一个 PROLOG 数据库系统

〔英〕 DeYi Li 著

林耕 吕颖 译

林耕 校

科 海 培 训 中 心

一九八八年 四月

一个 PROLOG 数据库系统

〔英〕 DeYi Li 著

林耕 吕颖 译

林耕 校

科 海 培 训 中 心

一九八八年 四月

前　言

近年来，逻辑程序设计语言PROLOG正受到越来越多的重视，PROLOG语言已经进入许多领域。由Deyi Li博士著的本书，描述了一个用PROLOG语言实现的智能数据库管理系统ILEX，具体地向我们展示了一个PROLOG应用的成功范例。本书从数据库（管理系统、查询语言）和PROLOG应用两个方向进行了讨论，使得两者紧密地结合起来。书中附有许多例子，可供读者参考。

本书第一章对数据库查询语言做了一般性的概括和介绍；第二、三章讨论了关系数据库（查询语言）与逻辑的关系，并基于一种受限的逻辑，为多种查询语言提供了一个通用界面；第四到第六章介绍了ILEX系统提供的三种典型的查询语言；第七章介绍了ILEX系统中一些较高级的功能；第八章讨论了ILEX中特有的全局优化；第九、第十章讨论了ILEX与数据库机器间的连接和一些今后的研究方向。从第三章到第八章，都带有具体的PROLOG实现程序。

在翻译过程中，我们纠正了原书中的个别错误。

由于水平有限，译文难免有不妥之处，望读者指正为盼。

译　者

1988年3月

目　录

第一章　引言

1.1　人机系统中人的一面 ……（1）
1.1.1　背景 ……（1）
1.1.2　要求 ……（2）
1.2　关系查询语言的现状 ……（2）
1.2.1　关系查询语言分类 ……（2）
1.2.2　现有的查询语言 ……（3）
1.3　查询语言面临的问题 ……（4）
1.3.1　多样性 ……（4）
1.3.2　灵活性 ……（4）
1.4　ILEX系统的目标和好处 ……（5）
1.5　系统设计轮廓 ……（6）

第二章　关系数据库和逻辑

2.1　关系数据库 ……（8）
2.1.1　关系模型中的术语 ……（8）
2.1.2　例子数据库 ……（10）
2.2　逻辑程序设计 ……（12）
2.2.1　一阶谓词演算 ……（12）
2.2.2　句式 ……（13）
2.2.3　Horn子句 ……（14）
2.3　PROLOG做为一种逻辑程序设计语言 ……（15）
2.3.1　PROLOG简单回顾 ……（15）
2.3.2　PROLOG的消解证明过程 ……（16）
2.3.3　PROLOG中的非逻辑机制 ……（17）
2.4　逻辑和关系数据库间的关系 ……（18）

第三章　查询语言的一个规范逻辑形式

3.1　关于一致性研究的一般评论 ……（20）
3.1.1　相容性和界面 ……（20）
3.1.2　查询处理的考虑 ……（21）
3.2　查询公式 ……（22）
3.3　规范式的形式定义 ……（24）

3.4 规范式的关系处理能力……（25）
3.5 一个入门性的例子……（29）

第四章 一种类图语言PL

4.1 QBE做为ILEX系统的一个候选……（31）
4.2 PL 语言……（31）
4.2.1 检索操作……（32）
4.2.2 内部函数……（35）
4.2.3 存储操作……（36）
4.3 PL的语法……（38）
4.3.1 查询表索引……（38）
4.3.2 查询元组……（39）
4.4 PL 的屏幕编辑……（40）
4.4.1 用户—系统交互……（40）
4.4.2 转换（conversion）……（41）
4.4.3 分解（digestion）……（42）
4.4.4 识别（recognition）……（44）

第五章 一个类英语语言 EL

5.1 SQL做为ILEX 系统的一个候选……（46）
5.2 EL 语言……（46）
5.2.1 检索操作……（46）
5.2.2 内部函数……（50）
5.2.3 存储操作……（51）
5.3 EL 的语法……（52）
5.4 EL 的语法分析器……（54）
5.4.1 把关键字做为运算符……（54）
5.4.2 变量的创建……（57）
5.4.3 条件的处理……（57）

第六章 一个类数学语言ML

6.1 关系代数 运算……（62）
6.2 ML 语言……（62）
6.2.1 ML 的语法……（62）
6.2.2 简单检索……（63）
6.2.3 选择—投影—等值连接表 达式……（63）
6.2.4 复杂查询……（64）

6.3 ML 的分析器……（65）
6.3.1 代数操作的策略……（65）
6.3.2 规则系统……（66）
6.3.3 任意代数查询的统一……（68）
6.4 ML和EL 间的相容性……（72）
6.4.1 五个标准模式……（72）
6.4.2 ILEX 的共享库……（73）

第七章 系统ILEX中的否定和聚合

7.1 把否定做为不可证明的处理……（75）
7.1.1 开放世界和封闭世界的假定……（75）
7.1.2 否定作为失败……（76）
7.2 二阶谓词 "all"、"group"和 "update"……（76）
7.2.1 谓词“all”……（76）
7.2.2 谓词 "group"……（77）
7.2.3 谓词 "update"……（78）
7.3 ILEX 中的聚合函数……（78）
7.3.1 形式定义……（79）
7.3.2 聚合函数的实现……（79）
7.3.3 二阶谓词的组合……（81）
7.4 规范逻辑式的可扩充性……（83）

第八章 基于规范逻辑式的查询优化

8.1 ILEX 中的查询优化策略……（84）
8.2 一个代价度量公式……（84）
8.2.1 代价度量的重要性……（84）
8.2.2 代价公式……（85）
8.3 动态优化……（86）
8.3.1 假定……（86）
8.3.2 合取中目标的排序……（87）
8.3.3 优化方法的实现……（88）
8.4 提高效率的潜在能力……（89）

第九章 基于 MEMEX 的 ILEX 系统结构

9.1 PROLOG 和 RDBM 间的界面……（92）
9.2 内容地址机制……（94）
9.3 一个基于Zipf定律的压缩技术……（94）

9.4 顺序联想系统MEMEX ……（96）
9.5 基于MEMEX的特殊硬件 ……（97）

第十章 结论和以后的工作

10.1 一般结论 ……（98）
10.2 关于进一步研究的建议 ……（100）

参考书目 ……（101）

附录1 内部谓词和运算符的优先级
附录2 规范式的关系实现
附录3 查询例子
索引

第一章 引 言

1.1 人机系统中人的一面

1.1.1 背景

在计算机时代刚开始时，只有专业人员才使用计算机系统；在系统设计时，经常不考虑用户。今天，先进的科技使得许多人都成为计算机用户，并且有了各种各样的计算机应用。这就产生了许多有关计算机——用户界面的问题。今天，无视用户的存在就会引起灾难，这已经是不言而喻的了。造价、速度、可靠性和许多其它“效率”概念在测试机器时是比较容易掌握的，而人的因素则不然。但人的因素在系统中是一个决定性的部分，换句话说，如果系统设计得不符合用户的需要和能力，系统资源的使用效率就会降低。当计算能力的代价下降时，所需的支持这个计算能力的人工代价就升高了。由于这个原因，问题就变得更为严重了。这个事实产生了一个新的研究方向，它研究计算机系统中人的因素。

最近一些年里，人们开展了若干用户研究。例如，测试了系统特征和用户行为之间的关系；利用实验心理学的技术，许多工业上和学术上的实验者开始研究数据库查询语言。一个查询语言是一个形式化定义的，有着有限功能的语言工具。它的表达式与人们想向数据库所做的提问相对应。对于每个提问或叫查询，都有一个相应的答案。在许多情况下，查询语言是面向非程序员而设计的。由于这样的用户缺少计算机知识，所以一个成功的语言应该对于他们，容易学、容易用和容易记。对一个查询语言人的因素的测试，可以决定它是否满足这些标准。最近，不仅仅是对这些各种各样查询语言的研究，而且对于这些研究的简单评述也已经以文章和书的形式出现了。人的因素的测试方向和方法也已经提出。

P.Reisner在她的评述文章[1]中提到了许多数据库查询语言的经验研究和一些关于两种或多种语言间易用性的比较。这些工作被用来衡量一个查询语言的易用性，并且向设计者提供反馈信息来改进该语言。正如她所总结的，人们用QBE语言写查询要比用SQL写查询快二到三倍，但人们遇到QBE中合格性的麻烦，而在SQL中，拼写错误和结束出错则是经常的。Greenblatt和Waxman[6]在比较了QBE，SEQUEL[7]和关数代数哪个更容易学习之后，指出QBE更容易掌握。Reisner[9]等人对SEQUEL和SQUARE[10]做了研究，得出结论，SEQUEL相对地易于学习。Lochovsky[11]研究了许多数据模型和它们相应的数据操作语言。在经过对SEQUEL和TABLET[12]进行比较之后，C.Welty和D.W.Stemple[13]发现人们在书写比较复杂的查询时，用过程化的语言比用非过程化语言的正确率更高。

做这项工作的动机是多种多样的。这些研究的共同点是他们把行为科学中的技术运用到对查询语言问题的研究中，即他们试图用那些能够进行数量度量的概念来定义模糊概念“易用性”，并且他们研制了过程来度量它，研制的度量技术是经验性的。

不幸的是，用户研究是一个相当年轻的学科，并且诸如“对用户友好性”和“自然性”等都是经常被使用，但却很少详细定义的概念。并且似乎对这些概念的含义也没有一致的意见。上述的这些比较只是近似的，因为不同的实验在许多方面都不同，如，训练，技术，专业人员，测试问题，判分的严格性等等。所以结论只能基于特定的情况。并且必须注意，不

要过分推广某个发现。似乎现在还没有一个很清楚的方法来帮助设计查询语言，尽管许多实验都提供了一些设计反馈信息。然而，人的因素的研究的重要性正在得到日益增多的承认。如果系统中人的方面没有有效地工作，则系统的技术部分也不会有效地工作。系统的用户界面必须与用户的技能和任务要求相适应。唯一合理的态度是对用户性能和系统性能给予同样的重视。

1.1.2 要求

请注意用户的倾向和应用。我们要注意系统中人的部分和技术部分的矛盾，让我们引入**使用的灵活性**和**操作复杂性**的概念。使用的灵活性指的是提交任务给系统的用户的灵活性，它是由对于一个特定的用户任务，系统能够支持的处理方案的数目来决定的。操作复杂性指的是那些为完成任务，用户必须执行的操作活动的复杂性。人们可以区分客观操作复杂性和主观操作复杂性。前者是由系统给定的实际的复杂性，后者反映了用户的个人感觉。

把这两个概念转换到人——机界面的设计中就得出如下结果，从人的观点来看，并且有一个大的使用灵活性；而从技术的观点来看，在一个可接受的代价和性能的情况下，应该达到一个低的操作复杂性。在这些考虑下，一个最优的人—机界面设计可以被看成一个在最大使用灵活性与最小操作复杂性之间的优化问题[14]。

1.2 关系查询语言的现状

1.2.1 关系查询语言的分类

查询语言可以在许多方面加以区别，如语法形式，过程性，或隐含的数据模型等。由于查询语言的多样性，不同的分类是有用的。

在语言的外层，它与用户最接近，许多可用的交互模型设计用来满足各种用户，他们需要与数据库交互作用，这些模型包括：

i） 宿主语言，它们具有独立的语法形式，它们嵌入那些广泛应用的程序设计语言，如COBOL，PL/1，FORTRAN等中。嵌入方式可以是简单的子过程调用，也可以向宿主语言中加入新的语法，这需要一个预处理程序或者修改翻译器。SQL就是这样语言的例子，它嵌入宿主程序设计语言PL/1或COBOL中。

ii） 自包含语言，它们提供所有用于执行可用功能的机制。

iii） 计算机导引的功能规格说明。通过菜单选择，填空、参数性的请求。语法学习可以被避免，但可选择的范围是很小的。

iv） 自然语言界面。它能消除语法学习的必要。但要包含一个很长的说明性对话。它会使人以为机器智能是无限的，会阻碍用户的有益的思考。

不论最外层是什么样的交互模型，在概念层上表达查询的形式是查询语言最根本的性质。就我们所关心的关系模型而言，查询语言本质上分为两大类：

i） 代数语言。其中的查询通过对关系施以专门的运算符表示出来。

ii） 演算语言。其中的查询描述一个预期的元组集合，办法是指明元组必须满足的谓词。基于演算的语言又可以进一步分为两类：

a） 元组关系演算，其中的最基本对象是元组。

b） 域关系演算，其中的最基本对象是具有某种属性的域的元素。

这三种不同类型的查询语言可以当作评价现有的关系查询系统的基准。每一种在表达能力上都等价于其它两种，并且这个能力是由 E.F.Codd[15，16] 提出来表示任何基于关系模型的合理的查询语言所需的最小能力。实际的查询语言通常除了这些抽象语言的功能外，还提供了一些别的功能。

1.2.2 现有的查询语言

关系数据子语言有许多种实现，下面列出一部分：

语言名	语言类型	方法	软件	硬件	研制者
ALPHA	命令	元组演算	—	IBM	IBM(Codd)
APL	命令	演算	—	IBM	General Motors
APPLE	命令	演算	—	—	西北大学
ASTRID	命令	代数	EORTRAN	Honeywell	阿伯丁大学
CUPID	格式	元组演算	QUEL	UNIX	加州大学
INQUIRE	命令	演算	—	IBM	Infodata
ISBL	命令	代数	PRTV	IBM	Peterlee
LINUS	命令	演算	MROS	Honeywell	Honeywell
QUEL	命令	元组演算	INGRES	UNIX	加州大学
QBE	格式	域演算	—	IBM	IBM(Zloof)
REGIS	命令	演算	—	IBM	General Motbrs
RENDENZ_					
VOUS	自然语言	演算	ALPHA	IBM	IBM(Codd)
SQUARE	命令	元组演算	系统R	IBM	IBM
SEQUEL	命令	演算	系统R	IBM	IBM
SQL	命令	演算	系统R	IBM	IBM
TABEL	命令	代数	—	—	麻省大学
TAMALAN	命令	演算	—	CDC	CDC(Belg)
THESEUS	令命	演算	—	—	罗彻斯特大学

图 1.1 关系查询语言

这份表需要一点解释，对应于外层，语言类型分为：命令型，即用受限语法的命令驱动；格式型，即用固定格式来表示查询的对话式驱动；自然语言型，即自由的自然语言。方法表示了在概念层上，访问格式化数据的方法。

在这些语言中，我们有特殊兴趣的是下列系统和它们的语言。系统 ILEX 是基于它们建立的：

a） 系统 PRTY 和语言 ISBL：ISBL（Information System Based Language，信息系统基于的语言）是一个“纯粹的”关系代数语言。它是在 Peterlee 的 IBM 英国科学中心研制的。它用于实验性的 Peterlee Relational Test Vehicle（PRTV）系统。ISBL 没有聚合运算符，没有元组的插入、删除、修改机制，然而在 PRTV 系统的环境下，它可以利用 PL/1 程序来完成关系的处理。ISBL 和 PRTV 系统的讨论见 Todd [17]。

b） 系统 INGRES 和语言 QUEL 及 CUPID：

系统 INGRES（Interactive Graphics and Retrieval System，交互式图形和检索系统）是加州大学伯克利分校研制的，在 UNIX 操作系统上实现。它提供查询语言 QUEL（QUEry Language），一个元组关系查询语言，嵌于 C 程序中。QUEL 与一阶逻辑之间有一个简单的映射关系。CUPID 是一个图形语言，它在 QUEL 上实现。利用 CUPID，用户可以组成任意复杂的查询，办法是只需用光笔处理几个标准字符。INGRES 的设计实现细节见[18]。

c） 系统 R 和语言 SQUARE，SEQUEL，SQL：

系统 R 是一个在标准 IBM VM/370 操作系统上研制的实验性的数据库管理系统。它是由在 san Jose 的 IBM 研究实验室研制的。三种语言 SQUARE（Specifying Queries As Relationl Expression）、SEQUEL（Structured English QUery-Language）和 SQL（Structured Qurey Language）是在这个项目中应用评估过程的结果。人们可以按 SQUARE—SEQUEL—SQL 的顺序来称呼它们，以表明过去十年里，系统 R 的查询语言的发展历史。在 M、W、Blasgaen 等人[20，21]的文章中，对系统 R 做了结构上的总述。

d） 系统 SBA 和语言 QBE：

QBE（Query—By—Example）是一个域关系演算语言。由 Zloof[2，3]创建的 QBE 是面向最外层用户的，易于学习和使用。它通过对显示屏幕上的图形符号操作来完成查询。系统 SBA（System for Business Automation）是 QBE 的一个扩充，它在[20，21]中有讨论。

1.3 查询语言面临的问题

1.3.1 多样化

在比较了多种查询语言之后，我们得出结论：甚至最好的查询语言也不会是可普遍接受的。不同类型的用户有不同类型的需求和技能。有些用户喜欢串语言，如 QUEL 或 SQL 而别的用户则喜欢面向屏幕的二维数据子语言，如 QBE。一种语言可能适合于这个用户，但不适合于另一个。似乎不可能有一个查询语言能适应所有类型的用户的口味。应该有不同的语言来满足不同的需要。几乎可以绝对肯定，几个不同的查询语言，如 ISBL，QBE 和 SQL 在将来的许多年里可以共存。语言的选择是根据预期的用户环境而定的。因此，正确的说法应该是在什么样的环境下，某种语言优于其它语言，而不是哪种语言最好。

这件事实也许会在设计者心中引起混乱，并且会产生一些只在其应用范围内优化过的系统。所以需要有一种以一致的方式解决各种问题的方法，这种方法可以适用于各种查询语言。

1.3.2 灵活性

正如前面提到的，已经有了许多种关系数据子语言的实现。然而，由于语言类型的多样性，这些实现中的每一种都是面向某一特定的语言。这些工作是独立的，基本原理也是不同的，它们通常有自己的存取方法，有自己的规定来保证完整性控制等等。于是，如果我们希望系统能支持两种不同的语言，则会有大量软件功能的重复，于是会引起开发代价、维护代

价的大量增加。这显然会使得系统非常不如意。

并且，每种语言总是有自己的优缺点，总有改进的余地。而对隐含的数据库管理系统（DBMS）软件或查询语言本身做修改都将是很昂贵的，也是很费时间的。

基于上述情况，我们需要做一致性的研究。即对于这些语言的处理，尽可能地普遍化。这个想法十分诱人，因为一致性研究会使程序的可移植性增强，易于从一个系统移到另一个系统。

1.4 ILEX 系统的目标和处

为了解决具体的查询语言所面临的问题，我们实现了一个叫做 ILEX 的智能关系数据库系统。它为用户提供了至少两种查询语言以备选择，以便用户能够使系统行为更好地适应自己的需要。

在这项研究中，我们遵循着先发现普通规则，然后翻译到具体设计的原则。这个原则是我们工作的基础。在工程中，要尽全力保持理论与实践的平衡。

为了把用户的需求也当作系统设计的一部分，我们需要在一个通用数据库界面上提供一种高层次的查询语言的形式化表示。尽管要求这个通用界面具有所有查询语言的特征是荒谬的，但我们仍然打算表明，一个建立在逻辑程序设计和关系基础上的通用界面是可实现的，并且能与不同的关系查询语言通讯。它与那些单个的查询语言相比，确实有很多优点。

下面我们更详细地列出 ILEX 的目标：

i） 在同一查询系统中支持几种不同风格的查询语言。这个应用领域过去很少受到重视。

ii） 任何语言扩充，不管提供什么功能，都应该从用户的角度，而不是从系统的角度，加以设计。换句话说，设计应该是自外向内（outside in）进行的。

iii） 在任何可以利用现有语言特征、功能的地方，都不准备引入新的语言设计。这就是说，要进行的研究应该是形成一个现在查询语言用户使用的，已稳固建立的（查询语言）模式的自然扩充。

iv） 研究本身不应依赖于任何特定的查询语言。应该允许许多关系查询语言映射到通用界面中，但通用界面应不限于，或不倾向于任何特定的语言。

v） ILEX 系统的一个主要目标是以一种统一的，一致的方式来包括关系模型的三种类型——元组演算，域演算和关系代数。在系统中，所支持的这三类查询语言应该被合并为一个统一的形式，而不是三个分离的集合。

vi） 系统 ILEX 应该从用户角度看来，是一个封闭的系统，即，用户通常不需要知道系统提供的一切查询语言。但同时，也应该允许用户交替地使用各种查询语言，因为某个查询也许极其适合于使用某一查询语言。

vii） 系统应该有效地实现。由于使用各个独立的查询语言而引起的软件冗余应尽可能地减少。例如，同样的内部函数应该可以被所有查询语言所使用。事实上，ILEX 系统自身应该被看成是一个它所支持的所有查询语言的数据库。同时，适用于一切语言的高效存取路径应该被表示出来。

viii） 可以在不改变用户界面的情况下，修改基础软件，如数据组织，存取方法（优化）等。系统支持的各查询语言，应该是系统中可以被清楚分开的组成部分，并可以与系统其它部分无关地加以修改。

ix） 我们希望 ILEX 是一个用来研究查询语言的人类行为评价的良好版本，因为它对不

同的查询语言提供了一个通用界面。

x） 由于数据库机器受到了极大的重视，我们希望系统 ILEX 能够使用联想式硬件。

很多人认为，在九十年代，知识信息处理将会成为计算机应用的一个主要领域，而在这个领域里，问题求解和逻辑界面将是主要问题[22]、根据这个观点，今后工作的基础将是把计算机科学三个研究领域的概念合在一起。这三个领域是：人工智能（AI）和逻辑程序设计，数据库管理系统（DBMS），联想式处理。对于人工智能和逻辑程序设计，我们将利用演绎推理的问题——解答系统中的知识表达和一阶谓词演算的概念；对于DBMS，我们利用关系数据库中的概念；最后，利用联想式处理，我们可以避免冯·诺依曼机器的瓶颈问题。我们相信，数据库机器将是体系结构研究的必然结果。体系结构上的进展把数据处理的能力更紧密地分配到数据存贮器上。这些想法的结合，就形成了智能关系数据库，它会在系统——用户界面上产生许多新的、强有力的机制，并使得实现的代价降低，应用的灵活性增加。

1.5 系统设计轮廓

上述考虑使得我们决定选用 PROLOG 语言来实现这个数据库系统。PROLOG 是目前最好的、最广泛应用的逻辑程序设计语言。这个系统包括三个典型的高层次查询语言。现在，我们已经完成了一个有用的、高效的实现版本。这表明，即使在微机环境下，一个包括许多查询语言的通用界面也是可以实现的。

系统 ILEX 分三个主要阶段来处理一个查询：

查询 $\xrightarrow{\text{理解}}$ 逻辑 $\xrightarrow{\text{优化}}$ PROLOG $\xrightarrow{\text{执行}}$ 答案

（1） （2） （3）

图 1.2 查询的三步处理

图 1.2 给出了 ILEX 中知识信息处理的一个非常普遍的观点，即：

1） 这个查询意味着什么？——对要解决的问题的一个启发式理解

2） 我如何回答它？——优化，准备用有效的方法解决它。

3）答案是什么？——执行。这点在将来有希望与一个回溯——终点结构相连。这个回溯——终点结构可以是一个联想式检索系统。

一个用某具体查询语言写的查询的处理如图 1.3 所示：

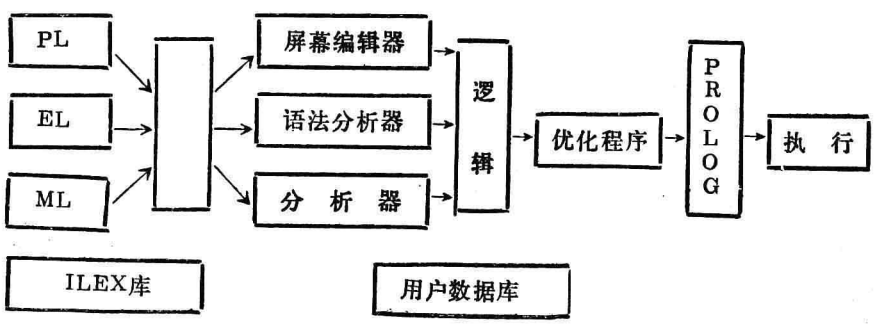

图 1.3 ILEX 的模块结构

在第二章中，我们将介绍后面各章所需的术语和概念。通过在第三章引入规范逻辑表达式，我们可以为任何查询建立一个通用的知识表示。第四、第五、第六章分别讲的是三种逻

然相异的查询语言：PL（一个类图语言），它基于QBE；EL（一个类英语语言），它基于SQL；以及ML（一个类数学语言），它基于关系代数，如何被翻译成规范逻辑表达式。ILEX系统中的否定和聚合功能的实现技术将在第七章描述。第八章提出了一个有效的优化策略。关于ILEX系统和数据库机器MEMEX〔25〕间的连接的一些考虑在第九章给出，这些考虑还有待于硬件实现。在第十章中，我们对我们的工作做一个总结，并展望一下将来的工作。

第二章　关系数据库和逻辑

为了为后面的章节铺平道路，本章给出了一些关于关系数据库及逻辑程序设计的基本概念。本章的目的是表明逻辑如何能够为研究关系数据库问题提供一个形式化的支持手段，以及在某些情况下，逻辑如何能够更进一步，首先帮助理解问题，然后帮助解决问题。此外，本章还将表明，PROLOG语言，由于它的逻辑特点，非常适用于关系数据库模型。

2.1　关系数据库

今天，数据处理面临的最关键的问题之一，是程序开发和维护费用的迅速增加。文件使用中，两个最基本的问题是：

i）　数据独立问题。

在非数据库系统中，应用是与数据相关的，我们没有办法改变存贮（数据的物理记录方式）或存取策略而不影响应用。

ii）　数据冗余问题。

每个应用都有其自己的私有文件。这件事实经常导致存贮数据的大量冗余，或造成存贮空间的浪费和数据不一致的风险。

数据库管理系统解决了这些问题，它通过对数据库进行中央控制，使得用户可以写出与数据独立的程序，并能保持数据的一致性。有三种著名的数据模型：关系模型，层次模型和网状模型。在关系模型中，数据被假定以表格形式存贮；在层次模型中，数据被假定以树结构形式存贮；在网状模型中，数据被假定以一般的图的形式存贮。通过对它们的比较，人们知道关系模型是基于稳固的数学学科的，是对数据的逻辑描述和操作做的具有重要意义的研究。简略地说，它把逻辑数据库看做是标准化关系的与时间相关的集合。一个关系数据库是由一些功能很强的运算符来操作，它们分解、合并关系中的各列。这些运算符中不包含位置、指针或存取路径等概念。这样一个高层次观点，在网状及层次结构中是不会立即实现的。然而，由于关系模型概念上的简捷性和通用性，就使得它更难于有效地实现。直至1979年，还没有商业化的全关系模型的DBMS，尽管IBM的QBE倾向于基于关系模型。实验性的关系系统R也许是已实现的最先进的关系数据库。

2.1.1　关系模型中的术语

关系模型带来了它自己的术语，这些术语更倾向于数学化。因此，关系模型有一个很好的理论基础[26]。为了方便起见，我们简略地总结如下：

〔定义〕

关系。设D_1，D_2…，D_n是一组给定的集合，R是一个在这n个集合上的关系，如果它是有序n元组$<d_1, d_2, \cdots, d_n>$的集合，其中d_1属于D_1，d_2属于D_2，…，d_n属于Dn。通常用表格的形式来表示关系。在表格中，每一列的头上标上属性，每一行是一个元组。表格和关系经常交替着使用。

关系模式。这是一个属性的集合，这些属性是一个关系中各列的名字。一个关系模式是不随时间改变的，而与之相应的关系则经常改变。

元数。一个关系的元数是其中列的个数。元数为1的关系称为一元关系，元数为2的关系称为二元关系，…，元数为n的关系称为n元关系。

属性。一个关系的属性是关系中列的名字。属性的值是列的内容。

域。域是关系属性可以取值的集合，即构成表格中列的元素的集合。分清域与列的差别是非常重要的。列表示在关系中域的应用。

元组。我们可以把关系看成是元组的集合。一个元组是表中的一行。一个关系中，每个元组都是唯一的。因为它是一个数学集合中的一个元素。

维数。关系R的维数，用|R|表示，是R中元组的个数。

完整性约束。数据库一致性是由完整性约束来保证的。完整性约束是数据库必须服从的断言。数据相关是完整性约束的一个特殊情况。

函数相关（FD）。函数相关X→Y是一个关于关系模式的断言。它指出，任何满FD的关系中的两个元组t_1和t_2，若在属性X上相同，则在属性Y上也相同，即如果$t_1[X]=t_2[X]$，则$t_1[Y]=t_2[Y]$。

多值相关（MD）。一个多值相关X→→Y是指，如果t_1、t_2是一个关系的两个元组，满足X→→Y，则该关系中必然有第三个元组t_3，其属性Y的值与一个元组的Y值，比如说t_1的Y值相同，而其余的列值与另一个相同。换句话说，X→→Y意味着与某一X值相关的Y值必须与其它属性值无关。

关键字。在满足完整性约束的情况下，关系中可以做为唯一标识的属性的最小集合就叫做一个候选关键字。一个关系可以有多个候选关键字。

标准化。标准化理论把相关类型与关系模式的预期性质联系起来。通过标准化，模式中的各关系中就没有上述相关类型的数据冗余。

数据库。一个数据库是一个关系的集合。这些关系的模式合起来形成数据库模式。

总之，一个关系数据库DB是一个集合，R={Ri}，即客观世界中具有某种属性D的事物的集合。每个关系Ri由一组属性Si={DjDj∈D}来刻划。Si称为Ri的模式，Ri由一组元组构成。每个元组是满足给定相关性的，由关系模式的属性到它们的域的一个映射。

图2.1表示了一个叫做'R'的关系，它由名为'AA'，'BB'，'CC'，'DD'的列组成。关系'R'中包含的每个术语都在图2.2中给出。

R			
AA	BB	CC	DD
a_1	b_1	c1	d1
a_1	b_2	c2	d2
a_1	b_1	c1	d2
a_1	b_2	c2	d1
a_2	b_3	c1	d1
a_2	b_3	c1	d2

图 2.1 一个关系(表格)

术　语	相应的名字	例　子
关　系	文件／表格	R
关系模式	表　头	<AA, BB, CC, DD>
属　性	列名／域类型	AA
属性值	隐含域	a1, a2
元　素	值	c2
元　数	度	4
元　组	行／实体／段／记录	<a1, b2, c2, d2>
维　数	—	6
FD	—	BB→CC
MVD	—	AA→→<BB, CC>

图 2.2 术　语

一个关系数据库由一组用户提交的任务来访问。用户提交任务是为了检索、删除、插入或修改任何数据子集。一般说来，检索是这些处理中最基本的部分。用户用查询的形式提出他们的请求，指出需要检索的数据和数据要满足的条件。查询处理的任务将指明哪组数据合适，访问这些数据的次序及对这些数据操作的类型。这个处理过程被不同的实现者分别称为查询翻译、寻找存取路径，或者叫优化等。

2.1.2 一个例子数据库

为了引入高层次的查询语言PL、EL和ML，我们对一个例子关系数据库进行查询。这个例子数据库是从C. J. Date[27]书中的例子扩充而来的。

这个数据库中，数据被组织成6个关系，（见图2.3）。它们是供应商（supplier），零件（part）、装运（shipment），库存（stock），部门（department）和雇员（employee）"supplier "表格对于每个供应商，包含一个供应号（sno），名字（sname），状态码（status）和所在城市名（city）；"parts "表格对每种零件包含一个零件号（pno），名字（pname），颜色（color），重量（weight）和该零件存放的地方（city）；"shipment）"表格对每一批装运，包含一个供应商号，一个零件号及装运的数量）qty）；"stock "表格对每一部门（dept），每一种零件，表明有多少库存；"department "表格，对每一部门描述部门所在的城市和部门经理；"employee "表格，包含雇员的名字（ename），年龄（age），工资（salary）和他所工作的部门（dept）。这些关系间的关系，以及它们的关键字由图2.4给出：

suPPlier

sno	sname	status	city
s1	smith	20	london
s2	jones	10	Paris
s3	blake	30	Paris
s4	clark	20	loudon
s5	adams	30	athens