

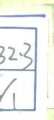
关系数据库

〔美〕 杨超植 著

刘动天 张龙祥 吴今培 译

刘尚威 校

電子工業出版社



TP311.132.3
YCZ.1/1

关系数据库

〔美〕杨超植 著

刘动天 张龙祥 吴今培 译

刘尚威 校



电子工业出版社

内 容 简 介

本书论述了关系数据库的基本原理,全书共九章,其内容包括:关系数据库中的基本数学概念,关系数据库模型,关系代数,关系谓词演算,函数依赖,多值依赖,联接依赖,规范化,查询语言与数据库管理系统。

本书系统性强,叙述严谨,既可作计算机类专业学生与研究生的教学用书,也值得从事信息管理、数据库开发及应用的科技人员参考。

JS281/04

RELATIONAL DATABASES

Chao-Chih Yang

关系数据库

[美] 杨超植 著

刘劭天 张龙祥 吴令培 译

刘海威 校

责任编辑:王惠民

*

电子工业出版社出版 (北京海淀区万寿路)

电子工业出版社发行 各地新华书店经销

中国科学技术情报研究所印刷厂印刷

*

开本: 787×1092毫米1/16 印张: 14.375 字数: 349 千字

1990年1月第一版

1990年1月第一次印刷

印数: 1—4,500册

定价: 5.30元

ISBN7-5053-0668-5/TP·109

前 言

本书是我以前在伯明翰阿拉巴马大学给研究生上数据库系统课时所写讲稿的基础上发展起来的。该教程包括涉及关系数据库理论的最先七章,原稿现已修改,进一步增加的内容包括查询语言ISBL、QUEL、QBE, PROLOG程序设计语言以及支持它们的数据库管理系统。本书将上述原理与实践一起融于一卷之中。

希望本书不仅作为大学高年级学生与一年级研究生的一本关于关系数据库的教科书,而且对于该领域内从实践与研究的人也是一本有用的参考书。

感谢IBM公司的莫谢 M. 佐洛夫 (Moshe M. Zloof) 博士允许我使用他的百货商店数据库作为全书的例子。还要感谢沃伦 T. 琼斯 (Warren T. Jones) 博士和凯文 赖利 (Kevin Reilly) 博士对我的鼓励。感谢苏珊 迪安 (Susan Dean) 博士对整个原稿做了技术校对,并提出了不少宝贵意见。

伯明翰阿拉巴马大学两个年级使用过前面七章的初稿,并提出了许多有益的建议。特别是,我的学生理查德 O. 诺德 (Richard O. Nord) 发现了印刷上的错误,特德 雷诺 (Ted Reynold) 对QBE例题作了实验,阿克拉 萨纳 (Akram Salah) 提供了 PROLOG 语言方面的帮助。巴纳珍妮纳斯 爱德普甘迪 (Balanjaninath Edupuganty) 和刘蒂娜 (Tina Liu) 各帮助打字两章。

感谢我夫人玉华和女儿小佩在过去两年里对我的鼓励与关怀。

最后,对伯明翰阿拉巴马大学计算机与信息科学系提供我使用其有关设备,如 UNIX 字处理器和拼字程序、INGRES、PROLOG以及Versatek打印机表示由衷地感谢。

杨超植于北德克萨斯州立大学

译者的话

本书为杨超植教授的近作，是一本目前不可多得的专门讲述关系数据库原理与设计的教科书。

本书立足于坚实的数学基础之上。这不仅表现于最初章节对数学概念的回顾，而且这些概念贯穿于全书正文之中。全书共分为九章。第0章为数学概念的回顾，对书中涉及到的数学知识作了扼要阐述。第一章讲关系数据库模型以及关系数据库的一些基本概念。第二章讲述关系代数。第三章为关系谓词演算，包括元组关系演算与域关系演算。第四章讲述函数依赖。第五章讲述多值依赖。第六章讲述联接依赖。第七章讲述规范化问题。后面四章实际上是讲述数据库的设计问题。第八章讲述查询语言与数据库管理系统。作为一本关系数据库原理与设计的教科书，本书是完全可以信赖的。它适于高等学校计算机类专业（包括信息管理）的研究生以及高年级学生作为教材使用。而对于这方面的科研与工程技术人员来说，本书也是值得参考的好书。

杨超植教授祖籍湖南。在原书的最前页还有这样一段话：“本书献给我1948年后不曾见过已离人世的双亲杨申益与杨吴配文”，这表达了作者的思乡之情与愿为中华振兴出力之心。

译文尽量采用目前国内通用的术语，书末列有本书所用术语的英汉对照。

本书第0章由吴今培译，第一、二、三、八章由刘动天译，第四、五、六、七章、附录以及索引由张龙祥译。由于译者水平所限，译文错误之处在所难免，敬请读者批评指正。

1987年7月

序 言

许多知识领域最初只是一种技艺，直到后来才发展成为科学的领域。一个不成熟的领域只能由仍然还属技艺的尺度来衡量，它要求的也只是运用大量粗糙规则得出的直觉知识与实验，而每一这样的规则又只限于一个窄狭的范围。随着知识领域成熟，上面的这些特征逐渐消失，而由一个数量相当小但却强有力的可以系统应用的规则所取代。

七十年代中期我讲授数据库的时候，这一知识领域还只是一种技艺。（有些可说是魔术！）例如，有过三种竞争模式（关系模式、网络模式与层次模式），尽管如此，关系模式似乎提供了支撑基础，而其它两个模式则只具有特别实现的性质。

那时候，关系模式已经显示出若干第一流的面貌，特别是在应用关系代数或关系演算机会方面。另一方面，这种模式在关系规范化方面也表现出明显的缺点。试图实际设计关系数据库的人，由于远远超出了当时的理论而不得不后退到直觉知识与实验上来。

其后的十年取得了很大的进展。尤其是，提出了一种把关系改善为其“基本关键字范式”的算法，这是朝着关系数据库设计系统化方面迈出的重要一步。而且，它确认了长期受到怀疑的关系模式的优势；其余两种与之相竞争的模式，在对数据库设计提供系统化方法上都不是那么富有成效。

因此，现在似乎应当提出一种全新的数据库原理的表示方法，这种表示方法所体现的是支撑结构的完美特性，而摒弃其不必要的修饰。这就是杨超植博士写本书的目的与意义之所在。

本书正文分三部分。第一部分由一、二、三章组成，讲述关系模式的基本原理，包括关系代数及关系演算。第二部分为第四章至第七章，讲述数据库设计问题，特别是依赖与规范问题。第三部分即第八章，它包括上述章节导出的原则的四个实现。INGRES的叙述特别详尽，QBE所包括的范围相当广泛。PROLOG则为一种逻辑程序设计语言，已用于实现关系数据库系统，把它包括进来是希望将来使用它。杨博士用于INGRES、QBE以及PROLOG中的有关实例已在伯明翰阿拉巴马大学的系统中验证过。

本书适用于大学高年级或第一学年的研究生作一个学期的数据库原理教程。第一部分与第二部分应当在课堂讲授，加有星号的部分允许省略。全书各章配有习题，应特别鼓励学生做完。第三部分可作为补充资料发给学生，让他们发挥自己的创造性，并用他们容易访问的实际数据库系统去解决问题（例如，INGRES与PROLOG对于伯明翰阿拉巴马大学计算机与信息科学系的学生来说都是会使用的）。

数学是科学的语言。在强调数据库系统的科学性方面，可以说，这本书无愧是数学著作。杨博士在本书开始的一章以简明统一的数学符号和许多实例为读者提供了数学回顾。对于精通其内容的读者，本书提供了十分重要的基本理论的真谛，并精确阐明了数学术语。这样的书无疑是经久耐用的，因而相应的知识老化程度将相当缓慢。

然而，我不希望读者产生这样的想法：掌握了本书，数据库这门学科也就不那么光辉夺目了，它的设计者只需机械地按一下电键即可得到答案。超出基本关键字范式去改善关系模式仍然是一种技艺。把数据库应用于需要的场所将经常超出我们的知识范围。理解力、洞察力和创造性永远是数据库专业人员进行生机勃勃的一种特征。

总之，有一句老话仍然是真理：数据库的设计者必定能够运步于水上。杨博士的杰作能帮助这些设计者修补他们鞋子上的许多窟窿。

安东尼·巴纳德^① (Anthony Barnard)

^①安东尼·巴纳德是伯明翰阿拉巴马大学研究生院院长。——译者注

目 录

第0章 数学概念的回顾	(1)
0.1 引言	(1)
0.2 集合论	(1)
0.2.1 基本概念	(1)
0.2.2 集合运算	(1)
0.3 函数与关系	(2)
0.3.1 函数	(2)
0.3.2 关系	(3)
0.4 命题逻辑	(3)
0.4.1 原子与公式	(8)
0.4.2 公式解释	(4)
0.4.3 公式间的等价性	(4)
0.4.4 范式	(5)
0.4.5 逻辑后项	(5)
0.5 一阶逻辑	(5)
0.5.1 谓词	(5)
0.5.2 量化	(5)
0.5.3 公式	(6)
0.5.4 公式解释	(6)
0.5.5 范式	(7)
0.6 图论	(7)
0.6.1 无向图与有向图	(7)
0.6.2 超图	(8)
0.7 NP完全问题与NP困难问题	(9)
第一章 关系数据库模型	(10)
1.1 引言	(10)
1.2 属性和域	(10)
1.3 元组	(12)
1.4 关系数据库及模式	(15)
1.5 约束	(17)
1.6 数据库的关系模型	(22)
习 题	(22)
第二章 关系代数	(24)
2.1 引言	(24)
2.2 代数运算	(24)
2.2.1 并可兼容性	(24)
2.2.2 换名	(25)

2.2.3 并	(27)
2.2.4 差	(28)
2.2.5 复积	(28)
2.2.6 投影	(29)
2.2.7 选择	(31)
2.3 附加的代数运算	(32)
2.3.1 交	(32)
2.3.2 θ 联接	(33)
2.3.3 自然联接	(34)
2.3.4 商或除	(35)
2.4 关系代数	(35)
习题	(36)
第三章 关系谓词演算	(38)
3.1 引言	(38)
3.2 元组关系演算	(38)
3.2.1 公式	(38)
3.2.2 合法公式	(41)
3.2.3 元组演算表达式	(44)
3.2.4 安全元组演算表达式及其解释	(45)
*3.3 化关系代数为元组演算	(53)
3.4 域关系演算	(54)
*3.5 化元组演算为域演算	(56)
*3.6 化域演算为关系代数	(56)
习题	(60)
第四章 函数依赖	(63)
4.1 引言	(63)
4.2 函数依赖	(63)
4.3 逻辑等价性	(63)
4.4 函数依赖的独立推理规则	(65)
4.5 函数依赖的其它推理规则	(66)
4.6 函数依赖集的闭包	(68)
*4.7 推理规则的正确性与完备性	(71)
*4.8 关键字与超关键字	(72)
4.9 函数依赖集的覆盖	(73)
4.10 函数依赖的图形表示	(76)
习题	(89)
第五章 多值依赖	(91)
5.1 引言	(91)
5.2 多值依赖	(91)
5.3 逻辑等价性	(92)

5.4	多值依赖的立推理规则	(94)
5.5	多值依赖的其它推理规则	(96)
5.6	函数依赖和多值依赖的推理规则	(97)
5.7	依赖基	(99)
*5.8	嵌入多值依赖	(100)
*5.9	无矛盾的虚关键字	(101)
5.10	关于逻辑等价性的最后评语	(102)
	习题	(102)
第六章	联接依赖	(104)
6.1	引言	(104)
6.2	联接依赖	(104)
6.3	联接不丢失性的检验	(107)
6.4	存在定理	(109)
6.5	完全联接依赖的成员关系问题	(110)
*6.6	联接依赖的推理规则	(111)
	习题	(113)
第七章	规范化	(114)
7.1	引言	(114)
7.2	第一范式	(114)
7.3	第二范式	(114)
7.4	第三范式	(118)
7.5	基本关键字范式	(122)
7.6	基于函数依赖图的综合算法	(124)
7.6.1	求非冗余图覆盖的算法	(124)
7.6.2	求最小图覆盖的算法	(124)
7.6.3	求LR最小图覆盖的算法	(127)
7.6.4	寻找数据库模式	(131)
7.7	博依斯-科德范式	(132)
*7.8	关系模式的3NF、EKNF和BCNF之间的关系	(134)
7.9	第四范式	(135)
7.10	投影联接范式	(137)
7.11	水平规范化	(141)
7.12	关于规范化的最后评语	(141)
	习题	(142)
第八章	查询语言与数据库管理系统	(144)
8.1	引言	(144)
8.2	ISBL与PRTV	(144)
8.2.1	概述	(144)
8.2.2	个体、属性与有名变量	(144)
8.2.3	运算	(145)

8.2.4	用户扩展功能	(146)
8.2.5	PRTV 的其它特征	(147)
8.2.6	查询处理	(148)
8.3	QUEL 与 INGRES	(149)
8.3.1	概述	(149)
8.3.2	QUEL	(149)
8.3.3	INGRES	(152)
8.3.4	EQUEL	(152)
8.3.5	建立或删除数据库	(153)
8.3.6	进入或离开INGRES 环境	(153)
8.3.7	建立关系模式	(153)
8.3.8	建立或删除关系	(154)
8.3.9	显示与存储关系	(156)
8.3.10	打印查询缓冲器或查询缓冲器清零	(156)
8.3.11	存储结构	(157)
8.3.12	二级索引	(158)
8.3.13	系统关系	(159)
8.3.14	编辑工作空间	(160)
8.3.15	查询处理	(161)
8.4	Query by Example 与 QBE数据库管理系统	(165)
8.4.1	概述	(165)
8.4.2	进入QBE环境	(166)
8.4.3	定义数据表格	(167)
8.4.4	命令表格与条件表格	(169)
8.4.5	单表处理	(170)
8.4.6	多表处理	(172)
8.4.7	集聚	(178)
8.4.8	换名	(178)
8.4.9	删除关系模式	(178)
8.5	PROLOG逻辑程序设计与 PROLOG数据库管理系统	(179)
8.5.1	概述	(179)
8.5.2	逻辑子句形式与霍恩子句	(179)
8.5.3	UNIX与PROLOG 环境	(180)
8.5.4	术语与符合的对应	(180)
8.5.5	拷贝会话期	(180)
8.5.6	本文文件与索引文件	(181)
8.5.7	知识表达	(181)
8.5.8	系统关系的初始化与维护	(182)
8.5.9	建立关系	(183)
8.5.10	查询处理	(184)
8.5.11	修改与删除元组	(192)
	习题	(193)
	附录: 具有数据库选项的扩展 PROLOG	(196)

文献目录.....	(199)
英中名词对照表	(207)

第0章 数学概念的回顾

0.1 引言

本章中，我们要回顾一些有关的数学概念，这些概念是定义本书中关系数据库模型的基本工具。我们特别要重温一下集合论、关系、函数、命题逻辑、一阶逻辑以及图论。本章定义的许多重要概念，虽不是本书成立的必要前提，但在后面章节中将要多次涉及。

0.2 集合论

0.2.1 基本概念

不同个体的汇集称为一个集合，我们用大括号 $\{ \}$ 括起来表示。其中，用逗号将每个个体分开。不含个体的集合称为空集，用符号 ϕ 表示。所有个体（不必是相互区别的）的集合称为多重集，我们用方括号 $()$ 表示。其中，每一个体用逗号分开。本节我们仅回顾有关集合的概念。

集合的个体称为集合的元素或集合的成员。若集合 e 是集合 S 的一个元素，则说 e 属于 S ，或者说 e 在 S 中，记作 $e \in S$ 。相反，若 e 不属于 S 或 e 不在 S 中，则记作 $e \notin S$ 。集合的基数记作 $|S|$ ，它是 S 中元素的数目。根据基数为有限或无限而确定集合是有限的还是无限的。基数为1的集合称为奇异集。

若集合 S_1 和 S_2 由同样的元素组成，则称集合 S_1 与 S_2 相等，记作 $S_1 = S_2$ 。如果 S_1 的每一元素也是 S_2 的一个元素，则集合 S_1 为集合 S_2 的真子集。 S_1 （真正）包含或包括于 S_2 ，或者说 S_2 （真正）包含或包括 S_1 ，记作 $S_1 \subseteq S_2$ ($S_1 \subset S_2$)。当 S_1 为 S_2 的子集时，我们也说 S_2 为 S_1 的超集。一个集合既是它自身的子集，也是它自身的超集。按照 S_1 与 S_2 之间的包含关系， $S_1 = S_2$ 则等价于 $S_1 \subseteq S_2$ 及 $S_2 \subseteq S_1$ 。

一个有限集合 S 的幂集由 S 的所有子集组成（包括 S 本身及空集 ϕ ），记作 $P(S)$ 。即

$$P(S) = \{S_j | S_j \subseteq S, j = 1, 2, \dots, 2^{|S|}\}$$

0.2.2 集合运算

两个集合 S_1 与 S_2 的并集是指 S_1 或 S_2 （或两者）中所有元素组成的集合，记作 $S_1 \cup S_2$ 。即

$$S_1 \cup S_2 = \{e | e \in S_1 \text{ 或 } e \in S_2\} \quad (0.1a)$$

S_1 与 S_2 的并集具有既包含 S_1 也包含 S_2 的性质。

两个集合的交集是指同属于 S_1 及 S_2 的所有元素组成的集合，记作 $S_1 \cap S_2$ 。即

$$S_1 \cap S_2 = \{e | e \in S_1 \text{ 且 } e \in S_2\}$$

S_1 与 S_2 的交集具有它既包含于 S_1 又包含于 S_2 中的性质。若两个集合 S_1 与 S_2 的交集为空集 ϕ ，则称它们是互不相交的或不可兼容的。如果某集合族的每两个不同的集合都是互不相交的，则称此集合族为互不相交的。

集合 S_1 与 S_2 的差是指包含在 S_1 中而不包含在 S_2 中的所有元素组成的集合，记作 $S_1 - S_2$ 。即

$$S_1 - S_2 = \{e | e \in S_1 \text{ 且 } e \notin S_2\}$$

如果 S_2 是 S_1 的子集, 则 $S_1 - S_2$ 称为 S_2 相对于 S_1 的补集。若 S_1 为已知, 则 S_2 (相对于 S_1) 的补集可以记作 $\sim S_2$ 。即

$$\sim S_2 = \{e | e \in (S_1 - S_2) \text{ 且 } S_2 \subseteq S_1\}$$

两集合 S_1 与 S_2 的复积是指 S_1 中的 a 与 S_2 中的 b 形成的所有串 ab 组成的集合, 记作 $S_1 * S_2$ 。其中, a 与 b 是连接在一起的。即

$$S_1 * S_2 = \{a b | a \in S_1 \text{ 且 } b \in S_2\} \quad (0.2a)$$

注意, a 与 b 之间的二元连接被省略了, 而 a 与 b 间至少用一个空格分开。

两集合 S_1 与 S_2 的笛卡儿积是指所有的 a 属于 S_1 和 b 属于 S_2 的有序对 (a, b) , 记作 $S_1 \times S_2$ 。即

$$S_1 \times S_2 = \{(a, b) | a \in S_1 \text{ 且 } b \in S_2\} \quad (0.3a)$$

二元运算交、并及复积具有结合性, 因而可以自然地扩充至 K 元运算。用前缀符号表示如下:

$$\begin{aligned} \cap (S_1, \dots, S_k) &= \{e | e \in S_1 \text{ 且 } \dots \text{ 且 } e \in S_k\} \\ \cup (S_1, \dots, S_k) &= \{e | e \in S_1 \text{ 或 } \dots \text{ 或 } e \in S_k\} \end{aligned} \quad (0.1b)$$

$$* (S_1, \dots, S_k) = \{e_1 \dots e_k | e_j \in S_j, 1 \leq j \leq k\} \quad (0.2b)$$

式 (0.2b) 中, 每一元素 $e_1 e_2 \dots e_k$ 称为长度为 k 的串, k 为串中连接的个体数。

二元运算笛卡儿积不具有结合性。即 $(S_1 \times S_2) \times S_3 = \{((a, b), c) | (a, b) \in S_1 \times S_2, c \in S_3\}$ 且 $S_1 \times (S_2 \times S_3) = \{(a, (b, c)) | a \in S_1, (b, c) \in S_2 \times S_3\}$, 其中 a, b, c 分别属于 S_1, S_2, S_3 。 k 元笛卡儿积不能自然地由二元形式扩充而成, 但它可以定义为

$$\times (S_1, \dots, S_k) = \{e_1 \dots e_k | e_j \in S_j, 1 \leq j \leq k\} \quad (0.3b)$$

式 (0.3b) 中的每一元素称为 k -元组 (或简称元组), 其目为 k , 即元组中的个体数为 k 。若 $S_1 = \dots = S_k = S$, 则式 (0.3b) 的 k 元笛卡儿积可简写为 S^k 。

0.3 函数与关系

0.3.1 函数

设 S 与 S_j 为个体的集合 (对每一个 $j, 1 \leq j \leq n$)。 n 自变量的函数 $f: \times (S_1, \dots, S_n) \rightarrow S$ (即从笛卡儿积 $\times (S_1, \dots, S_n)$ 到集合 S 的映射) 是一笛卡儿积 $\times (S_1, \dots, S_n, S)$ 的子集, 使得对于任意两个元组 $(a_1, \dots, a_n, a_{n+1})$ 与 $(b_1, \dots, b_n, b_{n+1})$, 如果有 $a_j = b_j$ (对每一个 $j, 1 \leq j \leq n$), 则我们必须有 $a_{n+1} = b_{n+1}$ (或者等价地, 如果有 $a_{n+1} \neq b_{n+1}$, 则我们必须有 $a_j \neq b_j$ (对某个 $j, 1 \leq j \leq n$))。即

$$f = \{(a_1, \dots, a_n, a_{n+1}) | (a_1, \dots, a_n) \in \times (S_1, \dots, S_n) \text{ 且 } S \text{ 中有唯一的 } a_{n+1}\}$$

n 自变量的函数又称为 n 元函数。函数也称为映射。如果 $(a_1, \dots, a_{n+1})$ 属于 f , 则最后的元素 a_{n+1} 称为自变量的元组的 f 值, 或称为自变量的元组 $(a_1, \dots, a_n)$ 在 f 作用下的映象。如果不存在 a_{n+1} 使 $(a_1, \dots, a_{n+1})$ 属于 f , 则元组 $(a_1, \dots, a_n)$ 的 f 值或元组 $(a_1, \dots, a_n)$ 在 f 作用下的映象为未被定义的 (当被定义时, 这意味着部分函数)。

函数 f 的域为一集合

$$D_f = \{(a_1, \dots, a_n) | (a_1, \dots, a_{n+1}) \in f\} \quad (0.4)$$

f 的域为笛卡儿积 $\times (S_1, \dots, S_n)$ 的一个子集。如果 f' 的域为 D_f 的真子集, 则函数 f 的限制 f'

是一映射。 f 的限制可以使用同样的名字 f 。 f 的范围为集合。

$$R_f = \{ \langle a_{n+1} \mid \langle a_1, \dots, a_{n+1} \rangle \in f \}$$

f 的范围是集合 S 的一个子集, 集合 S 称为 f 的共域(即范围 R_f 的超集)。

若 $R_f = S$, 则函数 $f: \times (S_1, \dots, S_n) \rightarrow S$ 从其域 D_f 向其共域映射, 而若 $R_f \subset S$, 则函数 f 从其域 D_f 映射入其共域内。如果函数的域为 $S_1, \dots, S_{n-1}$ 及 S_n 的笛卡儿积的真子集(即 $D_f \subset \times (S_1, \dots, S_n)$), 则函数 $f: \times (S_1, \dots, S_n) \rightarrow S$ 为一部分函数。在此情况下, $\times (S_1, \dots, S_n)$ 与 f 作用下的 D_f 之差的每个元组的映象是未定义的。

如果对于属于函数 f 的任意两不同元素 $\langle a_1, \dots, a_{n+1} \rangle$ 与 $\langle b_1, \dots, b_{n+1} \rangle$ 不存在 $a_{n+1} = b_{n+1}$ 的情况, 则函数 f 为一对一的, 否则为多对一的。通常称一对一及映上函数 f 为一一对应。如果函数 f 为一对一, 则函数的逆(表示为 f^{-1})存在, 使得 f 与 f^{-1} 的合成(表示为 $f \circ f^{-1}$)与合成 $f^{-1} \circ f$ 为相同的。其中, 两个合成定义为恒等函数, 即 $I = f \circ f^{-1} = f^{-1} \circ f$ 。符号“ $\circ$ ”表示“合成”。

0.3.2 关系

$\times (S_1, \dots, S_{n-1})$ 与 S 间 n 自变量的关系是笛卡儿积 $\times (S_1, \dots, S_{n-1}, S)$ 的子集。 n 自变量的关系也称 n 元关系。因此, n 自变量的关系是 $n-1$ 自变量函数的推广。若关系为一对多或多对多的, 则它不是函数。

特别使人感兴趣的是集合 S 上(即 S 与 S 间)的二元关系 r 。如果对每一属于 S 的 e , $\langle e, e \rangle$ 属于 r , 则关系 r 为反射性的, 否则不是反射性的。如果对每个 S 中的 e , $\langle e, e \rangle$ 不属于 r , 则关系 r 为非反射性的。如果无论何时 $\langle e_1, e_2 \rangle$ 属于 r , $\langle e_2, e_1 \rangle$ 也属于 r , 则关系 r 为对称的, 否则为不对称的。如果无论何时 $\langle e_1, e_2 \rangle$ 属于 r , 而 $\langle e_2, e_1 \rangle$ 不属于 r , 则关系为非对称的。如果 $\langle e_1, e_2 \rangle$ 与 $\langle e_2, e_1 \rangle$ 均属于 r , 意味着 $e_1 = e_2$, 则关系 r 为反对称的。如果无论何时 $\langle e_1, e_2 \rangle$ 与 $\langle e_2, e_3 \rangle$ 都属于 r , 而 $\langle e_1, e_3 \rangle$ 也属于 r , 则关系 r 为传递性的, 否则不为传递性的。如果关系 r 是反射性的、对称性的与传递性的, 则关系 r 为 S 上的等价关系。

集合 S 的划分是集合 S 某些子集的集合, 使得各子集的并集等于 S , 而每两个不同的子集则是互不相交的。集合 S 上的等价关系定义出 S 上的划分, 其元素则称为等价类。

0.4 命题逻辑

0.4.1 原子与公式

我们在命题逻辑中运用五种逻辑运算符或联接词: $\neg$ (“逻辑非”或“求反”), $\wedge$ (“逻辑与”或“合取”), $\vee$ (“逻辑或”或“析取”), $\Rightarrow$ (“蕴涵”或“如果...则”), $\equiv$ (“等价”或“当且仅当”(简写为iff))。

一个真或假的陈述句(不可能同时既真又假)称为一个命题。用来标记命题的符号通常称为原子公式(或简称原子)。我们可以用五种逻辑运算符由简单命题构成复合命题。通常, 一个代表命题或复合命题的表达式称为一个合式公式(或称为公式)。在命题逻辑中, 公式递归定义如下:

(1) 一个原子是一公式。

(2) 如果 F 是一个公式, 则 $\neg F$ 是一个公式。

(3) 如果 F 和 G 是公式, 则 $(F \cdot G)$ 、 $(F \vee G)$ 、 $(F \Rightarrow G)$ 和 $(F \equiv G)$ 也是公式。

(4) 除此之外均不是公式。

如果事先指定逻辑运算符运算的先后次序, 从高权到低权为 $-$ 、 $\cdot$ 、 $+$ 、 $\Rightarrow$ 、 $\equiv$, 我们就能省略括号。

如果 F 为假, F 的求反公式 $\bar{F}$ 则为真, 否则为假。如果 F 和 G 都为真, 则 F 和 G 的合取式 $F \cdot G$ 也为真, 否则为假。如果 F 和 G 中至少有一个为真, 则 F 和 G 的析取式 $F + G$ 也为真, 反之则为假。如果 F 为真 G 为假, 则式 $F \Rightarrow G$ 读作“ F 蕴涵 G ”或“如果 F 则 G ”为假, 反之则为真。若无论何时 F 和 G 具有同样的真值 (或为真或为假), 则 $F \equiv G$ 为真, 读作“ F 等价于 G ”或“ F iff G ”, 反之则为假。

0.4.2 公式解释

给定一个公式 F , 令 $A_1, \dots, A_{n-1}$ 及 A_n 为 F 中出现的原子。 F 的解释是按对 $A_1, \dots, A_{n-1}$ 及 A_n 所赋真值而定, 其中每个 A_j (对 $j, 1 \leq j \leq n$) 赋值为其真或为假。既然对 n 个原子有 2^n 个可能的方法赋真值, 故 F 就存在 2^n 种解释。如果按解释赋值给 F 为真, 则公式 F 称为在此解释下为真, 否则为假。如果一个公式在所有的解释下均为真, 则称之为永真式或重言式, 否则为永假式。如果一个公式在所有解释下均为假, 则其称为不相容的或矛盾, 否则称为相容的。如果一个公式是永真的, 则就为相容的, 但反之并不成立。同样, 如果一个公式是不相容的, 则它是永假的, 但反之也不成立。例如, 公式 $F \Rightarrow \bar{F}$ 是永假的, 因为它不是重言式, 但它是相容的, 因为它不是矛盾。如果一个公式 F 在解释 M 下为真, 则我们说 M 满足 F 或 F 满足于 M 。另一方面, 如果公式 F 在解释 M 下为假, 则我们说 M 不满足 F 或 F 不满足于 M 。

0.4.3 公式间的等价性

如果 F 与 G 的真值在每个解释下都相同, 则两公式 F 和 G 等价或 F 等价于 G , 记作 $F \equiv G$ 。例如在简化公式时, 下述等价关系非常有用。

$$F \Rightarrow G \equiv \bar{F} + G \quad (0.5)$$

$$(F \equiv G) \equiv \bar{F} \cdot \bar{G} + F \cdot G \quad (0.6)$$

式 (0.6) 中, 因为计算其值的扫描方向习惯上是从左至右, 故 $F \equiv G$ 的括号可以省略。

设 F 是一个矛盾, T 是重言式, G 与 H 是公式, 下列等价关系在处理第四章和第五章中的函数依赖和多值依赖时非常有用。

$$G + F \equiv G \quad (0.7a)$$

$$G + T \equiv T \quad (0.8a)$$

$$G + \bar{G} \equiv T \quad (0.9a)$$

$$\bar{T} \equiv F \quad (\text{狄摩根 (DeMorgan) 定律}) \quad (0.10a)$$

$$G + \bar{G} \cdot H \equiv G + H \quad (0.11a)$$

$$\bar{G} + \bar{H} \equiv \overline{G \cdot H} \quad (\text{狄摩根定律}) \quad (0.12a)$$

按对偶原理, 我们得到更有用的等价关系

$$G \cdot T \equiv G \quad (0.7b)$$

$$G \cdot F \equiv F \quad (0.8b)$$

$$G \cdot \bar{G} \equiv F \quad (0.9b)$$

$$\bar{\bar{F}} \equiv F \quad (\text{狄摩根定律}) \quad (0.10b)$$

$$G \cdot (\bar{G} + H) \equiv G \cdot H \quad (0.11b)$$

$$\overline{G \cdot H} \equiv \overline{G} + \overline{H} \quad (\text{狄摩根定律}) \quad (0.12b)$$

此外, 等价关系

$$(F+G) \cdot (F+H) \equiv F+G \cdot H \quad (0.13)$$

对产生简化公式也非常有用。在两个公式等价时, 为达到简化或甚至小型化, 较复杂的公式可用其等价公式代替。

0.4.4 范式

一个原子或原子求反(叫做求反过的原子)称为一文字。如果公式F是子公式 F_1 至 F_k 的合取, 其中每个子公式 $F_j, 1 \leq j \leq k$ 是某些文字的析取, 则F属于合取范式。如果公式F是子公式 F_1 至 F_k 的析取, 其中每个子公式 $F_j, 1 \leq j \leq k$ 是某些文字的合取, 则F属于析取范式。

0.4.5 逻辑后项

给定公式 $F_1, \dots, F_n$ 和G, 如果对任何解释公式 $F_1, \dots, F_n$ 为真, G也为真, 则G称为 $F_1, \dots, F_{n-1}$ 和 F_n 的逻辑后项, 或者称 $F_1, \dots, F_{n-1}$ 和 F_n 逻辑蕴涵G。

命题逻辑中有一个在关系数据库中很有用的定理。这个定理叙述为: “给定公式 $F_1, \dots, F_n$ 与G, 如果公式 $F_1 \cdot \dots \cdot F_n \Rightarrow G$ 为永真式, 则G是 $F_1, \dots, F_{n-1}$ 和 F_n 的逻辑后项。”由式(0.5)与式(0.12b), 公式 $F_1 \cdot \dots \cdot F_n \Rightarrow G$ 等价于 $\overline{F_1} + \dots + \overline{F_n} + G$ 。如果 $F_1, \dots, F_n$ 和G是原子, 则公式 $\overline{F_1} + \dots + \overline{F_n} + G$ (仅仅有一个不取反的原子)称为一个Horn子句。无任何不取反的原子的公式 $\overline{F_1} + \dots + \overline{F_n}$ 也是一个Horn子句。

0.5 一阶逻辑

0.5.1 谓词

一阶逻辑有三个逻辑概念: 项, 谓词及量词。项递归定义为

(1) 一个常量是一个项。例如一个数值常数或一个非数值符号(字符串为一例子)。

(2) 一个变量是一个项。

(3) 如果f是一个n元函数符号, $t_1, \dots, t_{n-1}$, 和 t_n 是项, 则 $f(t_1, \dots, t_n)$ 是一个项。

(4) 除此之外均不是项。

一个n元谓词是一个从 $\times(S_1, \dots, S_n)$ 到 $\{\text{true}, \text{false}\}$ 的n元函数, 其中“true”与“false”为真值。n元谓词也称为n位谓词。一阶逻辑中的原子正式定义如下: 如果p是一个n元谓词符号, $t_1, \dots, t_{n-1}$ 及 t_n 是项, 则 $p(t_1, \dots, t_n)$ 是一个原子。有四种类型的符号能用于构成原子: 常量, 变量, 函数符号与谓词符号。

0.5.2 量化

在程序设计语言中, 当一个过程P内包含一个过程Q且一标识符x在Q中出现时, 若x在Q中说明, 则x是一个局部变量, 否则就是Q中的全程变量。在一个公式F中, 一个变量x在F中是自由变量或约束变量, 分别类似于程序设计语言中的全程变量与局部变量。

全称量化变量为 $\forall x$, 读作“对所有x”, “对每个x”或“对一切可能的x”。存在量化变量为 $\exists x$, 读作“存在一个x”, “对某些x”或“对至少一个x”。符号 $\forall$ 与 $\exists$ 分别称为全