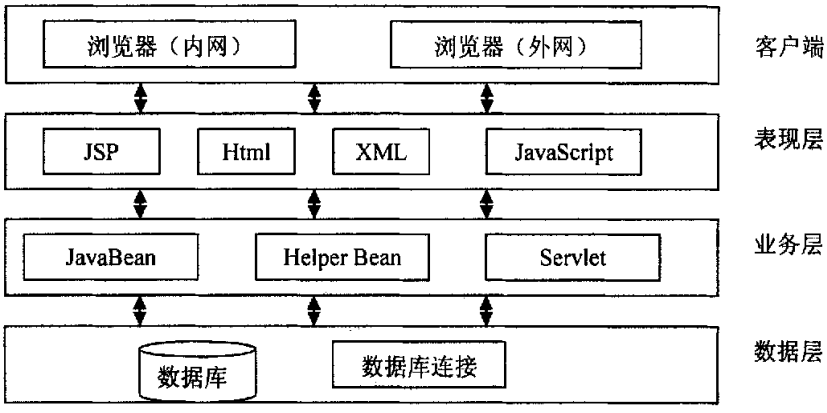


图 3-2 北化科研系统逻辑结构
Fig.3-2 the logic structure of the sytem

客户端：由于选择的是 B/S 结构，所以访问系统的界面是浏览器，而且分为两个系统，内网的浏览器的访问需要 IP 的限制，外网的访问没有限制。

表现层：表现层主要是业务数据的表现形式，如上图，可能是动态的 JSP 页面，也有可能是静态的 Html，使用 JavaScript 增强动态性。使用 XML 进行一些配置工作。

业务层：业务层分为业务外观层和业务规则层。业务规则层也叫做控制层或者表现服务层，主要隔离用户 UI 和各业务规则，为表现层提供业务数据服务的接口层。业务规则层对各种业务规则和逻辑的判断处理、验证处理。调用数据层的接口（JDBC），进行数据访问和存储。

数据层：数据层为上一层提供数据服务，利用数据库连接池管理数据库连接，用 JDBC 执行 SQL 语句或者调用存储过程进行数据的处理。

3.2.2 数据库设计的基本步骤

在理解了北化科研系统的功能和结构的基础上，我们再来论述数据库设计的基本步骤。按照规范化设计的方法，考虑数据库及其应用系统开发全过程，将数据库设计分为以下六个阶段（如图 3-3 所示）：

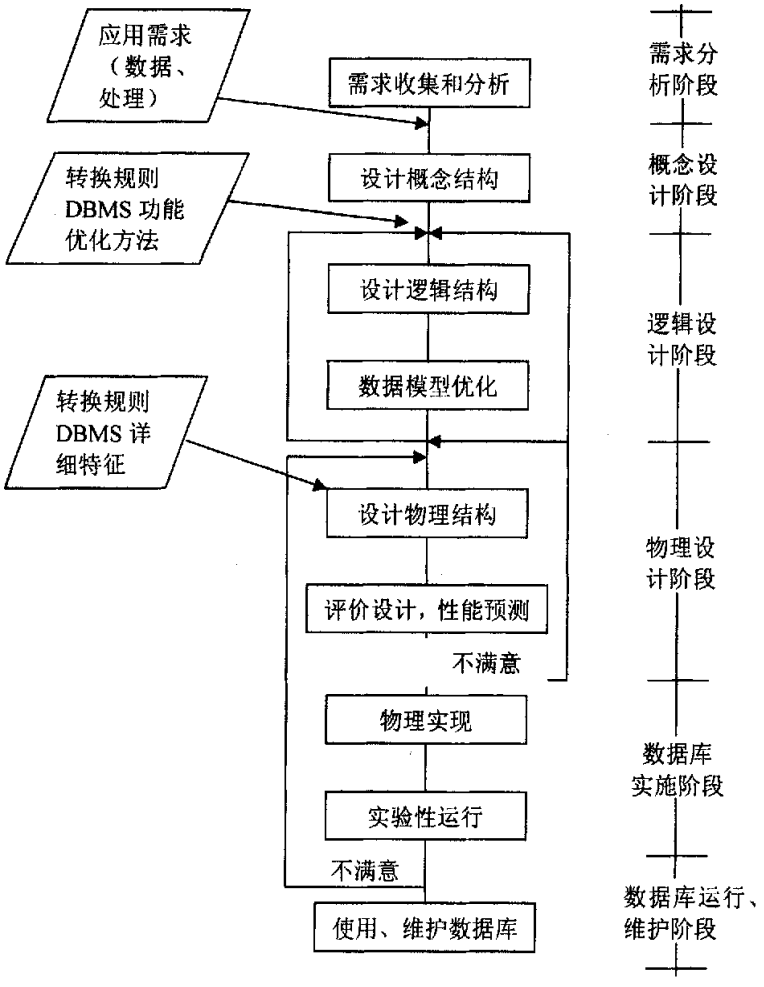


图 3-3　数据库设计步骤

Fig.3-3　the steps of database design

1、需求分析阶段

进行数据库设计首先必须准确了解与分析用户需求（包括数据与处理）。需求分析是整个设计过程的基础，是最困难、最耗费时间的一步。作为低级的需求分析是否做得充分与准确，决定了在其上构建数据库大厦的速度与质量。需求分析做得不好，甚至会导致整个数据库设计返工重做。

2、概念结构设计阶段

概念结构设计是整个数据库设计的关键，它通过对用户需求进行综合、归纳与抽象，形成一个独立于具体 DBMS 的概念模型。

3、逻辑结构设计阶段

逻辑结构设计是将概念结构转换为某个 DBMS 所支持的数据模型，并对其进行优化。

4、数据库物理设计阶段

数据库物理设计视为逻辑数据模型选取一个最合适应用环境的物理结构（包括存储结构和存取方法）。

5、数据库实施阶段

在数据库实施阶段，设计人员运用 DBMS 提供的数据语言及其宿主语言，根据逻辑设计和物理设计的结果建立数据库，编制与调试应用程序，组织数据入库，并进行试运行。

6、数据库运行和维护阶段

数据库应用系统经过试运行后即可以投入正式运行。在数据库系统运行过程中必须不断地对其进行评价、调整与修改。

设计一个完善的数据库应用系统是不可能一蹴而就的，它往往是上述六个阶段的不断反复。

需要指出的是，这个设计步骤既是数据库设计的过程，也包括了数据库应用系统的设计过程。在设计过程中把数据库的设计和对数据库中数据处理的设计紧密结合起来，将这两个方面的需求分析、抽象、设计、实现在各个阶段同时进行，相互参照，相互补充，以完善两方面的设计。事实上，如果不了解应用环境对数据的处理要求，或没有考虑如何去实现这些处理要求，是不可能设计一个良好的数据库结构的。按照这个原则，设计过程各个阶段的设计描述，可用表 3-1 概括地给出。

表 3-1 数据库结构设计阶段

Table.3-1 the structure design phase of database

设计阶段	设计描述	
	数据	处理
需求分析	数据字典、全系统中数据项、数据流、数据存储的描述	数据流图和判定表（判定树）、数据字典中处理过程的描述
概念结构设计	概念模型（E-R 图） 数据字典	系统说明书包括： （1）新系统要求、方案和概图 （2）反映新系统信息流的数据流图
逻辑结构设计	某种数据模型 关系　非关系	系统结构图（模块结构）
物理设计	存储安排 方法选择 存取路径建立 分区 1 分区 2	模块设计 IPO 表 IPO 表…… 输入： 输出： 处理：
实施阶段	编写模式 装入数据 数据库试运行 Creat…… Load……	程序编码 变异联结 测试 Main（） …… if…… then end
运行维护	性能监测、转储/恢复 数据库重组和重构	新旧系统转换、运行、 维护（修正性、适应性、改善性维护）

按照这样的设计过程，数据库结构设计的不同阶段形成数据库的各级模式，如图 3-4 所示。需求分析阶段，综合各个用户的应用需求；在概念设计阶段形成独立于机器特点，独立于各个 DBMS 产品的概念模式，在科研管理信息系统中是 E-R 图；在逻辑设计阶段将 E-R 图转换成具体的数据库产品支持的数据模型，如关系模型，形成数据库逻辑模式；然后根据用户处理的要求、安全性的考虑，在基本表的基础上再建立必要的视图（View），形成数据的外模式；在物理设计阶段，根据 DBMS 特点和处理的需要，进行物理存储安排，建立索引，形成数据库内模式。

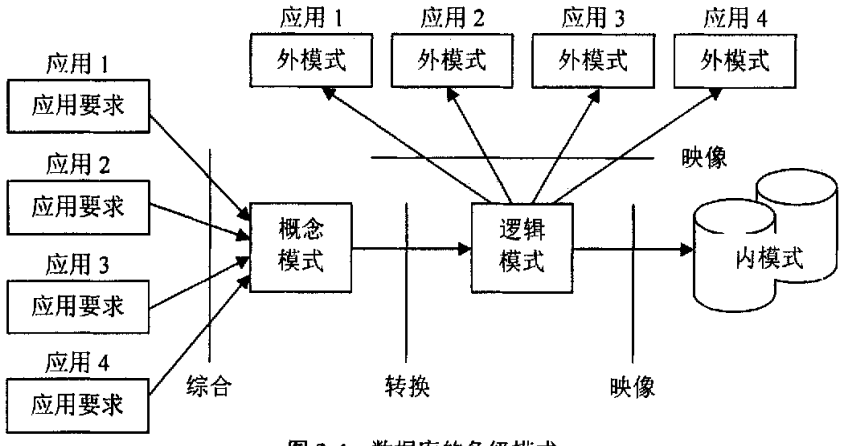


图 3-4 数据库的各级模式
Fig.3-4 the all levels mode of database

3.2.3 北化科研系统数据库设计的需求分析

3.2.3.1 数据库设计的需求分析

数据库的需求分析简单地说就是分析用户的要求。需求分析是设计数据库的起点，需求分析的结果是否准确地反映了用户的实际要求，将直接影响到后面的各个阶段的设计，并影响到设计结果是否合理和实用。

1、需求分析的任务

需求分析的任务[2]是通过详细调查现实世界要处理的对象（组织、部门、企业等），充分了解原系统（手工系统或计算机系统）工作概况，明确用户的各种需求，然后在此基础上确定新系统的功能。新系统必须充分考虑今后可能的扩充和改变，不能仅仅按当前应用需求来设计数据库。

需求分析[25]是为了了解和掌握数据库应用系统开发对象（用户）的工作业务流程和每个岗位、每个环节的职责，了解和掌握信息从开始产生或建立，到最后输出、存档或消亡所经过的传递和转换过程，了解和掌握各种人员在整个系统活动过程中的作用；通过同用户充分地交流和沟通，决定哪些工作应由计算机来做，哪些工作应由手工来做，决定各种人员对信息和处理各有什么要求，对视屏操作界面和报表输出格式各有什么要求，对数据（信息）的安全性（保密性）和完整性各有什么要求，等等。

2、需求分析的目的

需求分析调查的重点是“数据”和“处理”，通过调查、收集和分析，获得用

户对数据库的如下需求：

（1） 信息需求：指用户需要从数据库中获得信息的内容与性质。由信息要求可以到处数据要求，即在数据库中需要存储那些数据；

（2） 处理需求：指用户需要完成什么处理功能。明确用户对数据有什么样的处理要求，从而确定数据之间的相互关系；

（3） 安全性与完整性要求。

3、需求分析的方法

进行需求分析首先是调查清楚用户的实际要求，与用户达成共识，然后分析与表达这些需求。

在调查过程中，可以根据不同的问题和条件，使用不同的调查方法。常用的调查方法有：

（1）跟班作业。通过亲身参加业务工作来了解业务活动的情况。这种方法可以比较准确地理解用户的需求，但比较耗费时间。

（2）开调查会。通过与用户座谈来了解业务活动情况及用户需求。座谈时，参加者之间可以相互启发。

（3）请专人介绍。

（4）询问。对某些调查中的问题，可以找专人询问。

（5）设计调查表请用户填写。如果调查表设计得合理，这种方法是很有效，也易于为用户接收的。

（6）查阅记录。查阅与原系统有关的数据记录。

做需求调查时，往往需要同时采用上述多种方法。但无论使用何种调查方法，都必须有用户的积极参与和配合。调查了解了用户的需求以后，还需要进一步分析和表达用户的需求。

在需求分析阶段，要特别注意以下两点：

- 需求分析阶段的一个重要而困难的任务是收集将来应用所涉及的数据，设计人员应充分考虑到可能的扩充和改变，使设计易于更改，系统易于扩充，这是第一点。
- 必须强调用户的参与，这是数据库应用系统设计的特点。数据库应用系统和广泛的用户有密切的联系，许多人要使用数据库。数据库的设计和建立又可能对更多人的工作环境产生重要影响。因此用户的参与是数据库设计不可分割的一部分。在数据分析阶段，任何调查研究没有用户的积极参与是寸步难行的设计人员应该和用户取得共同的语言，帮助不熟悉计算机的用户建立数据库环境下的共同概念，并对设计工作的最后结果承担共同的责任。

3.2.3.2 北化科研系统数据库的需求分析

北化科研系统的数据库的需求分析是在用户提供需求说明书的基础上，通过对系统功能和结构的理解，经过我们与用户面对面的进行分析得来的。由需求分析我们可以得到北化科研系统的数据流程图如下（图 3-5）：

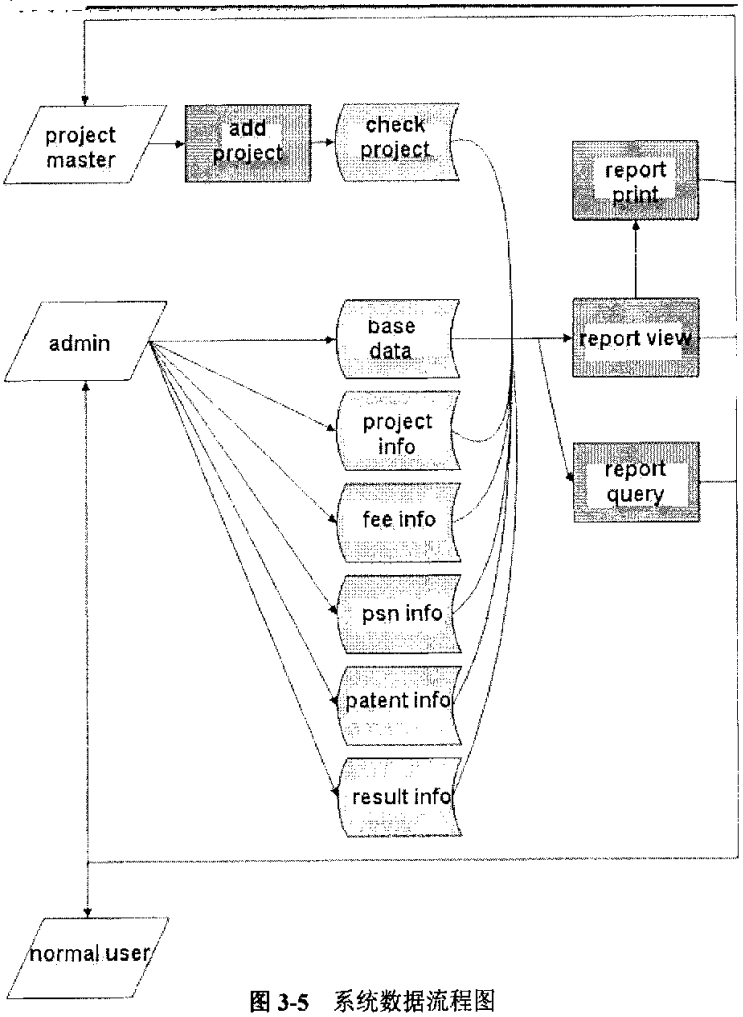


图 3-5 系统数据流程图

Fig.3-5 the data flow chart of the system

由需求分析结合系统的功能我们可以得到北化科研系统基础数据、合同管理、经费管理、论文管理、成果管理等模块的数据项和数据结构如下：

（1） 基础数据

基础数据模块主要是对一些基本的数据信息进行分类处理，其中包含很多部分。例如：学院、机构、重点研究基地、人、职称、学历、项目来源、项目分类、学科分类、结算办法、退费比例、鉴定方式、奖励种类、奖励等级、发

明类型、论文分类等31个基本组成类别。下面简单列出其中的一部分：

- 学院：院所编号、院所名称、科研负责人、成立时间、联系电话、电子邮箱、邮编、地址、传真、网址等。
- 机构：机构编号、机构名称、批准部门、机构类型、学科分类、组成形式、所属学院、成立时间、机构负责人、联系电话、电子邮箱、网址、服务的国民经济行业、从业人员数 (其中按照学历包括：博士毕业、硕士毕业、本科毕业、其他)（其中按照职称包括：正高级职称、副高级职称、中级职称、初级职称、其他)、实验室地址、实验室面积、固定资产、主要研究方向、机构评估情况（年度)、 项目参加人员培养：博士生（人)、硕士生（人)。
- 重点研究基地：重点研究基地编号、重点研究基地名称、批准部门、成立时间、重点研究基地负责人、基地联系电话、基地电子邮箱、基地网址、机构类型、组成形式、学科分类、服务的国民经济行业、从业人员数 (其中按照学历包括：博士毕业、硕士毕业、本科毕业、其他)（其中按照职称包括：正高级职称、副高级职称、中级职称、初级职称、其他)、实验室地址、实验室面积、固定资产、主要研究方向、培养博士生人数、培养硕士生人数。
- 人：姓名、是否职工、性别、出生日期、民族、党派、技术职务、职称、学历、最后学位、所在院所、所在机构、所在重点研究基地、联系电话、手机、电子邮箱、学科分类、外语语种、现在身份（在职、离退休）主要学习及工作经历、专业特长、主要研究开发领域及成果、进入机构时间、离开机构时间、在机构工作年限、团队协作（是or 否）、备注。
- 职称：职称编号、职称名称、职称代码。

（2）合同管理

合同管理模块中主要是对合同的一些基本信息进行管理，数据项和数据结构简单分析如下：

- 合同：项目类型（纵向、横向)、项目状态（立项、申请)、合同总数（数据库中现有的合同数量)、合同号、合同名称、合同金额（合同总金额、我校金额、转出经费)、实到经费（实到总经费、实到我校经费、实到转出经费)、合同起始时间、合同截止时间、项目来源、项目分类、服务的国民经济行业、学科分类、合同类别（基础研究、应用研究等)、完成形式（独立、合作)、合同状态（进行、结项、中止、延期、完成)、是否归档、归档时间、最终结题方式、结题时间、合同参加人员情况（人员所属院系、人员姓名、职称、学历等)、经费预算情况（到款年度、到款金额)、研究的内容及进度、项目完成情况评价等。

（3）经费管理

经费管理模块主要是按照经费的处理流程来进行分析设计，分为到款、入账、结算、转账、退费和分配几个小的模块，其数据项和数据结构如下：

- 到款（主要是负责记录到达学校的科研经费）：到款编号、到款日期、来款单位、到款日期、来款金额、备注。
- 入账（主要是负责记录到达学校的经费所入账号的情况）：入账单编号、入账日期、入账账户名称、到款日期、来款单位、入账金额、备注。
- 结算（主要是解决到达经费如何分配、分配给谁的问题）：结算单编号、结算日期、合同号、项目名称、院系单位、经费负责人、来款单位、结算办法、经费卡号、结算金额、备注。
- 转账（主要是处理与校外合作项目的经费问题）：转账单编号、转账日期、来款单位、所属院系、项目负责人、合同号、经费卡号、转账金额、转入单位、转入银行、转入账号、备注。
- 退费（主要是处理项目中出现固定资产时的情况）：退费单编号、退费日期、经费本号、经费负责人、院所单位、退费办法、退费比例、仪器编号、购置费金额、退费金额、备注。
- 分配（主要是处理项目有多个参加人的经费问题）：分配单编号、分配日期、项目负责人、合同号、项目来源、结算金额、结算日期、分配金额、经费卡号、备注。

（4）论文管理

论文管理模块包括著作论文、期刊论文和会议论文三个小的模块。其数据项和数据结构如下：

- 著作论文：著作名称、出版单位、出版日期、著作类别（专著、编著、译著、科普、教材，其中教材分为精品教材与否以及级别，级别指的是国家级获省部级）、发行代码、字数（单位：千字）、备注以及作者信息（校内/校外作者、作者身份、姓名、所在单位、机构、职称、学历等）等。
- 期刊论文：论文分类（自然科学、人文社科）、出版范围（国内、国外）、期刊类别（公开、内部）、核心期刊（是、否）、刊物名称（如为核心期刊，请从系统中选取，否则录入）、出版单位、论文名称、发表年度、卷、期、页、备注以及作者信息（校内/校外作者、作者身份、姓名、所在单位、机构、职称、学历等）等。
- 会议论文：论文名称、会议名称、主办单位、举办日期、举办地点、举办单位、会议性质（国内、双边国际、多边国际）、报告类别（大会、特邀、分组）、是否有论文集、是否被 ISTP 收录（由管理人员操作）、收录年度、备注以及作者信息（校内/校外作者、作者身份、姓名、所在单位、机构、

职称、学历等）等。

（5）成果管理

成果管理模块包括鉴定、奖励和专利信息三个小模块。其数据项和数据结构如下：

- 鉴定管理：鉴定成果编号、合同号、项目负责人、项目名称、所在院所、所在机构、所在重点研究基地、协作单位、项目起始时间、项目终止时间、项目来源、项目分类、鉴定地点、主持部门、组织单位、鉴定时间、鉴定方式、成果水平、成果证号和参加人员等。
- 奖励管理：获奖编号（根据获奖时间自动生成）、授奖时间、获奖人、获奖项目名称 学科分类、所在院所、所在机构、所在重点研究基地、授奖单位、奖励级别、奖励种类、奖励等级、奖金、证书号、参加人员（顺序、姓名以及人的基本信息）、本单位在本项奖励中居第几位、完成单位总数、第一完成单位名称等。
- 专利信息：授权专利总数、申请号、申请日期、发明名称、发明类型、是否国际专利、负责人、所在院所、所在机构、所在重点研究基地、授权时间、专利终止时间、证书号、发明人（顺序、姓名及人的基本信息）、申请单位总数、第一完成单位名称、本单位在本项发明中居第几位、备注等。

3.2.4 北化科研系统数据库的概念结构设计

3.2.4.1 数据库的概念结构设计

数据库的概念结构设计是将分析得到的用户需求抽象为概念模型的过程。即在需求分析的基础上，设计出能够满足用户需求的各种实体以及它们之间的相互关系概念结构设计模型。这样才能更好、更准确地用某一 DBMS 实现这些需求。它是整个数据库设计的关键。

1、概念结构设计的特点[2]

（1） 能真实、充分地反映现实世界及其事物与事物之间的联系，能满足用户对数据的处理要求，是现实世界的一个真实模型。

（2） 易于理解，从而可以用它和不熟悉计算机的用户交换意见，用户的积极参与是数据库设计成功的关键。

（3） 易于更改，当应用环境和应用要求改变时，容易对概念模型修改和扩充。

（4）易于向关系、网状、层次等其他数据模型转换。

数据的概念模型是其他各种数据模型的共同基础，它比数据模型更独立于机器，更抽象，从而更稳定。

2、描述概念模型的工具

（1）E-R 模型

E-R 模型设计是对用户需求和约束详细分析、反复处理和最终实施的过程。设计的最终结果应该准确地描述用户真实的数据环境。所以，正确理解用户的业务规则是设计的基础和关键。设计人员必须详细地调查用户的业务规则和应用需求。在确定实体、属性、联系以及约束条件时，应该考虑最终用户的各种需求，主要包括：应用的功能需求、数据的安全性、数据共享性、数据的完整性、一致性、响应速度等。最终设计的结果要能够满足用户所有应用的数据需求和约束条件，这是检验数据库设计合理性的唯一标准，否则这个设计是毫无价值的。

E-R 模型提供了实体、属性和联系三个抽象概念。这三个概念简单、明了、直观易懂，用以模拟现实世界比较自然，且可方便地转换关系、层次、网状数据模式。用 E-R 表示数据模式时，只需关心有哪些数据（即有哪些实体和属性）以及数据间的关系（实体关系），而不必关心这些数据在计算机内如何表示和用什么表示。

（2）面向对象数据模型

引入面向对象数据模型是因为现实世界中存在着许多含有复杂数据结构的应用领域，例如 CAD 数据、图形数据等，它们需要更高级的数据库技术表达这类信息。此外，这种模型可以根据不同的应用要求对模型进行扩充，又可以重用。

3、设计概念结构通常用四类方法：

（1） 自顶向下：指的是首先定义全局概念结构的框架，然后逐步细化；

（2） 自底向上：指的是首先定义各局部应用的概念结构，然后将它们集成起来，得到全局概念结构；

（3） 逐步扩张：指的是首先定义最重要的核心概念结构，然后向外扩充，以滚雪球的方式逐步生成其他概念结构，直至总体概念结构；

（4） 混合策略：指的是将自顶向下和自底向上相结合，用自顶向下策略设计一个全局概念结构的框架，以它为骨架集成由自底向上策略中设计的各局部概念结构。

其中最常采用的策略是自底向上方法，即自顶向下进行需求分析，然后再自底向上设计概念结构[24]。

5、数据库概念结构设计的主要步骤（如图 3-6 所示）：

（1） 进行数据抽象，设计局部概念模式；

（2） 将局部概念模式综合成全局概念模式；

（3） 评审。

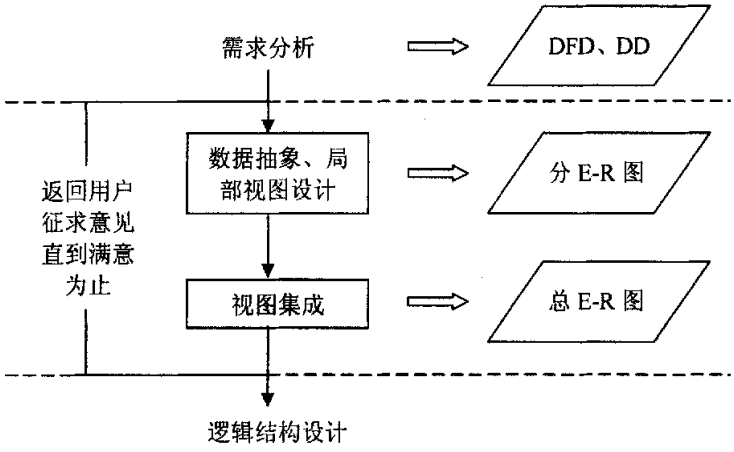
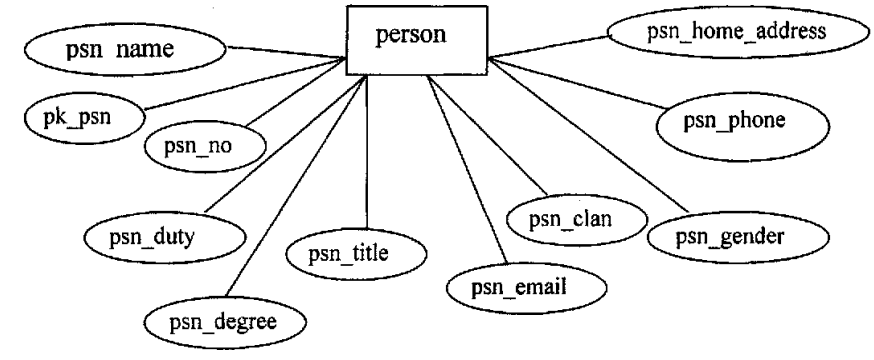


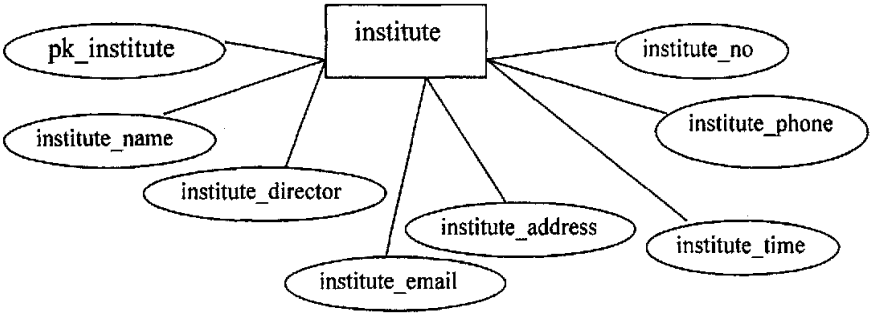
图 3-6 概念结构设计步骤

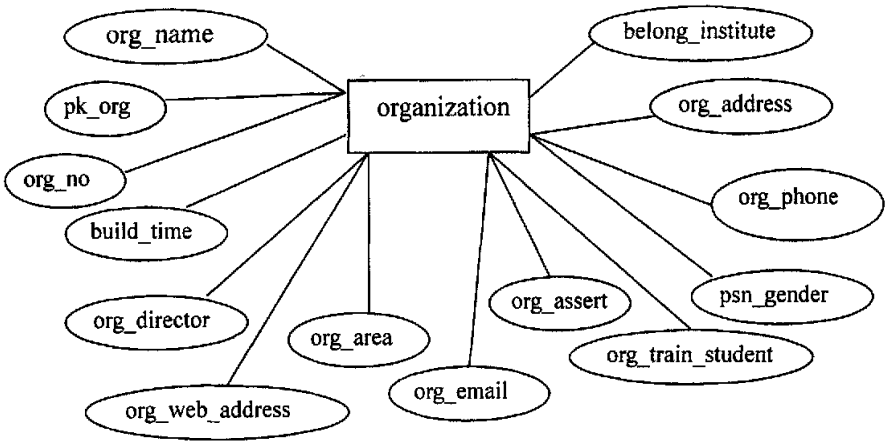
Fig.3-6 the steps of concept structure design

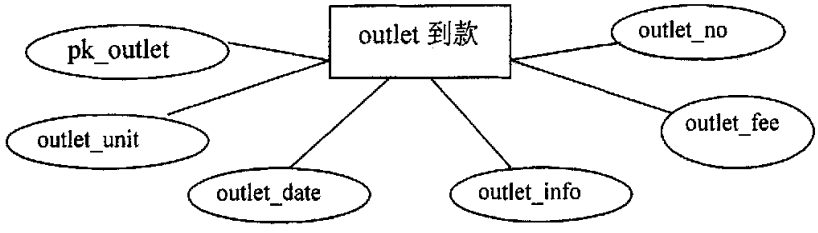
3.2.4.2 北化科研系统数据库的概念结构设计

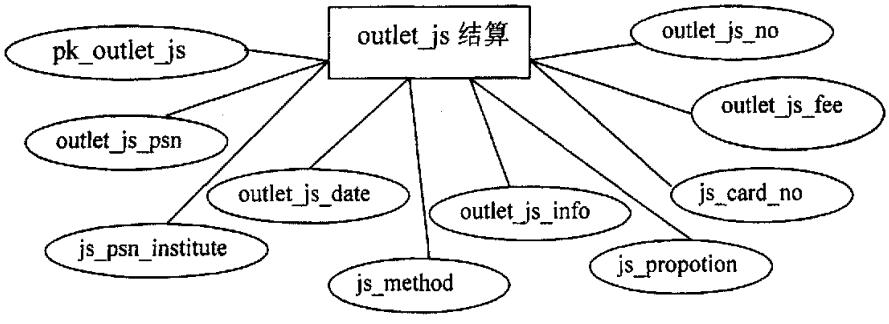
北化科研系统数据库的概念结构设计是在结合了需求分析和系统功能的基础上得到的。我们采用的是 E-R 模型来表示。首先我们设计出分 E-R 模型（图 3-7），再组合得到总 E-R 模型如下（图 3-8）：

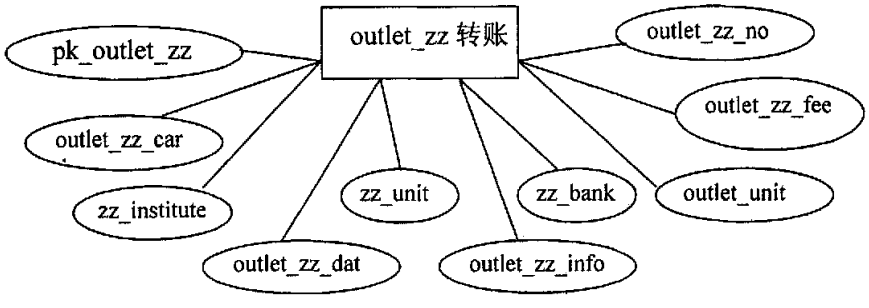





org_name
pk_org
org_no
build_time
org_director
org_web_address
organization
org_area
org_email
org_assert
org_train_student
belong_institute
org_address
org_phone
psn_gender


pk_outlet
outlet_unit
outlet_date
outlet 到款
outlet_info
outlet_no
outlet_fee


pk_outlet_js
outlet_js_psn
js_psn_institute
outlet_js_date
js_method
outlet_js 结算
outlet_js_info
js_propotion
outlet_js_no
outlet_js_fee
js_card_no


pk_outlet_zz
outlet_zz_car
zz_institute
outlet_zz_dat
zz_unit
outlet_zz 转账
outlet_zz_info
zz_bank
outlet_unit
outlet_zz_no
outlet_zz_fee

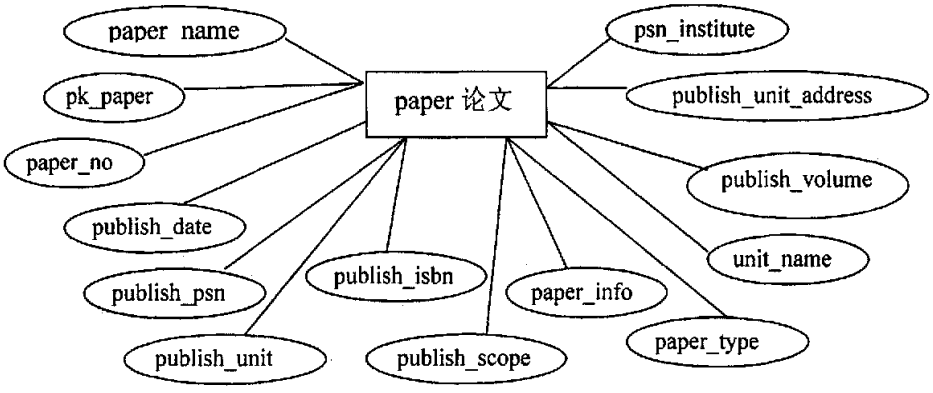


图 3-7 分 E-R 图（部分）

Fig.3-7 the partial E-R chart of concept structure (not all)

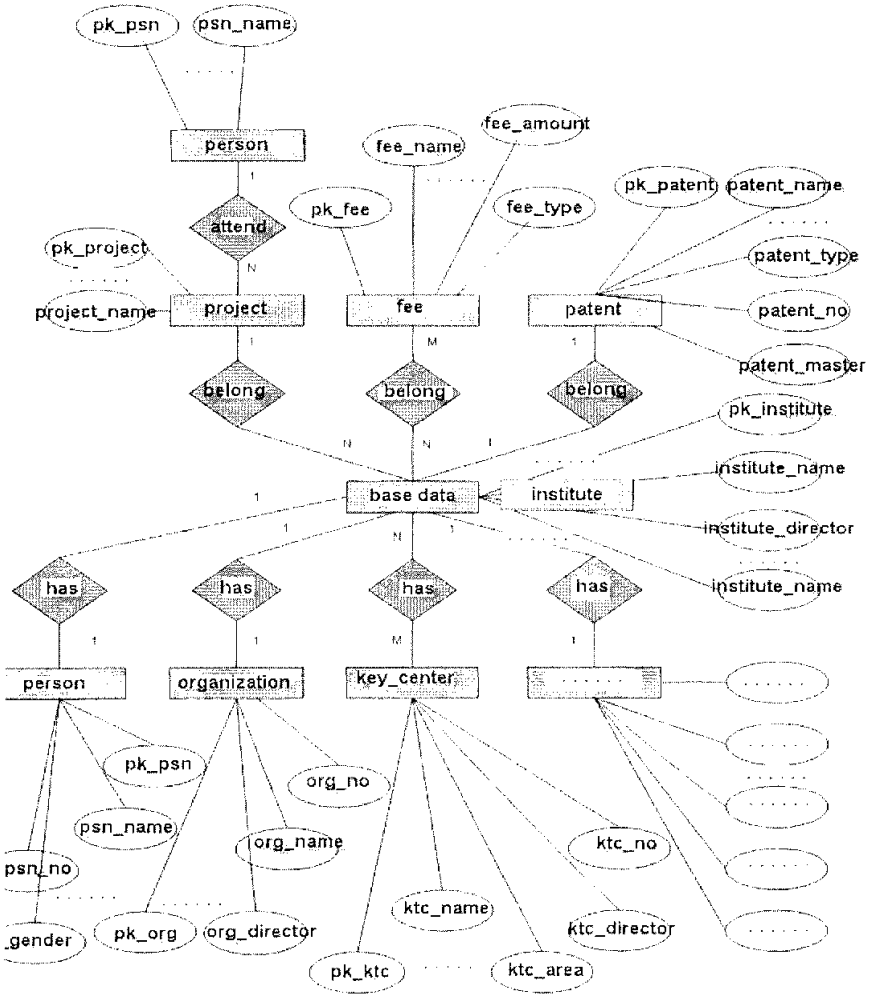


图 3-8 数据库概念结构图（总 E-R 图）

Fig.3-8 the concept structure of database(the full E-R chart)

3.2.5 北化科研系统数据库的逻辑结构设计

3.2.5.1 数据库的逻辑结构设计

E-R 模型表示的是用户的模型。它独立于任何一种数据模型，任何一个具体的数据库管理系统。因此，需要把 E-R 表示的概念模型转换为某个具体的数据库管理系统所支持的数据模型，然后建立用户需要的数据库[26]。

从理论上讲，设计逻辑结构应该选择最适于相应概念结构的数据模型，然后对支持这种数据模型的各种 DBMS 进行比较，从中选出最合适的 DBMS。但实际情况往往是已给定了某种 DBMS，设计人员没有选择的余地。目前 DBMS 产品一般支持关系、网状、层次三种模型中的某一种，对某一种数据模型，各个机器系统又有许多不同的限制，提供不同的环境与工具。所以设计逻辑结构一般要分为三步进行（如图 3-9 所示）：

（1）将概念结构转换为一般的关系、网状、层次模型；

（2）将转换来的关系、网状、层次模型向特定 DBMS 支持下的数据模型转换；

（3）对数据模型进行优化。

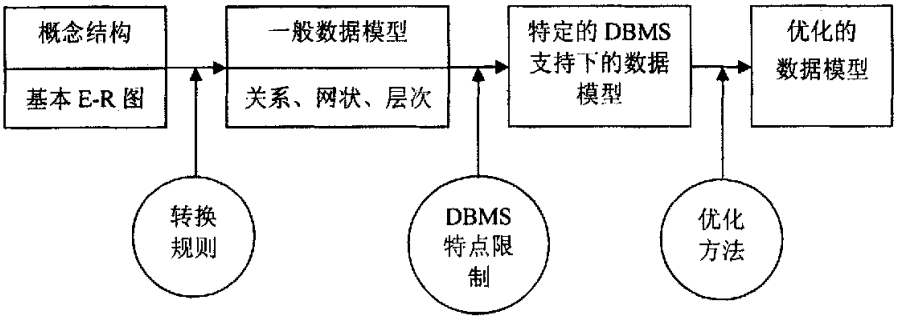


图 3-9 数据库逻辑结构设计的三个步骤

Fig.3-9 the three steps of database's logistic structure design

1、E-R 模型向关系模型转换

（1）转换原则

a）一个实体型转换为一个关系模式，实体的属性就是关系的属性，实体的键就是关系的键。

b）一个联系转换为一个关系模式，与该联系相连的各实体的键以及联系的属性转换为该关系的属性。

（2）具体做法

a）把每一个实体转换为一个关系。

先分析实体属性，从中找出主键和属性间的依赖关系，然后用关系模式来表示。

b）把每一个联系也转换为关系模式。

在这类关系的属性集中，包含了由它联系的各个实体中主键及它本身的属性。

2、数据模型的优化

数据库逻辑设计得结果不是唯一的。为了进一步提高数据库应用系统的性能，还应该根据应用需要适当地修改、调整数据模型的结构，这就是数据模型的优化。关系数据模型的优化通常以规范化理论为指导，方法为：

（1）确定数据依赖。用数据依赖分析和表示数据项之间的联系，写出每个数据项之间的数据依赖。

（2）对于各关系模式之间的数据依赖进行极小处理，消除冗余的联系。

（3）按照数据依赖的理论对关系模式逐一进行分析，考察是否存在部分函数依赖、传递函数依赖、多值依赖等，确定关系模式分别属于第几范式。

（4）按照需求分析阶段得到的处理要求，分析这些模式对于这样的应用环境是否合适，确定是否要对某些模式进行合并或分解。

需要注意的是，并不是规范化程度越高的关系就越优。例如，当查询经常涉及到两个或多个关系模式的属性时，系统经常进行连接运算。连接运算的代价是相当高的，可以说关系模型低效的主要原因是连接运算引起的。这时可以考虑将几个关系合并为一个关系。因此在这种情况下，第二范式甚至第一范式也许是合适的。

（5）对关系模式进行必要的分解，提高数据操作的效率和存储空间的利用率。

规范化理论为数据库设计人员判断关系模式的优劣提供了理论标准，可用来与此模式可能出现的问题，使数据库设计工作有了严格的理论基础。

3.2.5.2 北化科研系统数据库的逻辑结构设计

根据数据库概念结构设计阶段得到的北化科研系统数据库 E-R 模型，我们可以得到以下的逻辑结构模型（下划线的为关系的 Key）：

（1）基础数据（部分）

- 学院（院所编号，院所名称，科研负责人，成立时间，联系电话，网址，电子邮箱，地址，邮编，传真）
- 机构（机构编号，机构名称，批准部门，所属学院，成立时间，机构负责人，机构类型，学科分类，组成形式，联系电话，网址，电子邮箱，服务的国民经济行业，从业人数，实验室地址，实验室面积，固定资产，主要

研究方向，机构评估情况）

- 人（人员编码，姓名，职工类别，性别，出生日期，民族，党派，所在院所，所在机构，所在重点研究基地，技术职务，职称，学历，最后学位，联系电话，电子邮箱，学科分类，外语语种，主要学习及工作经历，专业特长，主要研究开发领域及成果，备注）
- 职称（职称编号，职称名称，职称代码）

（2）合同管理

- 合同（合同号，合同名称，项目类型，项目状态，项目来源，项目分类，合同金额，实到金额，合同起止时间，是否归档，最终结题方式，结题时间，经费预算情况，参加人员，研究内容记进度，项目完成情况评价）

（3）经费管理（部分）

- 到款（到款编号，到款日期，来款单位，到款日期，来款金额，备注）
- 结算（结算单编号，结算日期，合同号，项目名称，院系单位，经费负责人，结算办法，经费卡号，来款单位，结算金额，备注）
- 转账（转账单编号，转账日期，所属院系，项目负责人，经费卡号，转账金额，合同号，转入单位，转入银行，转入账号，备注）

论文管理模块和成果管理模块的关系模型类似于以上所列出的几个模块，就不逐一列举了。

3.2.6 北化科研系统数据库的物理结构设计

数据库在物理设备上的存储结构与存储方法称为数据库的物理结构，它依赖于给定的计算机系统。为一个给定的逻辑数据模型选取一个最适合应用要求的物理结果的过程，就是数据库的物理设计。

1、数据库的物理结构设计的步骤：

（1）确定数据库的物理结构，在关系数据库中主要指存取方法和存储结构；

（2）对物理结构进行评价，评价的重点是时间和空间效率。

2、北化科研系统数据库的物理结构设计

（1）数据库中的表

进行数据库物理设计的第一步就是要创建一个数据库。北化科研系统的数据库采用的是在 Oracle 数据库系统，我们首先在 Oracle 数据库中创建了 buct 数据库。为了便于编程，本系统中所有表、列名均采用英文或缩写，例如，院系名称的代码为 INSTITUTE_NAME，人（PERSON 缩写为 PSN）的姓名代码为 PSN_NAME。

数据库中表的名称和标识符如表 3-2 所示：

表 3-2　数据库中的表（部分）

Table.3-2　the tables in database system(not all)

序号	表名称	字符标识
1	用户信息表	users
2	院系信息表	institute
3	机构组织信息表	organization
4	重点研究基地信息表	key_tech_center
5	人员信息表	person
6	职称表	title
7	学历表	degree
8	结算办法表	outlet_method
9	合同信息表	project
10	到款信息表	outlet
11	结算信息表	outlet_js
12	转账信息表	outlet_zz
13	退费信息表	outlet_tf
14	著作论文信息表	article
15	鉴定信息表	checkup
16	奖励信息表	award

2、表的结构设计

（1）基础数据模块（部分）：

- 学历表结构如下（图 3-10）：

	Name	Type	Nullable	Default	Comments
▶	PK_DEGREE	VARCHAR2(256)	☐		键值
	DEGREE_NO	VARCHAR2(256)	☑		学历编号
	DEGREE_NAME	VARCHAR2(256)	☑		学历名称
	DEGREE_CODE	VARCHAR2(64)	☑		学历代码

图 3-10　学历表结构

Fig.3-10　the structure of degree table

- 院系信息表结构如下（图 3-11）：

Name	Type	Nullable	Default	Comments
INSTITUTE_NAME	VARCHAR2(64)	☑		机构名称
INSTITUTE_MAIL_ADDRESS	VARCHAR2(64)	☑		机构邮编
INSTITUTE_EMAIL_ADDRESS	VARCHAR2(64)	☑		机构电子邮件
INSTITUTE_DIRECTOR_NAME	VARCHAR2(64)	☑		科研负责人姓名
INSTITUTE_OFFICE_PHONE_NUMBER	VARCHAR2(64)	☑		机构电话
INSTITUTE_FAX_NUMBER	VARCHAR2(64)	☑		机构传真
INSTITUTE_WEBSITE	VARCHAR2(64)	☑		机构网址
INSTITUTE_BRIEF_INFO	VARCHAR2(64)	☑		
PK_INSTITUTE	VARCHAR2(64)	☐		
INSTITUTE_NO	VARCHAR2(64)	☑		
INSTITUTE_FOUND_TIME	DATE	☑		
INSTITUTE_NICK_NAME	VARCHAR2(64)	☑		
INSTITUTE_IS_DELETED	NUMBER	☑	0	是否被删除，0未删除，1删除
INSTITUTE_DELETED_TIME	DATE	☑		删除日期
INSTITUTE_SECRETARY	VARCHAR2(256)	☑		科研秘书

图 3-11　院系信息表结构
Fig.3-11　the structure of institution table

- 机构信息表结构如下（图 3-12）：

Name	Type	Nullable	Default	Comments
ORG_NAME	VARCHAR2(64)	☑		机构名称
ORG_NO	VARCHAR2(64)	☑		机构编号
ORG_BELONG_TO	VARCHAR2(64)	☑		所属院所
ORG_SUBJECT_TYPE	VARCHAR2(64)	☑		学科分类
ORG_FONDATION_TIME	DATE	☑		成立时间
ORG_DIRECTOR_NAME	VARCHAR2(64)	☑		机构负责人
ORG_OFFICE_PHONE_NUMBER	VARCHAR2(64)	☑		联系电话
ORG_EMAIL_ADDRESS	VARCHAR2(64)	☑		电子邮件
ORG_WEBSITE	VARCHAR2(64)	☑		网址
ORG_NATION_SERVICE	VARCHAR2(64)	☑		服务的国民经济行业
ORG_PERSON_COUNT_PHD	VARCHAR2(64)	☑		博士生
ORG_PERSON_COUNT_MASTER	VARCHAR2(64)	☑		硕士生
ORG_PERSON_COUNT_BECHLOR	VARCHAR2(64)	☑		本科生
ORG_PERSON_COUNT_SENIOR	VARCHAR2(64)	☑		
ORG_PERSON_COUNT_VICE_SENIOR	VARCHAR2(64)	☑		
ORG_PERSON_COUNT_MID	VARCHAR2(64)	☑		
ORG_LAB_LOCATION	VARCHAR2(64)	☑		实验室地址
ORG_LAB_AREA_SIZE	VARCHAR2(64)	☑		实验室面积
ORG_FIX_ASSET	VARCHAR2(64)	☑		固定资产
PK_ORG	VARCHAR2(64)	☐		

图 3-12　机构信息表结构
Fig.3-12　the structure of organization table

（2） 合同管理模块

- 合同信息表结构如下（图 3-13）：

Name	Type	Nullable	Default	Comments
PROJECT_NAME	VARCHAR2(64)	☑		项目名称
PROJECT_SOURCE	VARCHAR2(64)	☑		项目来源
PROJECT_CATEGORY	VARCHAR2(64)	☑		项目类别
PROJECT_DIRECTOR	VARCHAR2(64)	☑		项目第一负责人
PROJECT_VICE_DIRECTOR	VARCHAR2(64)	☑		项目第2负责人
PROJECT_THIRD_DIRECTOR	VARCHAR2(64)	☑		项目第3负责人
PROJECT_OTHER_DIRECTORS	VARCHAR2(64)	☑		项目其他负责人
PROJECT_SECRET_DEGREE	VARCHAR2(64)	☑		项目密级
PROJECT_SUBJECT_TYPE	VARCHAR2(64)	☑		学科分类
PROJECT_EXPLORE_TYPE	VARCHAR2(64)	☑		研究类别,关联研究
PROJECT_NATION_SERVICE	VARCHAR2(64)	☑		服务的国民经济行
PROJECT_PERMITED_TIME	VARCHAR2(64)	☑		申报批准时间
PROJECT_START_TIME	DATE	☑		项目起始时间
PROJECT_END_TIME	DATE	☑		项目终止时间
PROJECT_STATUS	VARCHAR2(64)	☑		项目状态
PROJECT_IS_GOING_IDENTIFY	VARCHAR2(64)	☑		是否验收
PROJECT_IS_ARCHIVE	VARCHAR2(64)	☑	'否'	是否归档
PROJECT_CORPORATE_TYPE	VARCHAR2(64)	☑	'dep'	合作形式
PK_PROJECT_PSN_DETAIL	VARCHAR2(64)	☑		项目名称
PROJECT_NO	VARCHAR2(64)	☑		项目号
PK_PROJECT	VARCHAR2(64)			主键
PROJECT_ATTRIBUTE	VARCHAR2(64)	☑		项目性质（正式合
PROJECT_TYPE	VARCHAR2(64)	☑		项目分类
PROJECT_DELEGATE_UNIT	VARCHAR2(64)	☑		

图 3-13 合同信息表结构

Fig.3-13 the structure of project table

（3）经费模块（部分）

- 到款信息表结构如下（图 3-14）：

Name	Type	Nullable	Default	Comments
PK_OUTLET	VARCHAR2(1024)			
OUTLET_NO	VARCHAR2(1024)	☑		到款单编号
OUTLET_DATE	DATE	☑		到款日期
OUTLET_FROM_UNIT	VARCHAR2(1024)	☑		来款单位
OUTLET_FEE	NUMBER	☑		到款金额
OUTLET_TYPE	VARCHAR2(1024)	☑		经费类型（纵、横）
OUTLET_INFO	VARCHAR2(1024)	☑		到款备注
OUTLET_STATUS	VARCHAR2(1024)	☑		经费状态（到款是否经过处理）
OUTLET_STATUS_NO	VARCHAR2(256)	☑		入账或结算编号
OUTLET_IS_DELETED	NUMBER	☑	0	是否被删除（0未删除，1删除）
OUTLET_DELETED_TIME	DATE	☑		删除时间

图 3-14 到款信息表结构

Fig.3-14 the structure of come-outlay table

- 结算信息表结构如下（图 3-15）：

Name	Type	Nullable	Default	Comments
PK_OUTLET_JS	VARCHAR2(1024)			
PK_OUTLET	VARCHAR2(1024)	☑		
OUTLET_JS_FEE	NUMBER	☑		结算金额
OUTLET_JS_NO	VARCHAR2(1024)	☑		结算单编号
OUTLET_JS_DATE	DATE	☑		结算日期
PROJECT_NO	VARCHAR2(1024)	☑		合同号
PROJECT_NAME	VARCHAR2(1024)	☑		项目名称
PROJECT_DIRECTOR	VARCHAR2(1024)	☑		项目负责人
PSN_IN_INSTITUE	VARCHAR2(1024)	☑		负责人所在院系
PK_PROJECT	VARCHAR2(1024)	☑		
PROJECT_FEE	NUMBER	☑		合同金额
OUTLET_FROM_UNIT	VARCHAR2(1024)	☑		来款单位
OUTLET_CARD_NO	VARCHAR2(1024)	☑		负责人经费卡号
OUTLET_JS_INFO	VARCHAR2(1024)	☑		结算备注
PK_OUTLET_METHOD	VARCHAR2(1024)	☑		
OUTLET_METHOD_NAME	VARCHAR2(1024)	☑		结算方法

图 3-15　结算信息表结构

Fig.3-15　the structure of balance table

- 转账表结构如下（图 3-16）：

Name	Type	Nullable	Default	Comments
PK_OUTLET_ZZ	VARCHAR2(1024)			
OUTLET_ZZ_NO	VARCHAR2(1024)	☑		转帐单编号
OUTLET_ZZ_DATE	DATE	☑		转帐日期
OUTLET_ZZ_BANK	VARCHAR2(1024)	☑		转入银行
OUTLET_ZZ_BANK_NO	VARCHAR2(1024)	☑		银行账户
PK_OUTLET	VARCHAR2(1024)	☑		
OUTLET_ZZ_FEE	NUMBER	☑		转帐金额
OUTLET_FROM_UNIT	VARCHAR2(1024)	☑		来款单位
PROJECT_NO	VARCHAR2(1024)	☑		合同号
OUTLET_CARD_NO	VARCHAR2(1024)	☑		经费卡号（
OUTLET_ZZ_INFO	VARCHAR2(1024)	☑		转帐备注
OUTLET_ZZ_UNIT	VARCHAR2(1024)	☑		转入单位
PK_OUTLET_IN	VARCHAR2(1024)	☑		
PSN_IN_INSTITUE	VARCHAR2(1024)	☑		负责人所在院
PROJECT_DIRECTOR	VARCHAR2(1024)	☑		项目负责人

图 3-16　转账信息表结构

Fig.3-16　the structure of out-outlay table

（4）论文管理（部分）

- 著作论文信息表结构如下（图 3-17）：

Name	Type	Nullable	Default	Comments
ARTICLE_NAME	VARCHAR2(1024)	☑		著作名称
ARTICLE_ISBN	VARCHAR2(256)	☑		发行代码
ARTICLE_TYPE	VARCHAR2(256)	☑		著作类别
PUBLISH_UNIT	VARCHAR2(1024)	☑		出版单位
PUBLISH_DATE	DATE	☑		出版日期
WORD_COUNT	VARCHAR2(246)	☑		撰写字数
ARTICLE_INFO	VARCHAR2(1024)	☑		
PK_ARTICLE	VARCHAR2(1024)			
SPECIAL_TEXTBOOK	VARCHAR2(1024)	☑		
TEXTBOOK_LEVEL	VARCHAR2(1024)	☑		
PSN_INSTITUTE	VARCHAR2(1024)	☑		所在院系
ORGANIZATION_NAME	VARCHAR2(1024)	☑		所属机构
KTC_NAME	VARCHAR2(1024)	☑		所在重点研究基地
PUBLISH_SCOPE	VARCHAR2(64)	☑		出版地（国内、国外）
SUBJECT_TYPE	VARCHAR2(256)	☑		学科分类（基础数据）
ARTICLE_WORD_COUNT	VARCHAR2(256)	☑		著作总字数
CHN_KEY	VARCHAR2(1024)	☑		中文关键字
ENG_KEY	VARCHAR2(1024)	☑		英文关键字
ARTICLE_ABSTRACT	VARCHAR2(1024)	☑		著作提要

图 3-17　著作论文信息表结构

Fig.3-17　the structure of article table

（5）成果管理（部分）

- 鉴定信息表结构如下（图 3-18）：

Name	Type	Nullable	Default	Comments
CHECKUP_DATE	DATE	✓		鉴定时间
PROJECT_NO	VARCHAR2(1024)	✓		合同号
PROJECT_DIRECTOR	VARCHAR2(1024)	✓		项目负责人
PROJECT_NAME	VARCHAR2(1024)	✓		项目名称
PSN_INSTITUTE	VARCHAR2(1024)	✓		所在院所
ORGANIZATION_NAME	VARCHAR2(1024)	✓		所在机构
KTC_NAME	VARCHAR2(1024)	✓		重点机构
CORPORATE_UNIT	VARCHAR2(1024)	✓		协作单位
START_DATE	DATE	✓		项目起始时间
END_DATE	DATE	✓		项目终止时间
PROJECT_SOURCE	VARCHAR2(1024)	✓		项目来源
PROJECT_CLASS	VARCHAR2(1024)	✓		项目分类
CHECKUP_ADDRESS	VARCHAR2(1024)	✓		鉴定地址
ORGANIZATION_UNIT	VARCHAR2(1024)	✓		组织单位
CHECKUP_METHOD	VARCHAR2(1024)	✓		鉴定方式
CHECKUP_LEVEL	VARCHAR2(1024)	✓		成果水平
CERTIFICATE_NO	VARCHAR2(1024)	✓		成果证号
FIRST_UNIT	VARCHAR2(1024)	✓		第一完成单位
OUR_POSITION	VARCHAR2(1024)	✓		
UNIT_NUMBER	VARCHAR2(1024)	✓		参加单位总数
CHECKUP_INFO	VARCHAR2(1024)	✓		备注
PK_CHECKUP	VARCHAR2(1024)			

图 3-18　鉴定信息表结构

Fig.3-18　the structure of checkup table

3、关系模式存取方法的选择

北化科研系统的数据库系统是多用户共享的系统，对同一个关系要建立多条存取路径才能满足多用户的多种应用要求。数据库物理设计的任务[21]之一就是要确定选择哪些存取方法，即建立哪些存取路径。

存取方法是快速存取数据库中数据的技术。数据库管理系统一般都提供多种存取方法。常用的有三类：第一类是索引方法，目前主要是 B+树索引方法；第二类是聚簇（Cluster）方法；第三类是 HASH 方法。

在北化科研系统数据库物理设计中我们采用的是索引存取方法。在科研管理信息系统中的合同和经费两个模块，是日常用的较多的两个模块，其中的表包括合同（project）、到款（outlet）、入账（outlet_in）、结算（outlet_js）、转账（outlet_zz）、退费（outlet_tf）等几个主要的表。我们对这几个表的主要属性建立了索引，通过这种方法可以提高查询等的效率。

4、确定数据库的存储结构

确定数据库物理结构主要是指确定数据的存放位置和存储结构，包括确定关系、索引聚日志、备份等的存储安排和存储结构；确定系统配置等。

确定数据存放位置和存储结构要综合考虑存取时间、存储空间利用率和维护代

价三个方面的因素。

（1）确定数据的存放位置

为了提高系统性能，通常根据应用情况将数据的易变部分与稳定部分、经常存取不符合存取频率较低部分分开存放。

根据实际应用情况，在北化科研系统的数据库中我们将表和建立的索引分开存放，这样可以达到提高物理 I/O 读写的效率的目的。由于北化科研系统的数据量不是很大，因此我们数据存放在一个磁盘上，但是从安全角度出发，我们对数据定期做备份处理。由于数据库的数据备份和日志文件备份等只在故障恢复适才使用，而且数据量很大，所以我们把它存放在与其他数据库对象（表、索引等）不同的磁盘上。

（2）确定系统配置

北化科研系统的数据库系统配置中，我们主要考虑的变量包括：同时使用数据库用户数，同时打开的数据库对象数，内存分配参数，缓冲区分配参数（使用缓冲区长度、个数），物理块大小等。

3.2.7 北化科研系统数据库的实施

目前，北化科研系统已经试运行，数据库也在运行中。我们已将以前的旧的科研管理系统中的数据装入到系统数据库中，数据库现在运行良好，处于维护、优化阶段。

3.3 小结

本章主要讨论了北化科研系统数据库的设计与实现。介绍了数据库设计的方法和基本步骤，探讨了数据库设计各个阶段的目标、方法、应注意的事项。并对北化科研系统数据库设计的需求分析和概念结构设计、逻辑结构设计以及物理结构设计阶段做了重点介绍。通过对北化科研系统数据库设计的总结，使我们对数据库设计的方法和步骤有了更加深刻的理解，为我们进一步的深入研究奠定了坚实的基础。

第四章　科研管理信息系统数据库的安全

北化科研系统现在已经在实际的运行当中，系统的安全性是关系到系统是否能够稳定运行的不可忽视的重要问题。而其数据库作为系统的关键组成部分，它的安全性也必然是系统能够稳定运行的关键因素。由于数据库系统安全与信息安全紧密相关，在本章中，我们首先对信息安全作了的简单介绍，然后对数据库的安全问题作了论述，最后对北化科研系统数据库的安全措施进行了论述。

4.1 信息安全概述

信息技术的普及和信息高速公路的建设热潮为信息资源的开发和共享创造了良好的基础。由于信息资源的开发，人们可以借助信息技术和远程通信技术，随时随地获取各种所需的信息，提高决策的准确性和客观性，以便在变化和竞争中掌握主动权并争取优势。然而，信息技术也和其它科学技术一样是一把双刃剑，当大部分人使用信息技术提高工作效率、为社会创造更多财富的时候，却也有一些人用它做着相反的事情。因此，信息安全问题随着信息技术的发展和信息化进程而变得日益重要和敏感。

4.1.1 信息安全的重要性

1、从技术角度看信息安全

随着数字化信息社会的发展，无论是企业管理、日常办公或是商务往来，甚至普通人之间的交往等的各个方面都需要信息安全技术的保障。研究数字化信息安全，必须从分析数字化信息社会所依靠的基本技术着手[28]。在网络时代，通过个人计算机在与他人的通讯过程中，安全问题便暴露了出来，且日益严峻。此外如今使用最广泛的网络协议是 TCP/IP 协议，它的主要设计目标是互联与互通，而不是安全。该协议中已有许多人所共知的安全漏洞和隐患。另一方面，由于软件设计方法本身的发展水平所限，设计人员无法在软件设计过程中考虑到计算机安全的所有方面，从而出现了一些重要的软件公司频频发布安全隐患的补丁以应急堵漏洞的现象。这些技术缺陷使得各类计算机犯罪及“黑客”攻击事件日益频繁，从而严重威胁着信息系统的安全。

因此，从信息技术角度来讲，有关专家认为，在信息技术中心片是细胞，计算

机是大脑，网络是神经，智能是营养，信息是血液，信息安全是免疫系统[29]。可见信息安全的重要性。

2、从社会角度看信息安全

信息是社会发展的重要战略资源。信息技术和信息产业正在改变传统的生产、经营和生活方式，成为新的经济增长点，信息使知识经济初见端倪。信息网络国际化、社会化、开放化、个人化的特点使国家的“信息边疆”不断延伸，甚至到了每一个上网者个人。国际上围绕信息的获取、使用和控制的斗争愈演愈烈，信息安全成为维护国家安全和社会稳定的一个焦点，全国都给以极大的关注与投入。随着全球信息化的飞速发展，我国大量建设和发展的各种信息化系统已经成为国家关键基础设施，其中许多业务要与国际接轨。诸如电信、电子商务、金融网络等。网络信息安全已成为急待解决、影响国家大局和长远利益的重大关键问题。信息安全是国家安全、国防安全的不可忽略的必要前提，同时也是21世纪与每个社会成员密切相关的重要问题之一。

4.1.2 信息安全问题的新局面

现代信息系统的飞速发展是以计算机译及计算机网络系统的飞速发展为标志的。信息化的社会在尽力追逐信息系统的时空性能，高速计算机、高速网络逐步民用化、商用化、家用化。正因为信息在人类社会活动、经济活动中起着越来越重要的作用，信息的安全就日益成为关系成败的关键要点，日益引起人们越来越深刻的重视[30]。

新世纪之初，社会信息化的进程明显地加速，计算机、网络、大规模的信息系统，使得社会在高度信息化的基础上高度自动化，同时也就导致对信息系统的高度依赖，依赖程度越高，信息系统和安全问题的严重性就越大，与早期的信息系统相比，现代信息系统面临着更严重的挑战。

1、信息存放的分散化

PC 机和工作站的大量使用造成了信息分散的局面。PC 机和工作站都有存储空间，大量信息存放在PC机和工作站上，给信息安全带来极大的隐患。

2、信息处理的网络化

大量的内部局域网系统与Internet联网形成了信息处理网络化的不可逆转的趋势。随着单位局域网与Internet联网，内部系统暴露在外部世界面前，虽然有许多系统安全和网络安全的技术可以使用，但用任何安全技术都是用程序实现的，程序的错误和漏洞难以避免，这就决定了任何安全技术都存在错误和漏洞。再加上

安全管理出现一些漏洞，与Internet联网无法绝对避免给信息安全带来的潜在风险。

3、应用模式的影响

客户机/服务器、Internet、Intranet等应用模式的兴起加速了信息存放和处理的分布化。客户机/服务器、Internet、Intranet等应用模式由于其灵活、效率高和成本低而大受欢迎。现在越来越多的应用采用客户机/服务器、Internet、Intranet等应用模式。但是，许多这样的应用虽然在功能上令人满意，而在安全性上却问题很多。

造成系统不安全的因素有很多，既有系统的稳定性或可靠性不定，环境干扰或自然灾害等客观因素，也有人员工作失误、操作不当等，但对系统的安全影响最大的是人为的攻击破坏，造成巨大的损失。

4.2 数据库的安全问题

对信息安全有所了解之后，就不难理解数据库安全问题了。作为信息安全的一个重要研究领域，数据库安全[31]有着非常重要的意义。

4.2.1 数据库安全问题的定义

要如何理解数据库安全问题呢？我们从数据库系统的目标入手来探讨。

1、数据库系统的目标

从狭义上而言，数据库系统一般可以理解成两部分：一部分是数据库，按一定的方式存取数据；另一部分是数据库管理系统（DBMS），为用户及应用程序提供数据访问，并具有对数据库进行管理、维护等多种功能。数据库管理系统（DBMS）为应用和用户集中统一管理数据，最初的动机和追求的目标大致可以归纳为以下几点：

（1） 提供数据共享，集中统一管理数据；

（2） 减少数据冗余：数据库管理系统集中统一管理数据，使得不同的应用可以共享信息和数据，同时达到减少数据冗余的目的；

（3） 简化应用程序对数据的访问：通过数据库管理系统的支持，应用程序得以在更深的逻辑层次上访问数据，从而简化应用程序对数据的访问；

（4） 解决数据一致性的问题：这可能是数据库管理系统最重要的作用，因为不同的应用程序在共同的数据环境中运行，要么是各行其是，同一逻辑数据难以保持一致，要么就是难以确认是否获得了可靠的共享；

（5） 保证数据独立性问题：数据及数据结构是客观世界的抽象，应用则是千变万化的人类活动的要求的体现，这些变化要求程序和数据的相互独立，否则应用程序很快就会变得不可收拾。数据库管理系统的主要功能之一是降低对数据及数据结构的依赖。

以上所列出的好处仅仅是数据库发展过程中几个最重要的动机，发展变化的应用对数据库提出的要求是无穷无尽的。

2、数据库系统的安全性

为了达到上述的目的，数据的集中存放和管理永远是必要的。即使是在面向对象数据库中，和方法封装在一起的数据仍然是要有专门的管理系统集中存放和管理的。数据在计算机系统中的集中存放和管理的主要问题，除了功能和性能方面的技术问题，最重要的问题就是数据库数据的安全问题。在计算机中存储的数据从业务系统的角度看都是信息，对于人类活动，信息的重要性是不言而喻的。如何既提供充分的服务又保证关键的信息不会被泄露出去而损害信息所有者的利益，是信息安全系统的任务，也是数据库管理系统的主要任务之一。

数据库管理员[32]（DBA）的一个重要责任是保证数据库的可靠性，程序和过程在遇到机器故障、程序错误和人为错误时能够提供数据的可靠性。DBA 为保证数据的安全可靠性所做的主要工作大致如下[33]：

- 创建用户和用户组，并进行权限管理
- 制定备份或转储数据库
- 定期备份或转储数据库
- 发生故障时恢复数据库

对数据库的攻击，主要有直接攻击和间接攻击，直接攻击比较好防备，而间接攻击相对来说就困难多了，一个数据库安全性的好坏主要体现在对间接攻击的防范策略上。可以采用有限制的响应封锁，以及响应追踪等多种方法防范间接攻击，有时为了某些数据的安全不得不拒绝服务或提供一些不准确的信息。

数据库安全问题主要集中在数据共享问题上。除了一般意义上的对数据资源的存取控制问题外，数据库管理系统本身的可信度更是重要的问题，同时数据库本身的可信度也直接关系到数据库是否安全可靠。

3、数据库系统安全的含义

数据库系统安全，包含系统运行安全和系统信息安全两层含义[34]：

第一层是指系统运行安全，它包括：

- 法律、政策的保护，如用户是否有合法权利，政策是否允许等。
- 物理控制安全，如机房加锁等。
- 硬件运行安全。
- 操作系统安全，如数据文件是否保护等。

- 灾害、故障恢复。
- 死锁的避免和解除。
- 电磁信息泄漏防止。

第二层是指系统信息安全，它包括：

- 用户身份标识。
- 用户身份鉴别。
- 用户存取权限控制。
- 数据存取权限、方式控制。
- 审计跟踪。
- 数据加密。

4.2.2 数据库系统所受到的威胁和主要安全特点

1、数据库系统的安全威胁

原则上，凡是造成对数据库内存储数据（包括敏感、非敏感的信息）的非授权访问——读取，或非授权的写入——增加、删除、修改等，都对数据库的数据安全造成了威胁或破坏。另一方面，凡是正常业务需要访问数据库时，令授权用户不能正常得到数据库的数据服务时，也称之为对数据库的安全形成了威胁或破坏。因为很显然，这两种情况都会对数据库的合法用户的权益造成侵犯，或者是信息被窃取，或者是由于信息的破坏而形成提供错误信息的服务，或者是干脆拒绝提供服务。

根据安全威胁的来源及攻击的性质，可将数据库的安全威胁大致分为以下几类[35]：

（1）逻辑的威胁

- 非授权访问：即用户对未获得访问许可的信息的访问。
- 推理访问数据：是指由授权读取的数据，通过推论得到不应访问的数据。
- 病毒：病毒可以自我复制，永久地或是通常是不可恢复地破坏自我复制的现场，达到破坏信息系统、取得信息的目的。
- 特洛伊木马：一些隐藏在公开的程序内部，收集环境的信息，可能是由授权用户安装的，利用用户的合法的权限对数据安全进行攻击。
- 天窗、隐通道：藏在合法程序内部的一段程序代码。特定的条件下（例如特殊的一段输入数据）将启动这段程序代码，从而许可此时的攻击可以跳过系统设置的安全稽核机制进入系统，以实现对数据防范的攻击和窃取数

据的目的。

（2）硬盘的威胁

- 磁盘故障：在计算机运行过程中最常见的问题就是磁盘故障，它会导致重要数据的丢失。
- I/O 控制器故障：控制器发生故障，会破坏数据的完整性。
- 电源故障：电源故障分为电源输入故障和系统内部电源故障。由于系统停电是不可预料的，因而不论处在哪种情况下都有可能是数据受到毁损。
- 存储器故障。
- 介质、设备和其它备份故障：数据存储在可移动介质上以作备份，恢复工作则是包括数据的拷贝。如果服务器出错、被毁，则存储设备或其使用的介质的任何错误都会导致数据的丢失。
- 芯片和主板的故障：芯片和主板的故障会导致严重的数据毁损。

（3）人为错误的威胁

操作人员或系统的直接用户的错误输入、应用程序的不正确使用，都可能导致系统内部的安全机制的失效，导致非法访问数据的可能，也可能导致系统拒绝提供数据服务。

（4）传输的威胁

目前，数据库大多是基于网络环境的。在网络系统中，无论是调用任何指令，还是任何信息的反馈均是通过网络传输[36]实现的，因此对数据库而言，就存在着网络信息传输的威胁。

- 对网络上信息的监听：对于网络上传输的信息，攻击者只需要在网络链路上通过物理或逻辑的手段，就能对数据进行非法的截获与监听，进而得到敏感信息。
- 用户身份的仿冒：对用户身份仿冒这一常见的网络攻击方式，能对数据库的信息产生严重的威胁。
- 对网络上信息的篡改：攻击者可能对网络上的信息进行截获并且篡改其内容（增加、删除或改写），使用户无法获得准确、有用的信息或落入攻击者的陷阱。
- 对信息进行重发："信息重发"的攻击方式，即攻击者截获网络上的密文信息后，并不将其破译，而是再次转发这些数据包，以实现其恶意目的。

（5）物理环境的威胁

- 自然的或意外的事故：地震、火灾、水灾等导致硬件的破坏，进而导致数据的丢失和损坏。
- 偷窃：这里的偷窃是指偷窃信息和服务。
- 蓄意破坏：主要是指恐怖组织的活动。

2、数据库系统的主要安全特点

从计算机系统的安全结构来考虑的话，数据库系统在整个计算机系统的逻辑位置，在安全周界的附近，应当受到安全操作系统的强有力的支持和保护。然而，面对上述的威胁，数据库系统有自身的特点，所需的安全功能，是操作系统不能完全提供和保障的[37]。

（1） 数据库中信息众多，需要保护的客体不多。其中的各种信息所要求的安全程度也不相同。

（2） 数据库中某些重要数据的生存周期较长，需要长期保护。

（3） 在开放的网络环境中，用户众多而分散，不同的用户所能享用的资源的权限很可能是不同的，导致安全性问题更加复杂。

（4） 操作系统中受保护的客体是实际资源，而在数据库系统中受保护的客体可能是复杂的逻辑结构。若干复杂的逻辑结构可能映射到同一物理数据客体上。

（5） 需要防范由非敏感数据推理出敏感数据的推理攻击。

（6） 高可用性和频繁的访问，也使得安全性问题更为突出。

4.2.3 数据库系统的安全要求及对策

1、数据库的安全要求

面对诸多的威胁，加上数据库系统自身的安全特点，对数据库的安全性就提出了诸多要求。计算机安全性的三个方面：完整性、保密性和可用（获）性，与数据库管理系统都有关系[38]。完整性既适用于数据库的个别元素也适用于整个数据库，所以在数据库管理系统的设计中完整性是主要的关心对象。保密性由于推理攻击而变成数据库的一大问题。最后，因为共享访问的需要是开发数据库的基础，所以可用性是重要的。但是可用性与保密性是相互冲突的，因此需要互相平衡。

数据库的安全性要求从以下六个方面来研究（见表 4-1）[39]。其中，物理上的数据库完整性、逻辑上的数据库完整性、元素的完整性和可审计性属于完整性方面的要求；访问控制和用户认证是保密性方面的要求；可获性也就是可用性。

表 4-1 数据库安全要求表
Table.4-1 the requirer of database's security

安全要求	注释
物理上的 数据库完整性	预防数据库数据物理方面的问题。 如掉电以及当被灾祸破坏后能重构数据库
逻辑上的 数据库完整性	保持数据的结构。比如： 一个字段的值的修改不至于影响其他字段
元素的完整性	包含在每个元素中的数据是准确的
可审计性	能够追踪到谁访问、修改过数据库的元素
访问控制	允许用户只访问被批准的数据， 以及限制不同用户有不同的方位模式，如读或写
用户认证	确保每个用户被正确的识别， 既便于审计追踪也为了限制对特定的数据进行访问
可获（用）性	用户一般可以访问数据库 以及所有被批准访问的数据

（1）数据库的完整性

数据库管理程序必须确保只有经批准的用户才有权更新，这意味着必须有访问控制，另外数据库系统还必须有防范非人为的外力灾难。

数据库的完整性是 DBMS 操作系统和计算系统管理者的责任。从操作系统和计算系统管理者的观点来看，数据库和 DBMS 分别是文件和程序。因此整个数据库的一种形式的保护是对系统上的所有文件周期性地做备份。数据库的周期性的备份可以控制由灾祸造成的损失。

为了维护在系统出错后能重建数据库，DBMS 必须维护对事务的记录。在出现系统失败的事故时，由数据库的备份开始重新处理记录之后的所有业务。

（2）元素的完整性

数据库元素的完整性是指它们的正确性和准确性。由于用户在搜索数据、计算结果和输入数值时可能会出现错误，所以 DBMS 必须帮助用户在输入时能发现错误，并在插入错误数据后能纠正它们。

DBMS 用两种方式维护数据库中每个项目的完整性：

第一种方式：字段检查在一个位置上的适当的值，这种检查防止输入数据库时可能出现的简单错误。通过访问控制来维护数据库的完整性和一致性。一个数据库可能包含几个来源的数据。而在开发一个数据库之前，可能在许多表中存储了重复的数据。需要一种策略来解决可能发生的数据冲突问题。

第二种方式：维护数据库的更改日志。更改日志是数据库每次改变的记录文

件，日志包括原来的值和修改后的值。数据库管理员可以根据日志撤销任何错误的修改。

（3）可审计性

在某些应用中，可能需要产生对数据库的所有访问（读或写）的审计记录。这种记录可以协助维持数据的完整性，或者至少可以帮助在事后发现谁以及何时影响过什么值。攻击者可能会以逐次递增的方式形成对被保护数据的访问，不是单用一次访问来揭露被保护的数据，而是用一组访问来揭示一些敏感的数据。在这种情况下，审计踪迹有利于辨别出已经给攻击者什么线索，可以作为是否还要告诉攻击者更多的东西的依据。

审计粒度成为审计中的一个问题。数据库的审计踪迹则必须包括对记录字段和元素一级的访问。对于大多数数据库应用，要求审计信息很详细。

在某些情况下，可能访问了一个记录，但是 DBMS 并没有把这个记录的真实情况告诉给用户，如 DBMS 会告诉用户这个记录所有域的和或者不直接访问某些元素而确定它们的值。因此，所以直接访问过的记录的日志可能夸大也可能低于用户实际知道的值。

（4）访问控制

数据库常常根据用户访问特权逻辑地被分割。如一般用户访问一般数据，对于科研管理系统来说，纵向用户只能访问到纵向信息，而不能访问到横向的信息等。

数据库管理的指定应该允许谁访问那些数据，这些数据可以是字段或记录，或者甚至是元素。DBMS 必须实施这一访问策略，确定对所有指定的数据的允许访问或者禁止访问。DBMS 批准一个用户或者程序可能有权读、改写、删除或附加一个值，可能增加或删除整个字段或记录，或者重新组织完全的数据库。

对数据库的访问控制和操作系统的访问控制有根本区别。事实上数据库中更为复杂。操作系统的目标，如文件相关联的项目，用户可能通过只读其他的文件而确定其相关联的文件的内容。而数据库中的记录字段和元素是相互关联的，用户有可能通过只读其他的数据库元素而确定一个元素，也就是说用户可以通过推理的方法从某些数据的值得到另外一些数据值。

通过推理访问数据可能不需要有对安全目标的直接访问权。限制推理则意味着为防止可能的推理而禁止一些推理路径。通过限制访问来控制推理，也限制了无意访问未经批准的数据的那些用户的查询，而为了检查也可能降低数据库访问的效率。

（5）用户认证

DBMS 要求进行严格的用户认证来保证对数据库的访问者是合法的用户。一个 DBMS 可能要求用户传递指定的密码和时间日期进行验证。这一认证是在操作

系统完成的认证之外另加的。DBMS 在操作系统之外作为一个应用程序被运行，这意味着它没有互操作系统的可信赖路径，因此必须怀疑它所收的任何数据，包括用户认证。因此 DBMS 最好有自己的认证机制。

（6）可用性

可用性是指当需要时能否存取所需的信息。当决定是否允许访问存取时，DBMS 必须考虑数据的可获性、访问的可接受性和用户的认证特性。

2、数据库系统的安全对策

为了保护数据库系统免受上述威胁的影响，达到其基本的安全要求，应当采取合理的安全策略。这些安全策略要能保证数据库中的数据不会被有意的攻击或无意的破坏，能保证不会发生数据的外泄、丢失和毁损。

实际上，安全问题并不是数据库系统所独有的，所有计算机化的系统中都存在这个问题，只是由于数据库系统中存放了大量数据，并为许多用户直接共享，使安全性问题更为突出。在计算机系统中，安全对策一般是分级设置的，图 4-1 表示的就是一种常见的安全模型[40]。

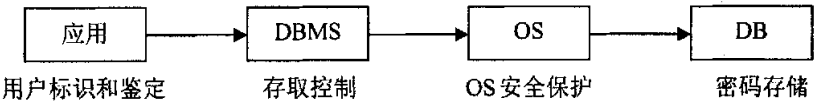


图 4-1 安全模型图

Fig.4-1 the model chart of security

在用户要求进入计算机系统是，系统首先根据用户输入的用户表示进行身份鉴定，只有合法的用户才允许进入计算机系统。对已进入系统的用户，DBMS 还要进行存取控制，只允许用户进行合法操作。操作系统一级也会有自己的保护措施。数据最后还可以加密存储。在这里，仅讨论与数据库系统有关的用户管理、存取控制、数据加密、审计跟踪和攻击测试这些安全对策。

（1）用户管理

用户需要防卫的数据库系统、操作系统、文件系统以及网络系统等在用户管理方面非常相似，采用的方法和措施也十分近似。在一个多用户系统中，进而在网络环境下，识别用户永远是安全控制机制中最重要的一个环节，也是安全防线的第一个环节。

这里的用户管理包括标识和鉴别。标识是指用户向系统出示自己的身份证明，最简单的方法是输入用户名和口令字。而鉴别则是系统验证用户的身份证明。身份认证是安全系统最重要而且最困难的工作。除口令控制外，用户身份认证还可以采用比较复杂的计算过程和函数来完成，而智能卡技术、数字签名技术和生理特征（如：指纹、体温、生育等）认证技术的迅速发展也为更具有高安全要求的用户身份认证提供了实用可行的技术基础。

（2）存取控制

数据库的存取控制机制是定义和控制用户对数据库数据的存取权限，以确保只授权给有资格的用户访问数据库的全新并防止和杜绝对数据库中数据的非授权访问。

一般而言，存取权限是由两个要素组成的：数据对象和操作类型。定义一个用户的存取权限就是要定义这个用户可以在哪些数据对象上进行哪些类型的操作。在数据库系统中，定义存取权限成为授权。在这些授权定义经过编译后，存放在数据字典中。当用户提出存取操作时，DBMS 查找数据字典，根据其存取权限对操作的合法性进行检查。若用户的操作请求超过了定义的权限，系统将拒绝执行此操作。这就是数据库的存取控制。

数据库管理系统 DBMS 中队数据库的存取控制要比操作系统中对文件的存取控制机制复杂得多。因为，DBMS 需要对更为精细的数据粒度加以控制；数据库中的粒度有记录、表格、属性、字段和值等。DBMS 中对数据库的存取控制是建立在操作系统的安全机制的基础之上的。一般来说，就存取控制而言，低安全级别的操作系统之上很难建立高安全等级的数据库系统；而高安全等级的操作系统之上建老的数据库系统也不一定就是高安全等级的[41]。

（3）数据加密

数据库系统，担负着存储和管理关键业务数据和信息的任务。每个信息系统都要保证安全性和保密性，一般而言，数据库系统提供的安全控制的措施能满足一般的数据库应用，但对于一些重要部门或敏感领域的应用，仅有这些是难以完全保证数据的安全性的。因此有必要对数据库中存储的重要数据进行加密处理，以强化数据存储的安全保护。

数据加密是防止数据库中数据泄露的有效手段，与传统的通信或网络加密技术相比，由于数据保存的时间要长得多，对加密强度的要求也更高。而且，由于数据库中数据时多用户共享，对加密和解密的时间要求也更高，同时也要求不会明显降低系统性能。

现在有许多强大的加密技术可保证数据的保密性、完整性、真实性和不可否认性，包括专用密钥加密和公钥加密。

（4）审计跟踪和攻击检测

审计功能在系统运行时，将自动对数据库的所有操作记录在审计日志中，攻击检测系统则是根据审计数据分析检测内部和外部攻击者的攻击企图，再现导致系统现状事件，以分析发现系统安全的弱点，追查有关责任者。

虽然存取控制在经典和现代安全理论中都是实性系统安全策略的最重要的手段，但软件工程技术目前还没有到到形式证明一个系统的安全体系的程度，因此不可能保证任何一个系统不存在安全漏洞，也还没有一种可行的方法可以彻底解

决合法用户在通过身份认证后滥用特权的问题。这样审计跟踪与攻击检测不仅是保证数据库安全的重要措施，也是任何一个安全系统不可缺少的最后一道防线。

4.3 北化科研系统数据库的安全

北化科研系统的数据库采用的是 Oracle 数据库系统，所以系统数据库的安全机制就包含两个方面，一是 Oracle 数据库自身的安全机制，二是科研管理信息系统所采取的安全机制。本节我们就从 Oracle 数据库的安全机制和北化科研系统数据库的安全机制两个方面来进行讨论。

4.3.1 Oracle 数据库系统自身的安全机制

安全的宗旨是保密性、完整性和可用性。具体而言，数据库的安全性是指保护数据库以防止不合法的使用所造成的数据泄露、更改和破坏。

在上述的安全模型中，曾提到在一般的计算机系统中，安全措施是一级一级设置的。在 DB 存储这一级可采用密码技术，当物理存储设备失窃后，它起到保密作用。在数据库系统这一级可提供两种控制：用户标识和鉴定，数据库存取控制。

在 Oracle 多用户数据库系统中，安全机制[42]作下列工作：

- 防止非授权的数据库存取；
- 防止非授权的对模式对象的存取；
- 控制磁盘使用；
- 控制系统资源使用；
- 审计用户动作。

Oracle 利用下列机制管理数据库安全性：

- 数据库用户和模式；
- 特权；
- 角色；
- 存储设置和空间份额；
- 资源限制；
- 审计。

1、Oracle 数据库的存取控制

Oracle 保护信息的方法采用任意存取控制来控制全部用户对命名对象的存取。

用户对对象的存取受特权控制。一种特权是存取命名对象的许可，为一种规定格式。

Oracle 使用多种不同的机制管理数据库安全性，其中有两种机制：模式和用户。模式为模式对象的集合，模式对象如表、视图等。一般情况下，数据库有一组模式。

每个 Oracle 数据库有一组合法的用户，可存取该数据库，可运行该数据库。当建立一数据库用户时，对该用户建立一个相应的模式，模式名与用户名相同。一旦用户连接该数据库，该用户就可以存取相应模式中的全部对象，一个用户仅与同名的模式相联系，所以用户和模式是类似的。

用户的存取权限受用户安全域的设置所控制，在建立一个数据库的新用户或更改一已有用户时，安全管理员对用户安全域有下列决策：

- 是出数据库系统还是由操作系统维护用户授权信息；
- 设置用户的缺省表空间和临时表空间；
- 列出用户可存储的表空间和在表空间中可使用的空间份额；
- 设置用户资源限制的环境文件，该限制规定了用户可用的系统资源的总量；
- 规定用户具有的特权和角色，可存取相应的对象。

每一个用户有一个安全域，它是一组特性，可决定下列内容：

- 用户可用的特权和角色；
- 用户可用的表空间的份额；
- 用户的系统资源限制。

（1）用户鉴别

为了防止非授权的数据库用户的使用，Oracle 提供两种用户验证方法：操作系统验证和相应的 Oracle 数据库验证。

如果操作系统允许，Oracle 可使用操作系统所维护的信息来鉴定用户。由操作系统鉴定用户的优点是：

- 用户可更方便的连接到 Oracle，不需要指定用户名和口令。
- 对用户授权的控制集中在操作系统，Oracle 不需要存储和管理用户口令。然而用户名在数据库中仍然要维护。
- 在数据库中的用户名项和操作系统审计跟踪相对应。

Oracle 数据库方式的用户验证：Oracle 利用存储在数据库中的信息可鉴定试图连接到数据库的用户，这种鉴别方法仅当操作系统不能用于数据库用户的鉴别时才使用。当用户使用 Oracle 数据库时执行用户鉴别。每个用户在建立时有一个口令，用户口令在建立数据库连接时使用，以防止数据库非授权的使用。用户的口令以密码的格式存储在数据库数据字典中，用户可随时修改其口令。

（2）用户的表空间设置和定额

关于表空间的使用有几种设置选择：

- 用户的缺省表空间；
- 用户的临时表空间；
- 数据库表空间的空间使用定额。

（3）用户资源限制和环境文件

用户可用的各种系统资源总量的限制是用户安全域的部分。利用显式地设置资源限制，安全管理员可防止用户无控制地消耗宝贵的系统资源。资源限制是由环境文件管理。一个环境文件是命名的一组赋给用户的资源限制。另外，Oracle为安全管理员在数据库级提供使能或使不能实施环境文件资源限制的选择。

Oracle 可限制几种类型的系统资源的使用，每种资源可在会话级、调用级或两者上控制。在会话级：每一次用户连接到一数据库，建立一会话。每一个会话在执行 SQL 语句的计算机上耗费 CPU 时间和内存量进行限制。对 Oracle 的几种资源限制可在会话级上设置。如果会话级资源限制被超过，则当前语句被中止（回滚），并返回指明会话限制已达到的信息。此时，当前事务中所有之前执行的语句不受影响，此时仅可作 COMMIT、ROLLBACK 或删除对数据库的连接等操作，进行其它操作都将出错。在调用级：在 SQL 语句执行时，处理该语句有好几步，为了防止过多地调用系统，Oracle 在调用级可设置几种资源限制。如果调用级的资源限制被超过，语句处理被停止，该语句被回滚，并返回一错误。然而当前事务的已执行所有语句不受影响，用户会话继续连接。

a）Oracle 资源限制

- 为了防止无控制地使用 CPU 时间，Oracle 可限制每次调用的 CPU 时间和在一次会话期间调用所使用的 CPU 的时间，以 0.01 秒为单位；
- 为了防止过多 I/O，Oracle 可限制每次调用和每次会话逻辑数据块读的数目。

Oracle 在会话级还提供其它几种资源限制：

- 每个用户的并行会话数限制；
- 会话空闲时间的限制，如果一次会话的 Oracle 调用之间时间达到该空闲时间，当前事务被回滚，会话被中止，会话资源返回给系统；
- 每次会话可消逝时间的限制，如果一次会话期间超过可消逝时间的限制，蛋清事务被回滚，会话被删除，该会话的资源被释放；
- 每次会话的专用 SGA 空间量的限制。

b）用户环境文件

用户环境文件是指定资源限制的命名集，可赋给 Oracle 数据库的有效的用户。利用用户环境文件可容易地管理资源限制。要使用用户环境文件，首先应该将数

据库中的用户分类，决定在数据库中的全部用户类型需要多少种用户环境文件。在建立环境文件之前，要决定每一种资源限制的值。例如：一类用户通常不执行大量的逻辑数据块读，那就可以将 LOGICAL-READS-PER-SESSION 和 LOGICAL-READS-PER-CALL 设置相应的值。在许多情况中决定一用户的环境文件合适资源限制的最好的方法是收集每种资源使用的历史信息。

2、特权与角色

（1）特权

特权是执行一种特殊类型的 SQL 语句或存取另一用户的对象的权力。有两类特权：系统特权和对象特权。

系统特权是执行一种特殊动作或者在对象类型上执行一种特殊动作的权利。Oracle 有 60 多种不同的系统特权，每一种系统特权允许用户执行一种特殊的数据库操作或一类数据库操作。系统特权可授权给用户或角色。一般，系统特权为管理人员和应用开发人员所有，终端用户不需要这些相关功能。授权给一用户的系统特权能被再授予其他用户或角色。反之，可从那些被授权的用户或角色回收系统特权。

对象特区是在指定的表、视图、序列、过程、函数、或包上执行特殊动作的权利。对于不同类型的对象。有不同类型的对象特权。对于有些模式对象，如聚集、索引、触发器、数据库链没有相关的对象特权，它们由系统特权控制。对于包含在某用户名的模式中的对象，该用户对这些对象自动地具有全部对象特权，即模式的持有者对模式中的对象具有全部对象特权。这些对象的持有者可将这些对象上的任何对象特权可授权给其他用户。如果被授权者包含有 GRANT OPTION 授权，那么该授权者也可将齐全了再授权给其他用户。

（2）角色

Oracle 利用角色更容易地进行特权管理。有下列优点：

- 减少特权管理，不要显式地将同一特权组授权给几个用户，只需将这特权组授给角色，然后将角色授权给每一用户。
- 动态特权管理，如果一组特权需要改变，只需修改角色的特权，所有授给该角色的全部用户的安全域自动地反映对角色所作的修改。
- 特权的选择可用性，授权给用户的角色可选择地使其可用或使其不可用。
- 应用可知性，当一用户经一用户名执行应用时，该数据库应用可查询字典，将自动地选择使角色可用或不可用。
- 专门的应用安全性，角色使用可由口令保护，应用可提供正确的口令使角色可用，达到专用的应用安全性。因用户不知其口令，不能使角色可用。

一般，建立角色服务于两个目的：为数据库应用管理特权和为用户组管理特权。相应的角色称为应用角色和用户角色。应用角色是授予的运行一数据库应用

所需的全部特权。一个应用角色可授给其它角色或指定用户。一个应用可有几种不同角色，具有不同特权组的每一个角色在使用应用时可进行不同的数据存取。用户角色是为具有公开特权需求的一组数据库用户而建立的。用户特权管理是受应用角色或特权授权给用户角色所控制，然后将用户角色授权给相应的用户。

Oracle 为了提供与版本的兼容性，预定义下列角色：CONNECT、RESOURCE、DBA、EXP-FULL-DATABASE 和 IMP-FULL-DATABASE。

3、审计

审计是对选定的用户动作的监控和记录，通常用于：

- 审查可疑活动。例如：数据被非授权用户所删除，此时安全管理员可决定对该数据库的所有连接进行审计，以及对数据库的所有表的成功或不成功地删除进行审计。
- 监视和收集关于指定数据库活动的数据。例如：DBA 可收集那些被修改、执行了多少次逻辑的 I/O 等统计数据。

Oracle 支持三种审计类型：

- 语句审计，对某种类型的 SQL 语句审计，不指定结构或对象。
- 特权审计，对执行相应动作的系统特权的使用审计。
- 对象审计，对一特殊模式对象的指定语句的审计。

Oracle 所允许的审计仅限于下列方面：

- 审计语句的成功执行、不成功执行，或者其两者。
- 对每一用户会话审计语句执行一次或者对语句每次执行审计一次。
- 对全部用户或指定用户的活动的审计。

当启动了数据库的审计时，在语句执行阶段产生审计记录。审计记录包含有审计的操作、用户执行的操作、操作的日期和时间等信息。审计记录可存在数据字典表（称为审计记录）或操作系统审计记录中。数据库审计记录实在 SYS 模式的 AUD$表中。

4.3.2 北化科研系统数据库的安全机制

以上我们阐述了 Oracle 数据库系统自身的安全机制，除了 Oracle 系统自身的安全机制，北化科研系统还从以下几个方面来保证数据库系统的安全性：

1、用户管理

在北化科研系统数据库中，我们把用户可以分为本地用户和远程用户两类用户。本地用户需要经过操作系统和数据库的双重认证才能访问数据库；远程用户

要访问数据库的数据必须通过数据库系统的用户验证。

我们还把系统的用户分为管理员和一般用户两类。管理员在得到数据库的验证许可后，可以操作数据库中的数据，可以进行数据的查询、添加、修改、删除等操作。为了日常管理的需要，把管理员分为系统管理员和模块管理员，系统管理员可以对整个科研管理信息系统的所有数据进行操作，但是模块管理员只能操作其有权限的模块。这样处理的好处是提高了系统的安全性，使系统的业务分工明确，数据流明晰，操作简单。

系统的一般用户是相对于管理员来说的，这部分用户通过数据库的验证也可以访问数据库中的数据，但是只能对特定的数据进行查询，并且不能进行修改、删除等重要操作。

2、存取控制

北化科研系统数据库的存取控制由以下几个方面组成：

（1）访问控制。我们设置数据库系统中用户不同，角色不同，能够访问到数据库中的数据就不相同，这同样也就决定了对数据库数据的存取权限的不同。对于什么用户可以访问数据库中的那些表，进行哪些操作，我们都作了限制。例如，系统管理员可以存取整个数据库中的数据，功能模块管理员只能对数据库中涉及到该功能模块的数据进行存取，而作为系统的一般用户只能对其中属于个人权限内的数据进行查询，并且不能进行修改、删除等其他操作。

（2）对数据的存取控制。数据库不同模块中的数据，用户能够对其进行的操作也不同。允许管理员进行查询、插入、修改、删除等操作。普通模块的数据在插入值有错误的时候，允许管理员进行修正；但是特殊模块（例如基础数据模块）中的数据如果出现插入错误的情况，只能由系统管理员来进行修正，并且基础数据模块中数据的删除只能由系统管理员来进行。因为可以对插入错误的数据进行修正，这就有效地保证了数据的准确性、一致性和可用性。

（3）除了上述对数据的控制外，数据库对于要插入到数据库中的数据的类型也进行检验。数据库中表的字段都设定了应该存储的数据类型，插入数据的类型如果与数据库中要求的类型不相符，则插入操作不成功。

（4）防止推理攻击。推理攻击是根据已有的数据，来推理可能的数据。系统根据用户的不同对所能进行的操作加以限制，有效地防止了推理攻击。

（5）用户锁（数据库锁）的利用。当一个用户正在对数据库中某个表中的数据进行操作时，就限制其他用户对此表中数据的插入、修改、删除等操作，但可以对其进行查询。这样就保证了数据库中数据的一致性和可用性，也保证了数据库的安全。

3、跟踪

对进入系统的用户进行跟踪，在数据库中记录用户什么时间进入系统，进行

了那些操作，产生系统日志。这样，可以明确数据库中数据的变化过程，有效地保证了数据的一致性和可用性，同样也可以保证数据的安全。

4、备份

为保障数据库在系统发生故障时数据的完整性和一致性，对系统中的数据定期进行备份，这是保证系统安全的一个重要方面。

4.4 小结

数据库安全是保证数据库完整性、一致性和可用性的重要保障。本章主要从数据库安全的基本问题入手，分析了可能对数据库安全带来威胁的几个方面，以及数据库安全的特点。最后从 Oracle 数据库系统自身的安全入手，介绍了北化科研系统的安全机制。通过对北化科研系统数据库安全性的研究，使我们对数据库安全重要性有了更加深刻的理解，也促使了我们对于信息安全的重视。同时，由于对 Oracle 数据库系统有了进一步的了解，使我们在今后的运用中更加的得心应手。

第五章　科研管理信息系统数据库的优化

北化科研系统的数据库采用的是 Oracle 数据库系统。Oracle 数据库系统提供了相应的应用工具，管理人员可以方便地对 Oracle 进行有效的管理。从而建立一个良好的环境，使系统发挥最大的效能。但是，在北化科研系统实际运行过程中，用户有时还是抱怨系统运行速度慢，对用户查询反应的时间长等问题，这就要求我们对 Oracle 数据库进行优化调整。本章我们首先对数据库优化作了简单介绍，然后论述了北化科研系统在数据库优化方面的措施。

5.1 数据库优化概述

数据库优化[43]是有目的地更改系统的一个或多个组件，使其满足一个或多个目标的过程。对 Oracle 来说，优化是进行有目的地调整组件以改善性能，即增加吞吐量，减少响应时间。简单的说，优化表示向 Oracle 提供适当数据的资源，使用户在完成自己的工作时不用等待所需的数据。

为什么需要优化呢？信息系统很少有系统是在对怎样使用它们和对它们有什么要求的可靠理解下设计出来的。很少有系统在经过建造阶段后没有对设计做过大的变动的。此外，由于在正常的使用（在某些情况下是特殊的使用）下也可能会对系统作出某些更改，存在需要优化的可能。诸如数据分布、更改数据查询方式以及数据量的增加等都会改变系统的性能特性。所以信息系统也需要优化[44]。

北化科研系统的数据库采用的是 Oracle 数据库系统，Oracle 系统的性能会受到硬件和软件的影响，其中有些组件甚至会使 Oracle 出现瓶颈[45]。如计算机的中央处理器（CPU）、内存、存储设备、网络等。解决数据库的瓶颈问题是数据库优化的一个方面，即性能优化；数据库优化的另一个方面是数据库的环境优化。下面我们从这两个方面来简单介绍数据库的优化。

5.1.1 数据库性能优化

在实际运行当中，影响数据库性能的因素有很多，如系统、数据库和网络等。为了保证 Oracle 数据库运行在最佳的性能状态下，在进行任何系统开发之前就应该考虑数据库的优化策略。优化策略一般包括服务器操作系统参数调整、Oracle 数据库参数调整、网络性能调整、应用程序 SQL 语句分析及设计等几个方面，其

中应用程序的分析与设计是在信息系统开发之前完成的。

分析评价 Oracle 数据库性能主要有数据库吞吐量、数据库用户响应时间两项指标[46]。数据库吞吐量是指单位时间内数据库完成的 SQL 语句的数目；数据库用户响应时间是指用户从提交 SQL 语句开始到获得结果的那一段时间。数据库用户响应时间又可以分为系统服务时间和用户等待时间两项，即：

数据库用户响应时间 = 系统服务时间 + 用户等待时间

上述公式告诉我们，获得满意的用户响应时间有两个途径：一是减少系统服务时间，即提高数据库的吞吐量；二是减少用户等待时间，即减少用户访问同一数据库资源的冲突率。

1、数据库性能优化的几个方面

（1）调整操作系统参数[47]

例如：运行在 UNIX 操作系统上的 Oracle 数据库，可以调整 UNIX 数据缓冲池的大小；每个进程所能使用的内存大小等参数；允许的最大进程数等。

（2）调整服务器内存分配

对内存的管理，主要是从合理分配交换空间的使用；物理页调换的效率；高速缓冲区利用率如何等方面分析。

内存分配是在应用系统运行过程优化配置的，数据库管理员可以根据数据库运行状况调整数据库系统全局区（SGA）的数据缓冲区、日志缓冲区和共享池的大小；还可以调整程序全局区（PGA）的大小。需要注意的是，SGA 不是越大越好，SGA 过大会占用操作系统使用的内存而引起虚拟内存的页面交换，反而会降低系统效率。

（3）调整硬盘 I/O

在应用系统开发之前，数据库管理员可以将组成同一个表空间的数据文件放在不同的硬盘上，做到硬盘之间 I/O 负载均衡。

（4）调整应用程序的整体设计

应用程序设计的调整修改原则：根据不同的业务需求，从不同角度设计数据库，其设计核心是对数据库的访问操作。

实际上，上述数据库优化的几个方面之间是相互联系的。Oracle 数据库性能恶化表现基本上都是用户响应时间比较长，需要用户长时间等待。但性能恶化的原因却是多种多样的，有时是多个因素共同造成了性能恶化的结果，这就需要数据库管理员有比较全面的计算机知识，能够敏感地觉察到影响数据库性能的主要原因所在。另外，良好的数据库管理工具对于优化数据库性能也是很重要的。

2、常用的数据库性能优化工具

（1）操作系统工具，例如 UNIX 操作系统的 vmstat，iostat 等命令可以查看到系统的系统级内存和硬盘 I/O 的使用情况，这些工具对于管理员弄清楚系统瓶颈出

现在什么地方有时很有用。

（2）Oracle Enterprise Manager（OEM），这是一个图形的用户管理界面，用户可以使用它方便地进行数据库管理而不必记住复杂的 Oracle 数据库管理命令。

（3）Oracle 数据库在线数据字典，Oracle 在线数据字典能够反映出 Oracle 动态运行情况，对于调整数据库性能很有帮助。

（4）SQL 语言跟踪工具（SQL TRACE FACILITY），SQL 语言跟踪工具可以记录 SQL 语句的执行情况，管理员可以使用虚拟表来调整实例，使用 SQL 语句跟踪文件调整应用程序性能。SQL 语言跟踪工具将结果输出成一个操作系统的文件，管理员可以使用 TKPROF 工具查看这些文件。

（5）EXPLAIN PLAN——SQL 语言优化命令，使用这个命令可以帮助程序员写出高效的 SQL 语句。

5.1.2 数据库环境优化

优化环境[48]是一个高效数据库必不可少的一部分。如果数据库环境没有被优化，数据库环境可能不顾事务的效率而消极影响用户执行的每一个事务。数据库环境优化主要从内存优化、输入/输出（I/O）优化、SQL 语句调整以及初始化参数等方面来进行。

5.1.2.1 内存优化

数据库内存优化可以从以下几个方面来进行：调整系统全局区（SGA）共享池、调整数据库缓存、调整重做日志缓冲区、调整排序等。

1、调整系统全局区（SGA）共享池

当 Oracle 数据库启动后，系统将为每个数据库服务器分配一个数据库的控制信息内存段，称为系统全局区（SGA）。SGA 的大小直接影响数据库系统对内存使用的性能。SGA 的共享池是一个高速缓冲结构。和所有其他的高速缓存结构一样，它是内存驻留数据结构。

对于几乎所有的高速缓存装置来说，保证高效性能的准则是命中大于 90%。高速缓存通常有一个最近最少使用的（LRU）算法来管理，该算法保证在任何给定的时刻，最近使用最多的（MRU）数据被保存在高速缓存中，并且最近最少使用的数据被拿走。

Oracle 的 SGA 中共享池部分有库高速缓存（library cache）、字典高速缓存（dictionary cache）和会话信息组成。在调整 Oracle 的内存时，共享池是最大的消耗成份。

调整 Oracle 数据库的 SGA，可以通过初始化参数文件中的 shared_pool_size 项的值来完成。共享池的大小不是精确值，确定共享池大小时需要考虑诸如并存的用户数、存储过程的使用扩展、专用同义词和授权公用同义词的使用以及 SQL 代码写的一致性等因素。

2、数据库缓冲区高速缓存调整

数据库缓冲区高速缓存是影响整个数据库系统运行的重要因素之一。数据库缓冲区高速缓存是由于 Oracle 块相同大小的内存块组成。所有 Oracle 操作的数据在使用前被装入到数据库缓冲区高速缓存中。数据的更新在内存块中完成。由于这个原因，合适大小的缓冲区高速缓存是至关重要的。访问内存的速度要比访问磁盘快几百倍。

Oracle 缓冲区高速缓存的大小是由 DB_BLOCK_SIZE 和 DB_BLOCK_BUFFERS 两个初始化参数决定的。DB_BLOCK_SIZE 参数用于在数据库创建时，设置 Oracle 块大小；DB_BLOCK_BUFFERS 参数决定分配给缓冲区高速缓存的块的数量。调整缓冲区高速缓存主要是对这两个参数的调整。

3、重做日志缓冲区调整

重做日志缓冲区用于在内存中存储未被刷新写入联机重做日志文件的重做信息。它是循环使用的缓冲区，这意味着从顶端到底端填充信息，然后又返回到缓冲区的起始点。当重做日志缓冲区填满时，将它的内容写到联机日志文件中。

重做日志缓冲区的大小是由 LOG_BUFFER 初始化参数决定的，以字节为单位，决定在内存中保留多少空间为缓存重做日志项。如果这个值设置的过低，进程之间相互竞争，日志写入（LGWR）进程读出和写入缓存，有可能会导致性能问题。

由于重做日志缓冲区是共享内存的一部分，因而 Oracle 利用闩（latches）来管理这部分高速缓存。闩对内存储器的作用就像锁对数据的作用一样，闩是用来保护 SGA 中共享数据结构的简单低级的串行机制，要求它们在非常短的时间内对存储器中的共享结构进行处理。重做分配闩（redo allocation latch）管理对重做日志缓冲区的空间请求，而重做拷贝闩（redo copy latch）管理将存于重做日志缓存区中的重做日志写入在线重做日志文件中。

将重做日志信息读入重做日志缓冲区，将重做日志缓冲区中的重做日志信息写入重做日志文件，保证这两者没有等待对于调整进程是很重要的，这同时也是重做日志缓冲区调整的目的之一。

4、排序区域大小调整

在 Oracle 数据库系统中，排序活动需要用到某些存储器块，排序活动包括在内存上的排序操作和在磁盘上的排序操作。初始化参数文件中的 SORT_AREA_SIZE 用于控制排序活动在内存中的配置大小。排序可由程序显式的提出，或由 Oracle 隐含地执行，应该希望尽可能多的排序活动在内存中进行，因为在内存上的排序操作比在磁盘上的排序操作要快得多。增加 SORT_AREA_SIZE 的值，对 SQL 中索引的创建和 ORDER BY 语句是有好处的。

5.1.2.2 输入/输出（I/O）优化

输入输出[49]（I/O）是数据库性能的最重要的方面，如果考虑到数据库是信息的组织者，接收者和分发者，那么显然数据库的重要功能就是向磁盘读写信息。数据写入和读出磁盘驱动器的开心与这些数据在磁盘上的存放点是密切相关的。设计数据库对象的布局和属性，即设计数据库中表空间、表、索引、重做日志文件和回滚段在磁盘上的分配是特别重要的。

1、表和索引段

表和索引对象之间应遵循的一条规则是：从表中分离（splitting）索引，并且表和其索引使用的数据文件应该存放在独立的磁盘上，以避免查询期间的竞争。如果索引和表存放在同一个驱动器上，对表索引访问的查询和对该表的访问有可能会引起 I/O 冲突。这可能会极大地降低系统的数据吞吐量。

随着将表和其索引分开放在独立的磁盘上，还要避免在同一个磁盘上存入不同的表。例如，如果同时访问两个表，将每个表存放在各自的磁盘上会改进性能。这种方法也同样适用于索引，如果一条 SQL 语句同时使用两个索引，将每个索引存放在各自的磁盘上也将会改进性能。

2、回滚段调整

回滚段（rollback segment）是一个数据库对象，它保存着进行数据操纵活动（insert、update、delete）时的信息，以便这些活动在需要的时候可以回滚到活动前的状态。当一个活动被回滚后，对数据的更改都被取消，数据返回到此操作发生前的状态。

应该创建回滚表空间来保存回滚段。为控制回滚段的使用，必须在初始化参数文件中指定它们。Oracle 数据库赋予事务回滚段的的顺序与它们在初始化参数文件中列出的顺序相同。因此应该调整回滚段的顺序以使第一个回滚段位于第一个表空间中而下一个回滚段位于另一个表空间。也就是说，相邻的两个回滚段不在一个表空间上。用这种方法，在处理一个事务时，下一个事务可以利用位于另一

个盘上的表空间中的某个回滚段。这样，可以减少磁头争用，可以在各磁盘之间分配 I/O 请求。

3、数据库碎片

数据库管理员面对的一个最常见的问题就是处理数据库对象的碎片。碎片浪费空间，导致性能问题，并给数据对象的管理带来更大的困难，数据库碎片问题包括分裂成碎片的表空间、分裂成碎片的数据库对象、链接行和转移行。

（1）分裂成碎片的表空间

表空间变成碎片是由于错误以及表空间中的数据库对象的无计划撤销和重建造成的。处理表空间的最好的办法就是通过仔细地计划和管理来避免表空间碎片的产生，这是需要额外的开销的。如果面临的是整理分裂碎片的表空间，最简单的方法是在表空间中导出所有的对象，删除此表空间，重新建立表空间，在将导出的对象倒入回去。这不仅能够聚合自由空间，而且导入操作也可以将数据库对象聚合到一个区域中。

（2）分裂成碎片的数据库对象

分裂成碎片的表空间令数据库的性能下降，但我们也容易忽视碎片问题发生在对象级。许多 DELETE 操作会产生一些很少被再利用的块，这些块不满足被放回自由表的条件。这就产生了对象级的碎片。在对象级的碎片中，对象在表空间级可能看起来很好，并且有一个单独的区域，但是在这个单独的区域中的检查可能会发现一些问题。

对象碎片会导致下面的问题：

- 由于在表和索引块中那些很少被再利用的块不能放回自由表空间导致空间的浪费；
- 读性能下降，因为数据不再被紧紧地排在一起，物理磁盘驱动器必须从一个比需要大的磁盘表面区域中搜寻和读数据。

为了解决对象碎片，最好的方法只能是导出表空间中所有的对象，重建表空间并将所有的对象再导入回去。

（3）行转移和行链接

行转移是在对行的修改引起行的长度比块中可利用空间大时发生的。当发生行转移时，行的一部分就转移到一个新的数据块中。这个新单元的地址存储在原始行单元中。当 Oracle 需要读这个行时，它首先读原始单元并找到一些数据和指向另一个块的指针，再在这个块中找到剩下的行数据。

行链接是在一个行太长以致于不能放入任何一个数据块时发生的。这导致该行被存储在一个或多个数据块的链中。行链接经常在包含 LONG、LONG RAW 数据类型的大行发生。

行转移和行链接会发生两个问题：

- Oracle 必须执行至少一次额外的 I/O 来获取行转移或链接的行，除数据块中包含的行数据外，它还必须读包含着行数据的块的指针。
- Oracle 必须和行数据一起存储额外的指针来供给行转移或行链接机制，这种状况会浪费一些磁盘空间。

解决行转移，最好的办法是将表中的数据全部导出，删除表，之后使用重建该表，但这种方法比较费时。解决行链接的办法是缩短表的行长或者增加数据块的大小。选择前一种方法通常需要对表结构进行重新设计，但是后一种方法需要导出整个数据库，重新创建然后全部导出。这两个方法都不是很好，但是如果一个表中的链接行引起了性能问题，那么也只能这样做了。

4、重做日志调整

重做日志（Redo Log）是在数据库运行时记录每个操作活动的事务日志，表明每个处理事务是否被提交或回滚。系统至少要有两个重做日志组，以便当一个日志组填满时，能有另一个日志组可用。如果系统是一个繁重的面向事务的系统，重做日志组会经常被写入。Oracle 的高性能部分在于这样一个事实：它不将对数据库的修改直接写入数据库，而是写入重做日志。由 DBWR 进程在出现检查点时把信息写入实际的数据库文件。检查点的出现频率由初始化参数文件的 LOG_CHECKPOINT_INTERVAL 参数确定。

因为重做日志经常要写入、应将它们与其它对象和表空间分开。因此，如果硬件条件允许，应将重做日志放在一个与所有的数据库对象都分开的磁盘上。

重做日志组由成组成员组成，每个组中的成员数目相同，各组成员的大小也相同。每个组中成员的最大数目由操作系统决定。在每个重做日志组中，应设有多个组成员，这样就不会因为某个日志文件的损坏而影响整个数据库系统。

5.1.2.3 SQL 语句调整

一个良好设计的应用程序，如果使用的 SQL 结构[50]不理想，仍然会遇到新能问题。在一个正确设计的数据库中，应用程序设计及 SQL 问题会引发大多数的性能问题。

SQL 语言是一种灵活的语言，相同的功能可以使用不同的语句来实现，但是语句的执行效率是很不相同的。程序员可以使用 EXPLAIN PLAN 语句来比较各种实现方案，并选出最优的实现方案。总的来讲，写 SQL 语句需要考虑如下规则：

1、避免无计划的全表扫描，尽量使用索引。

（1）避免对返回的行无任何限定条件，即不使用索引列进行查询；

（2）避免条件列在表达式中使用；

（3）避免条件中使用 NULL 或不等于（!=）；

（4）条件使用 like 操作时，应避免使其值以‘%’开始或是一个赋值变量。

2、选择联合查询的联合次序。联合查询的主表要进行整个表数据的扫描，所以主表应该数据量最小。联合条件要充分考虑表的索引、表的数据量大小。

3、在子查询中慎重使用 in 或者 not in 语句，使用 exists 和 not exists 的效果要好的多。

4、慎重使用视图的联合查询，尤其是比较复杂的视图之间的联合查询。一般对视图的查询最好分解为对数据表的直接查询效果要好一些。

5、避免在 SQL 里使用 PL/SQL 功能调用。

6、限制对远程表的访问。

SQL 优化的实质是在结果正确的前提下，用优化器可以识别的语句，充分利用索引，减少表扫描的 I/O 次数，尽量避免全表扫描的发生。其实 SQL 的性能优化是一个复杂的过程，上述这些只是在应用层次的一种体现，深入研究还会涉及数据库层的资源配置、网络层的流量控制以及操作系统层的总体设计。

5.1.2.4 几个关键的初始化参数

下面是建立一个健全的数据库环境，需要注意的几个关键参数[51]：

1、DB_BLOCK_SIZE 数据块大小

该参数是在创建数据库时设置的，它决定数据库里每个块的大小。DB_BLOCK_SIZE 的缺省值是 2048 字节（2KB），若操作环境运行，应该设为 4KB、8KB 或更高。通常，每个双倍大小的数据库块会减少大约 40%的 I/O 密集的批处理操作所要求的时间。

2、DB_BLOCK_BUFFERS 数据块缓冲区

该参数在数据库块里设置 SGA 中的数据块高速缓存存储器的大小。数据块高速缓存越大，为用户已经在内存里的共享数据提供的内存越大，则可以减少所需要的物理读。该参数值与用户的数目有关。

3、 SHARED_POOL_SIZE 共享池大小

该参数按字节设置 SGA 里的共享池的大小。在 init.ora 文件中为其提供了 3 种缺省值：SMALL、MEDIUM 和 LARGE。一般缺省设置为 LARGE 即可。若有许多用户，每次增大 DB_BLOCK_BUFFER 时，应该增大 SHARED_POOL_SIZE 参数值。使用的过程对象越多，共享池就应该越大。

4、 LOG_BUFFER 注册缓冲池

该参数按字节设置 SGA 里重写日志缓冲器的大小。对于操作系统，其缺省设置是数据库最大块尺寸的 4 倍。如果在 V$SYSSTAT 里的“redo log space requests”（重写日志要求）统计值是非零的，就应该增大 LOG_BUFFER 设置值以支持加载事务，避免被迫等待访问重写日志缓冲区。

5、DBWR_IO_SLAVES 数据库读写/输入输出/从属

操作系统允许使用多个 DBWR I/O 从属过程时，就将该参数设为大于 10 的值。

6、DB_FILE_MULTDBLOCK_READ_COUNT 数据库文件多块读计数

该参数确定在全表扫描期间数据库每一次读多少块。其值的设置，应使得读操作期间能最充分的利用操作系统的缓冲区。

7、SORT_AREA_SIZE、SORT_AREA_RETAINED_SIZE

- SORT_AREA_SIZE 排序区大小参数：按字节指定了提供给用户用于排序的内存总数的最大值。
- SORT_AREA_RETAINED_SIZE 排序区保留大小：按字节设置了最大值，这是用来排序的内存总数的最大值。

如果排序区域不够大，Oracle 会在排序操作期间开辟临时段，从而降低查询进行排序的性能。可以通过检测 V$SYSSTATK 中的“sort（memory）”和“sorts（disk）”的统计值来确定与排序有关参数是否设置的足够大。若排序要求的临时段比给予内存排序的百分比多 5%~10%，那么就应该增大两参数的值。

8、SORT_DIRECT_WRITES 排序直接写

为满足在排序操作期间，能够利用排序直接写技术向临时段里写，在 init.ora 文件中一般将此参数设为 TRUE。

9、ROLLBACK_SEGMENTS 回滚段

为了使当地支持数据库里的事务。必须有足够的回滚段。数据库里发生的当前事务越多，需要的回滚段就越多。

5.2 北化科研系统数据库的优化

Oracle 的数据库的优化没有绝对指标，即不能用一个绝对的数量来定义数据库优化的结果，应当用优化前后数据库的各种性能指标的对比、尤其是 sql 语句的执行速度，sql 语句带来的系统负担，应用的响应速度，甚至应用所服务的终端用户的直接感受等来衡量 Oracle 数据库优化的结果。基于数据库实例级的调整，大多用来解决数据库结构性故障，相应的也能解决因为结构性故障带来的普遍性的性

能问题。而大多数情况下，用户所真正关心的，切身感受到的单点数据库响应慢的问题往往是不良的 sql 语句，过期的统计数据和其导致得不良的执行计划，不良的数据表结构，过度的触发器使用，不良的应用同步所造成的。

5.2.1 北化科研系统数据库优化的方法

北化科研系统数据库是采用 Oracle 数据库来实现的。首先，我们对 Oracle 数据库的实例进行分析，判断数据库的物理分布，内存配比，数据库的服务模式是否合理，其次实施跟踪 sql 语句的执行，分析其执行计划，在其执行速度与执行期的系统消耗之间做出选择。

1、数据库的物理布局

理论上，在可能的情况下，为了获得尽可能多的 I/O 值，应当使各种数据库文件分布到不同的磁盘上。但是，如果单磁盘 I/O 能满足负载的 I/O 需求，那么除了基于安全考虑应当另行分布的日志、控制文件外，所有的数据文件在一个磁盘上也是可以认为是正常的，合理的。因为北化科研系统数据库的数据文件存放在单磁盘上也可以满足 I/O 负载的需求，所以，系统的数据库数据文件没有分开存放。但是因为安全因素数据库的日志文件、控制文件以及备份数据是分开存放的。

2、数据库的 SGA、UGA 和 PGA 的分布

为了使北化科研系统数据库的性能满足用户的需求，我们对数据库的基本参数作了调整。

- db_block_buffer(数据库高速缓存)设置的足够大。
- share_pool_size(数据库共享池)的值设置的比较适中，因为如果太大（超过 600MB）时会对 cpu 管理内存造成负担。
- sort_area_size（排序区）设置合理，因为系统是在 dedicate 模式下，再此模式下 PGA 不能超过所能使用的最大内存。PGA（用户进程所使用的内存总数)的计算方式为：

  PGA = process*(sort_area_size+2MB)

- large_pool_size（大型池）设为 500MB。因为在 dedicate 模式下，该内存区域存储 RMAN，以及大型的 plsql，所以在内存允许的条件下要设置的足够大。

3、北化科研系统数据库的服务模式

数据库的服务模式对数据库的性能有很多的影响。当并发用户数>400 时可以考虑采用 mts 模式，当并发用户数<400 时应当采用 dedicate 方式，当并发

用户数>400 且系统内存资源可用时应当采用 dedicate 模式，当并发用户数<400 且内存资源不够用时应当采用 mts 模式。根据实际情况，北化科研系统的数据库采取的是 dedicate（专用服务器）模式。不同模式的优点和弊端如下表（表 5-1）：

表 5-1 dedicate 和 mts 模式的优点和弊端

Table.5-1 the virtue and abuse of delicate or mts mode

不同方式	优 势	弊 端
dedicate	cpu 占用相对少	内存占用大
mts	cpu 占用相对大	内存占用小

4、跟踪系统的等待事件

在 Oracle 数据库中，当一个进程，或者该进程所代表的寄生于该进程的 session 在等待某系统资源时，就会触发一个 oracle 内部的事件。该事件将会被登记到 v$session_wait, v$system_wait 的 oracle 内部视图上，通过编写程序分时段定期跟踪该视图，可以收集定位出系统运行期内的资源瓶颈。通过系统的结构调整可以消除或者减弱该瓶颈对系统运行的影响。

同时，我们执行 oracle 数据库系统自带的诊断分析工具 statspack 来跟踪分析数据库系统的运行状态。监察系统的性能是否达到理想的状态。

5、跟踪、优化不良的 sql 语句

北化科研系统在运行之初，造成应用程序反应缓慢，系统资源消耗大的直接原因是不良的 sql 语句。一个不良的 sql 语句可以导致整个数据库系统高负荷运转，甚至对外暂停服务。所以我们对系统数据库初期优化的重点集中于跟踪、优化不良的 sql 语句。

方法如下：

（1）以系统资源消耗为着眼点（Windows 系统）

- 根据资源管理器得到系统资源（cpu/io）占用最高的进程号 npid;
- 以 sysdba 身份登陆 oracle 数据库；
- 执行以下 sql 语句得到相应的 oracle 数据库内部的 session id: nsid，查询语句如下：

  select vss.sid from v$session vss, v$process vp where vp.addr = vss.paddr and vp.spid = npid
- 执行以下 sql 语句得到该 session id 执行的 sql 语句，查询语句如下：

  select vsql.sql_text from v$sqltext vsql, v$session vss where vss.sql_address = vsql.address and vss.sid = nsid order by piece

（2）以数据库内部统计为着眼点

查询 oracle 数据库内部统计视图，得到系统占用资源最多的 sql：

select sql_text,, ROWS_PROCESSED,, (DISK_READS+BUFFER_GETS), BUFFER_GETS,, DISK_READS from v$sqlarea where rownum<5 order by (DISK_READS+BUFFER_GETS)

（3）以特定用户为着眼点

- 首先确定要跟踪的用户 id，语句如下：
  select sid, serail# from v$session where username = <所定义的用户名>
- 对该特定用户启用时间计数器，语句如下：
  exec sys.dbms_system.set_bool_param_in_session (sid, serial#, 'timed_statistics', true)
- 对该特定用户指定跟踪文件所在地，语句如下：
  exec sys.dbms_system.set_int_param_in_session(sid,serial#,'max_dump_file_size', 2147483647)
- 对该特定用户启用跟踪，语句如下：
  exec sys.dbms_system..set_ev(sid,serial#,10046,8,'')
- 一段时间后对该特定用户关闭跟踪，语句如下：
  exec sys.dbms_system.set_ev(sid,serial#,10046,0,'')

由上述可得到优化方法如下：

（4）得到该 sql 语句的执行计划

- 根据上述跟踪方法的第一、二两点得到的 sql 语句，在模拟环境执行得到执行计划具体的执行步骤。执行如下 sql 语句：
  alter system set timed_statistics = true
  set autotrace traceonly
  之后，执行 sql 语句，会得到该 sql 语句的执行计划。
- 根据上述跟踪方第三点得到的跟踪文件，运行如下命令，得到执行计划
  tkprof <跟踪文件名> <目标文件名>

（5） 根据执行计划，分析 sql 执行计划中的对大数据量表的全表扫描，以及对大量行纪录的函数操作、group by 操作。之后调整 sql 语句的 from、where 字句，或者 group by 字句的位置，重写 sql，减少大表全表扫描，减少大量数据行的函数操作，group by 操作。在此基础之上，分析表结构，评估新建索引带来的 insert,update,delete 的额外消耗，确定是否需要添加必要的索引。

（6）如果数据库系统的应用环境允许重新分析相关的统计信息，那么在执行步骤 1，2 之前需要重新收集系统的相关统计信息到数据字典，以使 oracle 在"选择"模式下，采用合理的优化器，生成合理的执行计划。

5.2.2 数据库连接池的设计

由于北化科研系统是一个多用户条件下的系统，基于对象的创建和删除会带来效率上的巨大损失，所以在系统运行过程中应当尽量减少对象的创建与删除。本系统中，在多用户情况下，所可能大量出现的对象就是数据库连接（java.sql.Connection），以及大量的数据对象。后者是不可避免的，但可以使用游标的方法减少该问题的影响。对于前者就必须使用数据库连接池。我们对数据库连接池的总体构想是这样的：系统启动时，连接池启动，同时保证系统中在任何时间只有一个数据库连接池的实例。该连接池，按照系统管理员的配置，生成规定数量的数据库连接对象，放入一个先进后出的栈中，用户调用时使用 getCon() 方法得到该连接池的一个连接，在使用完毕后使用 releaseCon()方法释放对该连接的占用。

北化科研系统数据库连接池有如下要求：

（1）任何时间内，在系统中只有一个连接池工厂实例

（2）该连接池的所有方法应当是线程安全的，也就是说多线程情况下该连接池应当运行稳定。

根据以上要求，我们对系统数据库连接池的设计采用了工厂模式和单例模式、多例模式相结合的方式进行设计。同时在设计过程中代码的每一个方法都是线程安全的。（数据库连接池的代码见附录 2）

5.2.3 优化效果对比

（1）优化前后的 SQL 执行时间对比如图 5-1 所示：

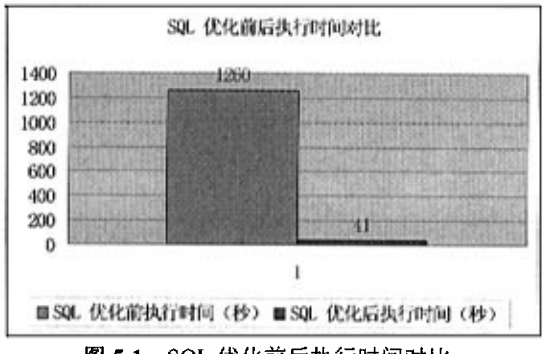


图 5-1 SQL 优化前后执行时间对比

Fig.5-1 the execute-time contrast in the before and after SQL optimization

（2）优化前后的 cpu 采样对比如图 5-1 所示：

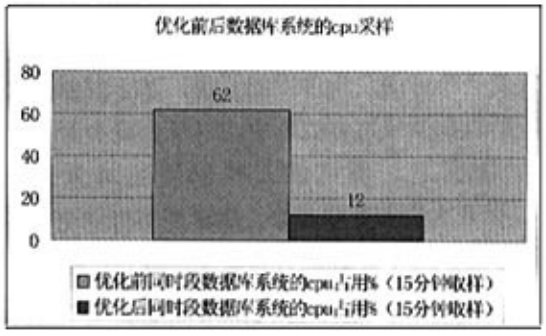


图 5-2 优化前后数据库 CPU 采样

Fig.5-2 the sampling of CPU in the before and after optimization

5.3 小结

Oracle 数据库的优化工作是一个长期的、复杂的、循环往复的工作过程。Oracle 数据库参数的调整能够解决数据库存在的普遍性的故障。但是每一次数据库结构，数据表结构，数据库统计信息的改变都有可能影响 oracle 数据库的 sql 语句执行计划。这种影响有可能是良性的，也有可能是恶性的；有可能造成执行迅速的 sql 变得执行缓慢，并且消耗大量的系统资源。所以对于数据库定期的优化是必要的，sql 执行所依赖的执行计划也不应当是一成不变的。

第六章 结束语

北京化工大学科研管理系统是以 Oracle 数据库为中心、采用 B/S 结构、基于 J2EE 技术管理平台，它的开发成功能提高了我校的科研管理工作水平，为科研管理工作人员提供一个实用、统一、方便的工作的工作平台。在数据库的设计与实现过程中，主要实现了以下内容：

一是对数据库的产生与发展现状作了概括性的介绍，重点对管理信息系统中数据库的地位和作用进行了论述。

二是通过对数据库的设计、安全和优化基础知识的介绍，重点论述了北京化工大学科研管理信息系统中数据库的设计、安全策略和优化方法。首先，在数据库系统的设计中，我们严格按照数据库设计的步骤，规范化的设计，使数据库系统具有良好的效率。其次，在安全策略方面，把 Oracle 数据库自身的安全机制与本系统的安全策略相结合，实现了系统的安全可靠。最后，在系统性能优化上，除采取优化数据库系统参数，优化 SQL 语句和执行结果等措施外，设计实现了数据库连接池，提高了数据库系统的连接速度，满足了用户实际操作中效率的要求。

北京科研管理信息系统试运行通过，成功应用。从整个系统来看，功能比较全面，包括了对科研管理信息的绝大部分业务，具有良好的通用性。从整个系统来看，功能比较全面，包括了对科研管理信息的绝大部分业务，具有良好的通用性。所以，整个系统的成功给了我们项目小组的成员是极大的鼓舞，为我们的下一步研究和学习奠定了扎实的基础。

总之，我们在管理信息系统的开发过程中积累了丰富的经验，对于今后开发同类产品提供了很好的借鉴作用。然后，由于软件技术的日益月新，我们还得继续学习，力求做得更好。

参 考 文 献

[1] 倪凯民, 唐弋清. 基于客户机/服务器环境下的MIS系统开发[D]. 成都: 计算机应用研究, 1997(3).

[2] 萨师煊, 王珊. 数据库系统概论(第三版)[M]. 北京: 高等教育出版社, 2003.

[3] 张龙祥, 等. 数据库原理与设计(第一版)[M]. 北京: 人民邮电出版社, 2002.

[4] 王鹏, 董群. 数据库技术及其应用[M]. 北京: 人民邮电出版社, 2000.

[5] Bernstein P, Dayal U, EwWitt DJ, Gawlick D, etc. Future directions in DBMS research—The laguna beach participants [R]. SIGMOD Record, 1989, 18: 17-26.

[6] Silberschatz A, StoneBraker M, Ullman JD. Database systems: Achievements and opportunities[J]. CACM, 1991, 34(10):110~120.

[7] Silberschatz A, StoneBraker M, Ullman JD. Database research, achievements and opportunities into the 21st century[R]. SIGMOD Record, 1996, 25(1):52~63.

[8] Siberschatz A, Zdonik SB. Stratetic directions in database systems—Breaking out of the box[J]. ACM Computing Surveys, 1996, 28(4):764~778.

[9] Bernstein P, Brodie ML, Geri S, DeWitt DJ, FrankLin MJ, etc. The asilomar report on database research[R]. SIGMOD Record, 1998, 27(4):74~80.

[10] Abiteboul S, AgraWal R, Bernstein P, Carey M, Ceri S, croft B, DeWitt D, Franklin M, Molina H, etc. The Lowell Database Research Self-Assessment Meeting[R], Lowell Massachusetts, 2003.

[11] Meng XF. Research on the technology of Web information integration[J]. Computer Applications and Software, 2003, 20(11):32~36(in Chinese with English abstract).

[12] Meng XF, Luo D, Lee ML. An J, OrientStore: A schema based native XML storage system[A]. In: Freytag JC, Lockemann PC, Abiteboul S, Carey MJ, Selinger PG, Heuer A, etc. Proc. of the 29th Int' 1 Conf. on Very Large Data Bases (VLDB). Berlin: Morgan Kaufmann. 2003. 1057~1060.

[13] Bobineau C, Bouganim L, Pucheral P, Valduriez P. PicoDBMS: Scaling down database techniques for the smartcard[J]. In: Abbadi AE, Brodie ML, Chakravarthy S, Dayal U, Kamel N, Schlageter G, Whang K-Y, eds. Proc. of the 26th Int' 1 Conf. on Very Large Data Bases (VLDB). Morgan Kaufmann, 2000. 11~20.

[14] Arumugam S, Nagarajan K. Survey of small footprint databases[R]. Technical Report, CIS 6930. Distributed Database Systems, 2000.

[15] Wetzel M, Stuttgart U, Informatik F. Small footprint databases[R]. Technical Report, University Stuttgart, 2003.

[16] Meng XF, Wang S, Wong KF. Overview of a Chinese natural language interface to databases: Nchiql[J]. IJCPOL, 2001, 14(3):213~232.

[17] Meng XF, Liu S, Wang S. Word segmentation based on database semantic in Nchiql[J]. Journal of Computer science and Technology, 2000, 15(4):346~354.

[18] 薛成华. 管理信息系统(第三版)[M]. 北京: 清华大学出版社, 2000.

[19] Wu Zhenghong. Management Information System Based on Data Warehouse[J]. Computer System Application, 1997(12).

[20] 刘卫宏. SQLServer2000实用教程[M]. 科学出版社, 2004.

[21] Ruth Baylis, Kathy Rich. Oracle Database Administrator's Guide, Release 2[M]. Oracle Corporation, 2002.

[22] Peter Rob(美), Elie Semaan(美). 数据库设计与开发教程(第一版)[M], 北京: 电子工业出版社, 2002.

[23] 王珊, 陈红. 数据库系统原理教程[M], 北京: 清华大学出版社, 1998.

[24] Ryan K, Stephens, Ronald R, Plew. Database Design[M], Macrnillan Computer Publishing U.S.A, 2001.

[25] 施伯乐, 等. 数据库技术[M]. 科学出版社, 2002.

[26] Abraham Silberschatz, Henry F. Korth, S. Sudarshan. Plew. Database System Concepts, Fourth Edition [M]. McGraw-Hill Companies, Inc, 2002.

[27] John Loney 著, 李晓军等译. Oracle8 数据库管理员手册(第 1 版) [M]. 北京: 机械工

业出版社, 1998.
[28] Wiseman Simonm. Security for Distributed Databases[R]. Information Security Technical Report, 2001, 6(2): 30-43.
[29] 吴基传. 信息技术和信息产业(第一版)[M]. 北京: 新华出版社, 2000.
[30] 关义章, 等. 信息系统安全工程学[M]. 北京: 电子工业出版社, 2002.
[31] Silvana Castano, etc. Database Security[M]. Addison-Vesley Publishing Company, 1994.
[32] 刘启原, 刘怡. 数据库与信息系统的安全(第 1 版)[M]. 北京: 科学技术出版社, 2000.
[33] R. Sandhu. Role-Based Access Control[M]. Advance in Computers, 1998.
[34] 高品均, 何光新. 数据库安全与保密[J]. 电子计算机, 1996(1): 33-36.
[35] 于慧龙. 网络信息系统的安全威胁分析[J]. 互联网世界网络安全专利, 2001(9): 22-24.
[36] Glover C, Mukkamala R. Multilevel Secure Database-A New Approach[J]. IEEE Transaction on Database Security, 1991, 3: 691-694.
[37] 秦拯, 吴中福, 等. 数据库系统安全体系结构探讨[J]. 电脑与信息技术, 1999(6): 13-15.
[38] Bertino E, Jajodia S, Samarati P. Database Security: Research and Practice[J]. Information Systems, 1995, 20(7): 537-556.
[39] 米雁辉, 等. 网络安全与黑客(第 1 版)[M]. 北京: 航空工业出版社, 2001.
[40] 王鹏, 董群. 数据库技术及其应用(第 1 版)[M]. 北京: 人民邮电出版社, 2000.
[41] 刘怡. 数据库安全与安全数据库管理系统设计[J]. 网络安全技术与应用, 2001(1): 28-40.
[42] 王晓萍. Oracle 数据库安全控制策略[J]. 计算机技术与自动化, 2001, 3.
[43] Gaja Krishna Baidyanatha(美), Kirtikumar Deshpande(美), John Kostelac(美)著. 钟鸣, 石永平, 等译. Oracle 性能优化技术内幕(第 1 版)[M]. 机械工业出版社, 2002.
[44] Brown, Lynnwood. Oracle database administration on Unix system[M]. Upper Saddle River, N.J. : 1997.
[45] 马尔.奥克拉(美). Oracle8.1.6 管理员指南[M]. 北京: 北京希望电子出版社, 2000.
[46] 齐学忠. 信息系统中的数据库设计与性能优化[J]. 计算机工程与应用, 2000, 11, 176.
[47] Sumit Sarin(美). Oracle 数据库管理员技术指南[M]. 北京: 机械工业出版社, 2001.
[48] Eyal Aronoff(美). Oracle 8 性能优化和管理手册[M]. 北京: 机械工业出版社, 2000.
[49] Joline Morrison(美). Oracle 数据库指南[M]. 北京: 机械工业出版社, 1999.
[50] T. Urhan, M.J. Franklin, L. Amsaleg. Costbased Query Scrambling for Initial Delays. SIGMOD, 1998.
[51] 布里森(美). Oracle9i UNIX 管理手册[M]. 北京: 机械工业出版社, 2003.

附 录

附录 1： 数据库性能初步测试代码

```
package dbtest;
/**
 * <p>Title: </p>
 * <p>Description: </p>
 * <p>Copyright: Copyright (c) 2003</p>
 * <p>Company: </p>
 * @author unascribed
 * @version 1.0
 */
public class dbtest {
  public dbtest() {
    super();
  }
  public static void main(String[] args) {
    dbtest dbtest1 = new dbtest();
    dbtest1.init_parameter();
    dbtest1.init_connection(3);
    dbtest1.insert_test(dbtest.con);
  }
  private String url="";
  private String Driver="";
  private String user="";
  private String password="";
  private String catalog="";
  private String schema="";
  private String tablename="";
  //sqlserver
  private String sqlUrl="jdbc:microsoft:sqlserver://localhost:1433;DatabaseName=test";
  private String sqlDriver="com.microsoft.jdbc.sqlserver.SQLServerDriver";
  private String sqluser="sa";
  private String sqlPassword="1";
  private String sqlCatalog="test";
  private String sqlSchemapattern="dbo";
  private String sqlTableNamePattern="test";
  //oracle
  private String oracleUrl="jdbc:oracle:thin:@127.0.0.1:1521:liyan";
  private String oracleDriver="oracle.jdbc.driver.OracleDriver";
  private String oracleUser="test";
  private String oraclePassword="1";
  private String oracleCatalog=null;
  private String oracleSchemapattern=this.oracleUser.toUpperCase();
  private String oracleTableNamePattern="test";
  //mysql
  private String mysqlUrl="jdbc:mysql://localhost/test?useUnicod=true&characterEncoding=GBK";
  private String mysqlDriver="org.gjt.mm.mysql.Driver";
  private String mysqlUser="administrator";
  private String mysqlPassword="1";
  private String mysqlCatalog="test";
  private String mysqlSchemapattern=this.mysqlUser;
  private String mysqlTableNamePattern="test";
  //Access
```

```
  private String AccessUrl="jdbc:odbc:mimi";
  private String AccessDriver="sun.jdbc.odbc.JdbcOdbcDriver";
  private String AccessUser="administrator";
  private String AccessPassword="1";
  private String AccessCatalog="test";
  private String AccessSchemapattern=this.AccessUser;
  private String AccessTableNamePattern="test";
  private java.util.Hashtable propTable=null;
  private static java.sql.Connection con=null;
  //init parameter
  private void init_parameter(){
     this.propTable=null;
     this.propTable=new java.util.Hashtable();
     java.util.Properties prop=new java.util.Properties();
     //sqlserver parameter
     prop.put("url",this.sqlUrl);
     prop.put("driver",this.sqlDriver);
     prop.put("user",this.sqluser);
     prop.put("password",this.sqlPassword);
     prop.put("catalog",this.sqlCatalog);
     prop.put("schema",this.sqlSchemapattern);
     prop.put("tablename",this.sqlTableNamePattern);
     this.propTable.put("sqlserver",prop.clone());
     //oracle parameter
     prop.put("url",this.oracleUrl);
     prop.put("driver",this.oracleDriver);
     prop.put("user",this.oracleUser);
     prop.put("password",this.oraclePassword);
     if(this.oracleCatalog!=null) rop.put("catalog",this.oracleCatalog.toUpperCase());
     prop.put("schema",this.oracleSchemapattern.toUpperCase());
     prop.put("tablename",this.oracleTableNamePattern.toUpperCase());
     this.propTable.put("oracle",prop.clone());
     //mysql parameter
     prop.put("url",this.mysqlUrl);
     prop.put("driver",this.mysqlDriver);
     prop.put("user",this.mysqlUser);
     prop.put("password",this.mysqlPassword);
     prop.put("catalog",this.mysqlCatalog);
     prop.put("schema",this.mysqlSchemapattern);
     prop.put("tablename",this.mysqlTableNamePattern);
     this.propTable.put("mysql",prop.clone());
     //access parameter
     prop.put("url",this.AccessUrl);
     prop.put("driver",this.AccessDriver);
     prop.put("user",this.AccessUser);
     prop.put("password",this.AccessPassword);
     prop.put("catalog",this.AccessCatalog);
     prop.put("schema",this.AccessSchemapattern);
     prop.put("tablename",this.AccessTableNamePattern);
     this.propTable.put("access",prop.clone());
     prop=null;
  }

  private void init_connection(int db_type){
     java.util.Properties prop=null;
     switch(db_type){
        case 1 :     prop= (java.util.Properties) this.propTable.get("sqlserver");break;
```

```
        case 2 :    prop= (java.util.Properties) this.propTable.get("oracle");break;
        case 3 :    prop= (java.util.Properties) this.propTable.get("mysql");break;
        case 4 :    prop= (java.util.Properties) this.propTable.get("access");break;
      }
      this.url=prop.get("url").toString();
      this.Driver=prop.get("driver").toString();
      this.user=prop.get("user").toString();
      this.password=prop.get("password").toString();
      this.catalog=prop.get("catalog").toString();
      this.schema=prop.get("schema").toString();
      this.tablename=prop.get("tablename").toString();
      try{
        Class.forName(this.Driver);
        this.con=java.sql.DriverManager.getConnection(this.url,this.user,this.password);
      }catch(Exception e){
        System.out.println(e.toString());
      }
    }

    public void insert_test(java.sql.Connection con){
      try{
        java.sql.Statement stmt=con.createStatement();
        stmt.executeUpdate("truncate table test");
        stmt.executeUpdate("commit");
        java.sql.DatabaseMetaData dmeta=con.getMetaData();
        String[] tablestype={"table".toUpperCase()};
        java.sql.ResultSet rs= null;
        String str1="insert into test (";
        String strend=") ";
        String str2="values (";
        String sql=null;
        rs=dmeta.getColumns(this.catalog,this.schema,this.tablename,null);
        String media="";
        while(rs.next()){
          media=rs.getString(4);
          str1=str1+media+",";
        }
        str1=str1.substring(0,str1.length()-1);
        for(int i=0;i<1000;i++){
          for(int i1=i;i1<i+20;i1++){
            str2=str2+"'"+"test"+i1+"',";
          }
          sql="";
          str2=str2.substring(0,str2.length()-1);
          sql=str1+strend+str2+strend;
          str2="values (";
          stmt.addBatch(sql);
        }
        long start=System.currentTimeMillis();
        stmt.executeBatch();
        long end=System.currentTimeMillis();
        System.out.print(con.getMetaData().getDatabaseProductName()+"used"+(end-start)+"ms!");
      }catch(Exception e){
        System.out.println(e.toString());
      }
    }
}
```

附录 2：数据库连接池代码

```
package dbmanagement;
/**
 * <p>Title: </p>
 * <p>Description: </p>
 * <p>Copyright: Copyright (c) 2003</p>
 * <p>Company: </p>
 * @author unascribed
 * @version 1.0
 */
public class dbManager {
  //oracle
  private String oracleUrl="jdbc:oracle:thin:@127.0.0.1:1521:liyan";
  private String oracleDriver="oracle.jdbc.driver.OracleDriver";
  private String oracleUser="test";
  private String oraclePassword="1";
  private String oracleCatalog=null;
  private String oracleSchemapattern=this.oracleUser.toUpperCase();
  private String oracleTableNamePattern="test";
  //mysql
  private String mysqlUrl="jdbc:mysql://localhost/test?useUnicod=true&characterEncoding=GBK";
  private String mysqlDriver="org.gjt.mm.mysql.Driver";
  private String mysqlUser="administrator";
  private String mysqlPassword="1";
  private String mysqlCatalog="test";
  private String mysqlSchemapattern=this.mysqlUser;
  private String mysqlTableNamePattern="test";
  //Access
  private String AccessUrl="jdbc:odbc:mimi";
  private String AccessDriver="sun.jdbc.odbc.JdbcOdbcDriver";
  private String AccessUser="administrator";
  private String AccessPassword="1";
  private String AccessCatalog="test";
  private String AccessSchemapattern=this.AccessUser;
  private String AccessTableNamePattern="test";

  private dbManager() {
  }
  private static dbManager mydbManager=null;
  //dbManager getInstance
  public static dbManager getInstance() throws Exception{
    synchronized("dbManager"){
      try{
        if(dbManager.mydbManager==null){
          dbManager.mydbManager=new dbManager();
          mydbManager.reload_Prop();
        }
        return dbManager.mydbManager;
      }catch(Exception e){
        throw new Exception("dbmanagement.dbManager.getInstance Exception :"+e.toString());
      }
    }
  }
  private static javax.naming.Context ctxt=null;
  private int poolsize=10;
  private String drivername=null;
```

```
private String url=null;
private String username=null;
private String password=null;
private int activeConnectionCount=0;
private java.util.Stack myConnectionPool=null;
// reload Prop
public void reload_Prop() throws Exception{
   try{
      this.url=this.mysqlUrl;
      this.drivername=this.mysqlDriver;
      this.password=this.mysqlPassword;
      this.username=this.mysqlUser;
      if(this.myConnectionPool==null){
       System.out.println("enter myconnection pool");
       this.myConnectionPool=new java.util.Stack();
       System.out.println("end myconnection pool");
       Class.forName(this.drivername);
       for(int i=0;i<this.poolsize;i++){
          java.sql.Connection con=java.sql.DriverManager.getConnection(url,username,password);
          this.myConnectionPool.push(con);
       }
      }
   }catch(Exception e){
       throw new Exception("read the setup file error : "+e.toString());
   }
}

public void reload_Prop(java.util.Properties prop) throws Exception{
   try{
      this.poolsize=Integer.parseInt(prop.getProperty("poolsize").trim());
      this.url=prop.getProperty("url");
      this.drivername=prop.getProperty("drivername");
      this.username=prop.getProperty("username");
      this.password=prop.getProperty("password");
      if(this.myConnectionPool==null){
       this.myConnectionPool=new java.util.Stack();
       Class.forName(this.drivername);
       for(int i=0;i<this.poolsize;i++){
          java.sql.Connection con=java.sql.DriverManager.getConnection(url,username,password);
          this.myConnectionPool.push(con);
       }
      }
   }catch(Exception e){
      this.myConnectionPool=null;
      throw new Exception("read the setup file error : "+e.toString());
   }
}

//getCon
public java.sql.Connection getCon() throws Exception{
   System.out.println(this.activeConnectionCount);
   try{
      if(this.myConnectionPool==null){
         throw new Exception("connection pool is empty ");
      }
      if(this.activeConnectionCount>=this.poolsize){
         Class.forName(this.drivername);
```

```
            java.sql.Connection con=java.sql.DriverManager.getConnection(url,username,password);
            System.out.println("new con");
            return con;
          }else{
            this.activeConnectionCount++;
            return (java.sql.Connection) this.myConnectionPool.pop();
          }
        }catch(Exception e){
          throw new Exception ("获得数据库连接错误 ： "+e.toString());
        }
      }

      //releaseCon
      public void releaseCon(java.sql.Connection con) throws Exception{
        synchronized(this){
          try{
            if(this.activeConnectionCount>0){
              this.myConnectionPool.push(con);
              this.activeConnectionCount--;
              System.out.println("release con");
            }
          }catch(Exception e){
            throw new Exception("释放数据库连接错误！ : "+e.toString());
          }
        }
      }
    }
```

致 谢

研究生生活即将结束，在此论文完成之际，首先向我的导师靳其兵教授致以崇高的敬意和衷心的感谢！

本文是在靳老师的悉心指导和关怀下完成的。在三年的学习期间，靳老师对作者给予了多方面的指导、关心和帮助。靳老师渊博的学识、严谨的治学态度、孜孜不倦的敬业精神、为人师表的高尚风范、宽广坦荡的胸怀、极大地激励着我不断求知和探索，并将对我今后的学习和工作产生深远的影响。

感谢北京化工大学信息学院、北京化工大学科技处，为我们提供了比较良好的实验环境。同时，感谢在学习和生活中给予我很大帮助的同门师兄弟赵大力、任勇和李晓波同学。

最后，向所有支持和帮助过我的老师和同学们表示感谢！

研究成果及发表的学术论文

发表及已接受的论文

1. 林乐杰, 靳其兵. 基于 JAVA 的动态类更新算法的研究. *计算机应用研究*, 已接受.

作者和导师简介

1．作者简介

林乐杰，男，1975 年 9 月出生，汉族，山东文登人，北京化工大学信息科学与技术学院硕士研究生，专业：计算机应用技术，研究方向：网络数据库应用技术。

2．导师简介

靳其兵，男，1971 年 10 月出生，汉族，湖北宜昌人，教授，博士，北京化工大学信息学院副院长；自动化研究所副所长。北京市科技新星称号获得者，先后主持科研项目 10 余项，其中省部级项目 4 项，第一作者发表科技论文 20 多篇。研究领域：1.先进控制及工业中的应用 2.智能仪器的研究及制作 3.企业信息化、管控一体化（包括数据库管理系统的开发）。