

数据库技术入门

C. J. 戴特 著

敬 喜 等译
王文叔

范依远 校

电子工业出版社

内 容 提 要

本书以简洁、通俗易懂的语言介绍了数据库技术中的基本概念,用户最关心的怎样使用数据库,以及数据库管理和数据库设计等重要方法。主要的内容有:数据库的建立和查询,典型维护操作,数据显示和报告的编写,数据库查询语言SQL、QBE和NOMAD,数据库dBASE II索引,视图的概念和使用,安全性和完整性概念,封锁技术以及数据库设计等。读者对象:假定你想使用或了解数据库系统,而你对计算机如何工作懂得很少或完全不懂,那么,本书就是您的最好向导。

数据库技术入门

C. J. 戴特 著

敬 喜 王文叔 等译

范依远 校

责任编辑 吴明卒

*

电子工业出版社出版 (北京市万寿路)

上海新华印刷厂排版 人民卫生出版社印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

开本: 850×1168 1/32 印张: 8.25 字数: 212 千字

1987年8月第1版 1987年8月第1次印刷

印数: 10000册 定价: 1.95元

ISBN 7-5053-0138-1/TN68

统一书号: 15290·496

目 录

前 言

第一编 数据库的存取与维护

第一章 第一编的引言	2
1.1 目标	2
1.2 本书的结构	3
第二章 数据库管理概论	7
2.1 什么是数据库系统	7
2.2 数据库系统带来的益处	9
2.3 多重表的例子	10
2.4 查询语言与数据操纵	14
2.5 其他操作	16
2.6 其他的例子	18
2.7 小结	21
第三章 机器在做什么	23
3.1 引言	23
3.2 计算机系统结构	24
3.3 程序	27
3.4 数据库管理系统	28
3.5 语言的级别	29
3.6 自然语言	35
3.7 数据独立性	37
3.8 其他数据库管理功能	38
3.9 小结	40
第四章 事例研究——数据库的建立和查询实例	42
4.1 引言	42
4.2 建立数据库	43
4.3 装填数据库	49
4.4 查询示例	51
4.5 小结	63

4.6 练习	64
4.7 答案	64
第五章 事例研究(续)——维护操作示例	66
5.1 引言	66
5.2 异常	68
5.3 多重语句更新	69
5.4 “罐装(canned)”过程	74
5.5 应用程序生成器(Application Generator)	78
5.6 小结	79
第六章 数据显示与报告的编写	80
6.1 引言	80
6.2 数据显示	80
6.3 报告的编写	82
第七章 数据库名录(catalog)	87
7.1 引言	87
7.2 一个例子	88
7.3 查询目录	90
7.4 更新目录	92
7.5 目录与字典的比较	94
7.6 小结	95
7.7 答案	95
第八章 用示例查询(QBE)	97
8.1 引言	97
8.2 查询	100
8.3 更新	103
8.4 数据定义	105
8.5 查询目录	106
8.6 小结	107
8.7 答案	108
第九章 NOMAD	111
9.1 引言	111
9.2 数据定义	113
9.3 查询和格式化操作	116
9.4 更新操作	123
9.5 查询目录	127
9.6 小结	128

9.7 答案	128
第十章 dBASE II	131
10.1 引言	131
10.2 数据定义与装填	132
10.3 查询	134
10.4 更新操作	136
10.5 “罐装”过程	140
10.6 报告的编写	145
10.7 查询目录	150
10.8 小结	151
10.9 答案	152
 第二编 数据库设计和控制	
第十一章 第二编的引言	156
11.1 预习	156
11.2 结构	158
第十二章 索引(indexes)	160
12.1 引言	160
12.2 直接存取与顺序存取的比较	163
12.3 进一步的考虑	165
12.4 系统的例子	168
12.5 小结	169
第十三章 视图(views)	171
13.1 引言	171
13.2 其他的例子	174
13.3 NOMAD设施	176
13.4 视图的优点	179
第十四章 安全性(security)	181
14.1 引言	181
14.2 权力的转让	182
14.3 视图和授权	184
14.4 权限的撤消	185
14.5 QBE设施	186
14.6 安全性的其他方面	186
第十五章 完整性(integrity)	190
15.1 引言	190

15.2	完整性约束	191
15.3	主码(primary keys)	192
15.4	外部码(foreign keys)	194
15.5	格式约束	195
15.6	范围约束	196
15.7	结论	197
第十六章 闭锁(locking)		199
16.1	引言	199
16.2	闭锁	201
16.3	闭锁的解除	204
16.4	死锁	205
16.5	小结	206
第十七章 数据库设计		208
17.1	引言	208
17.2	部门与雇员——一个好的设计	210
17.3	部门与雇员——一个差的设计	211
17.4	部门与雇员——另一个差的设计	214
17.5	讨论	216
第十八章 数据库设计(续)		221
18.1	引言	221
18.2	供应商与零件——一个好的设计	221
18.3	讨论	222
18.4	进一步的评论	224
18.5	各个用户的视图	226
18.6	小结	227
18.7	练习	228
18.8	答案	230
跋		234
附录A 关系模型		238
附录B 进一步阅读指南		245

第一编 数据库存取与维护

第一章 第一编的引言

1.1 目 标

数据库管理是现代计算机系统所提供的最重要的功能之一。它是如此之重要,以致于它常常成为请求首次得到计算机的主要理由。本书的首要目的是说明数据库管理对于您,即用户意味着什么。因此,目标是叙述从外部看起来数据库系统象什么,而不是从内部看它们怎么工作的,虽然我们将时时研究某些内情(但不是很深入的)。不过,我们必须首先弄清楚当我们提到“用户”这个术语时,其含义是什么。

一般地说,凡是有兴趣于通过自动化(如计算机化)系统在任一种复杂性级别上,来维护和询问有关任何题目的数据文卷的人都称之为“用户”。

更具体地说,“用户”这个术语在本书中的含义是指下列情况之一。

(1) 拥有自己的家庭计算机,如Apple II或TRS80或IBM PC或任何其他上市的机器的人,可称之为“用户”;

(2) 小企业,如小的会计公司,或地方零售店中的人,拥有一台微型或小型计算机以帮助运行该企业;

(3) 大机构中的最终用户,如银行、工业公司、医院或大学机构中的用户。他们通过计算机终端连到某个大的也可能是远地的由某个数据处理部门专家所控制的中心计算机,去存取数据。

这些不同类型的人们,虽然具有各种截然不同的要求,然而,从他们要求数据库去做的问题归纳起来,却有很多共同的地方。在

所有情况下,这些要求可能是把数据记录存入系统,也可能是根据请求即时检索信息,也可能是在需要时更新数据。因此,从外部来看系统在这三种情况下都非常相似。但从内部来说,它们的结构可能是有很大差异的。本书所关心的是外部,我们不必对这三者之间做过多的区分。

本书第二项目标(也可以说是第一目标的另一种不同表达形式)就是消除人们对数据库系统的神秘感。由于某种原因,对数据库管理这一课题曾经产生了一种神秘感。经常发现有些人(甚至计算机专业人员)被此课题所吓倒:“啊!那对我来说太复杂了,我永远也弄不懂它,不懂得高深的数学能使用数据库吗?”等等。也许是有关这一题目的术语促成了上述看法:由于大量使用术语、行话,使人手足无措,因而声名狼藉的领域(即计算机科学)中,数据库管理作为一个最坏的榜样显得尤为突出。数据库内部确实是个复杂的问题。至于设计和控制一个大的完善的数据库,确实也是一件复杂的工作。从而在一个巨大的组织里数据库系统的实现和数据库的管理都是非常专门化的工作。这两方面都要求高度熟练技能的人来从事这项工作。但我觉得这个任务不是由用户来完成的。从全局来看,所有这一切复杂性都是为了使用户使用方便。从用户的角度看数据库是简单的。(凭良心说,也许有些老式系统在使用时未能做到它本来应有的那么方便!但是一般来说,上述论断将随着时间的推移变得愈来愈符合实际情况。并且新的系统肯定能适应这种要求。)可惜有时候专用术语还是令人感到模糊不清。我们将在本书中试图把这些晦涩之处弄得清楚明朗一些。

1.2 本书的结构

你们可能看到,本书分两编,第一编由十章组成,第二编共有八章。纳入第一编中的是那些使你们最先感到兴趣的问题,即:怎么在一开始时使数据装入数据库?其次是随后如何取出(和当需

要时如何更换)。这些方面叫做数据库的存取与维护。相比之下,第二编中的题目在使用系统的初期不需要理解,但为了使你对系统在干些什么能有所理解却是应该知道的。第二编有它单独的引言(第11章)。

第一编结构如下:紧接着本章引言之后,第二章提供了整个数据管理功能的概况。为此,引进一个完整的例子,以表明一个简单的数据库看起来是什么样子,并说明你如何能在任何时候都可以对数据库进行存取(查询或更新数据)。假若此刻我能和你一起坐在计算机前,这就是我将在系统上向你演示的第一件事。然而这是不可能的,因此下一种较好的办法就是尽力在纸面上来做同样的演示,从而就安排了这个例子。当然,这个例子很简单,但它却是完整的。我相信对读者最重要的是一开始就有一个总体概念,这样在你心目中就有了一个框架,使你在阅读本书以后的章节中的材料时,你能够把它们与框架中的各部分一一对应起来。因此,到第二章结束时,尽管你们没能掌握很多细节(或者没有领会你所看到的全部细节),你也会对你与数据库系统打交道时的种种事项有个全面的印象。

在机器前面坐下的想法引起了另一点做法。本书中含有大量例子和为读者作的一些练习(其中一部分有答案)。如果您有机会使用机器,最好随时将这些例子和练习在机器上试一试。把本书和机器结合起来,你可以学得更多更快些。

再回到本书的结构问题上来。第三章简述了机器必须做些什么工作,才能提供第二章中所叙述的功能。这部分材料在性质上是提供背景(基础)信息;您并非一定要理解这些材料后才能去使用系统,但若你能对它的工作过程有所理解,你当然就有可能把系统使用得更好一些,因此不管怎样我都希望这些材料能使你感兴趣。当然,如果你对于计算机工作情况的基本概念已有了解,就可以跳过这一章。

第四、五章涉及到实例(实际应用)的研究。第四章讨论检索,

第五章讨论数据的更新。因为应用是一种实际应用，它不可避免的会包含有复杂性（实际生活永远不象课本上虚构例子那样简单）。事实上，在许多方面这两章特别是第四章是全书中最难的部分。但请不要被它吓住！这两章开始时都相当简单，逐渐地复杂。若读者发现在细节上被卡住了，只需跳过去，读下一章。为了能继续阅读后边的内容，无需领会这两章的所有细节。但是，当读者着手使用自己的数据库时，你还可以回过头来再学有关的细节。

第六章讨论“报告书写”问题，也就是讨论如何从你的系统获得打印的报告。第七章描述数据库系统的某一有用的特性，即所谓“目录”。

更深入地讨论具体的系统不是本书的目的。一方面这样的系统很多（这种系统差不多每天都会出现）。另一方面（也是更重要的一面），我们关心的是概念、原理和思想，而不是系统中的具体细节（随着系统的改进，细节总是在不断变化）。尽管如此，第二到七章中的材料主要是通过一个具体的系统（即查询语言SQL）来表述，其理由就是为了前后连贯与简单易学。因此，为了表明还有其他的办事方法（即与SQL不同的方法），第八至十章粗略介绍另外三种系统：QBE、NOMAD、dBASE II。请注意，我选择这些特定的系统是因为这些系统非常完美地显示了我在本书中想要表达的观点。我这样做并非认为所有的人都赞成这几个系统。同样地，某些系统在本书中未被介绍并不意味着那些系统无人赞同或被贬低。

让我再强调一次，本书的总任务并不是要教授有关任何具体系统精细的内容，而是企图给您以一些总的概念。让我在这里一开始就明确以下几点：

（1）为了清晰和一致，在叙述任何一个给定的系统时，我将毫不犹豫地简化次要的方面，并略去一些次要的限制。

（2）我有意没有对每个系统的每个方面进行阐述，甚至有几个十分重要的特性也没有提到。

这两点适用于本书讨论的所有系统，以后在每章中将不再重

复申明。

尽管放弃了这么多东西,您可以依然会感到与具体系统有关的细节仍比您所需要的要多。我无法否认确有相当多的这类细节,不过这些细节是经过相当审慎的考虑后留下的。根据我自己学习与讲授这部分材料的经验,我坚信您需要在不忽略主要细节的条件下观察几个具体例子,这样您才能真正领悟到数据库系统能够做到那些,不能够做到那些。我这样说并不意味着要求您搞懂全部细节,我只不过说您至少应“不费劲地再读一次”,并大概地领悟其中所包含的事实。请让我强调“不费劲地”这几个字。重要的是不要被拦路虎挡住。要是真发生这种情况,就改看下一个题目或下一章。这些材料事实上并不难,但有时可能会费点周折。等读到第十章末尾,你将对数据库系统能够做什么获得一个相当全面的了解,并且为阅读第二编作好准备。如前已指出,第二编中各章将介绍很多补充的题目,但这些题目绝大部分属于系统需要研究的事项,而不是直接与用户有关的考虑。不如说本书第十一章给出了第二编的轮廓。

现在读者即将阅读(第二章)数据库管理概论。让我在此表示:对本书部分或全部内容欢迎提出问题意见,批评和建议,作为引言的结束。来信请寄:

C.J. Date, c/o Addison-Wesley Publishing Company,
Reading, Massachusetts 01867.

谢谢,祝您阅读顺利。

第二章 数据库管理概论

2.1 什么是数据库系统

数据库实质上不过是计算机化的记录保存系统。传统的办法是把许多文件保存在纸上,现在能够更方便地保存在数据库中。例如在我的计算机中有一个小的数据库,它帮助我记住地下室中有关藏酒的情况。为了在任何时候都能精确地知道库存,可打入下述命令:

SELECT * FROM CELLAR

(我们随即就要研究这个命令的细节)。计算机的响应是:在荧光屏上显示出我所寻求的信息:

<u>BIN</u>	<u>WINE</u>	<u>PRODUCER</u>	<u>YEAR</u>	<u>BOTTLES</u>	<u>READY</u>	<u>COMMENTS</u>
2	Chardonnay	Buena Vista	79	1	82	
3	Chardonnay	Louis Martini	79	5	82	
6	Chardonnay	Chappellet	79	4	82	Thanksgiving
11	Jo. Riesling	Jekel Vineyard	80	6	82	
12	Jo. Riesling	Buena Vista	77	1	80	Late Harvest
16	Jo. Riesling	Sattui	79	1	81	very dry
21	Fume Blanc	Ch.St.Jean	79	4	81	Napa Valley
22	Fume Blanc	Robt.Mondavi	77	2	80	
25	Wh.Burgundy	Mirassou	80	6	82	
30	Gewurztraminer	Buena Vista	80	3	82	
43	Cab.Sauvignon	Robt.Mondavi	77	12	83	
50	pinot Noir	Mirassou	77	3	82	Harvest Sel.
51	pinot Noir	Ch.St.Jean	78	2	84	
64	Zinfandel	Mirassou	77	2	84	Anniversary
72	Gamay	Robt. Mondavi	78	2	82	

表(CELLAR表)中列出了地下室全部存酒清单。现在假设我决

定为午宴拿出一并Chardonnay，但是我并没有精确指明我要拿哪一瓶，只是想要先细看这一类酒的库存情况。我打入如下的命令：

```
SELECT * FROM CELLAR WHERE WINE = 'Chardonnay'
```

计算机回答如下：

<u>BIN</u>	<u>WINE</u>	<u>PRODUCER</u>	<u>YEAR</u>	<u>BOTTLES</u>	<u>READY</u>	<u>COMMENTS</u>
2	Chardonnay	Buena Vista	79	1	82	
3	Chardonnay	Louis Martini	79	5	82	
6	Chardonnay	Chappellet	79	4	82	Thanksgiving

将这张表与上一次显示结果相比较，可以看出，由于在SELECT命令中规定了WHERE子句，我已使计算机只把满足WHERE子句中的条件的那些记录(那些行)输出。还可以要计算机只显示某些列，也可以只显示某些行与某些列的组合，如下例所示：

```
SELECT BIN, PRODUCER, READY, BOTTLES
FROM CELLAR
WHERE WINE='Chardonnay'
```

(不论命令打在一行中或象这里那样分开打成几行上，这并不重要)顺便说一句，在这个例子里我已经改变了各列从左到右排列的次序，目的是为了表明可以这样做。执行上述的命令的结果，显示：

<u>BIN</u>	<u>PRODUCER</u>	<u>READY</u>	<u>BOTTLES</u>
2	Buena Vista	82	1
3	Louis Martini	82	5
6	Chappellet	82	4

前例中的SELECT * 是“SELECT表中各列”的简写。

假设我决定要一瓶Louis Martini (bin 3), 为了反映我就要去地下室拿一瓶酒的这一事实。现在我应该更新数据库：

```
UPDATE CELLAR
SET BOTTLES=4 (原先是 5)
WHERE BIN=3
```

假若现在打入：

```
SELECT *
FROM CELLAR
WHERE BIN=3
```

就会看到我的UPDATE命令对数据库的影响:

<u>BIN</u>	<u>WINE</u>	<u>PRODUCER</u>	<u>YEAR</u>	<u>BOTTLES</u>	<u>READY</u>	<u>COMMENTS</u>
3	Chardonnay	Louis Martini	79	4		82

假若我改选(那唯一的)一瓶*Buena Vista*。那么,就完全没有必要再保存那些数据库记录了,而应把它们删除:

```
DELETE
FROM CELLAR
WHERE BIN=2
```

我必须时时完成对酒窖数据库的其他操作是:INSERT(插入)一个新的记录[或为增加BOTTLES数而UPDATE(更新)一个已有记录]以及反映新的收藏情况,例如:

```
INSERT
INTO CELLAR
VALUES (53, 'Pinot Noir', 'Franciscan', 76,1,83, 'present from Joan')
```

我们已经看到对数据的全部四种基本操作:检索(SELECT)、修改(UPDATE),删除(DELETE)和插入(INSERT)。这四种操作通常叫做数据操纵运算。如果你使用一个数据库系统,就会知道经常地要用到这些操作(尤其是检索)。

2.2 数据库系统带来的益处

为什么要使用一个数据库系统?它的优点是什么?酒窖的例子是这么小,这么简单,因而其好处并不明显。但试再为一个大餐馆设想一个数据库,这个餐馆存有数以千计的瓶酒,而且频繁地改变着库存量。或者想象一个酒库,其库存量很大,同时频繁地调入调出。在这种情况下,计算机化系统的优点,就会明显地超过传统的以帐本为基础的方法。其中的一些优点是:

- (1) 紧凑:不需要庞大的纸页式卷宗文件;

(2) **速度快**: 用计算机检索和更改数据比人工快(尤其是对于立即查询, 例如, 对“我们的Zinfandel比Pinot Noir更多吗? 这个问题能很快的回答, 这里不需要耗费时间的任何手工或视觉搜索);

(3) **减轻劳力**: 许多全然单调的手工维护文件的劳动被取消了。(机械的任务总是用机械去做更好);

(4) **即时性**: 任何时候都能够得到准确最新的情报。

还有一个不是显而易见的好处(酒窖这个例子实在太简单, 所以不容易看出来), 那就是, 设计数据库这种工作本身就是有好处的。事实上, 有时你感到很难应付手头的工作, 其原因是不容易对你必须处理的数据获得一个恰当的理解。为此, 你就需要有一种训练, 一套指南, 帮助你以有秩序的方式工作, 而设计数据库正好提供了这种训练。为了作出数据库的设计, 就迫使你系统地处理问题, 并常常帮助你恰当地组织你的思考。不过, 这些说法仅在数据库是“大型”的时候才适用, 一般地说, 有较大的容量和更复杂的数据结构。容量越大或数据结构越复杂——看下一节——就越需要这样的训练。当然, 在这样的环境中, 数据库系统的使用也变得特别有吸引力, 更必要。“小型”的数据库的结构往往简单直观。在这个意义上, 本书讨论的多数数据库都是“小型”的。有关数据库的设计和训练以及大型数据库将在第二编中讨论。

在多用户共享数据库中还有其他一些好处会表现得更明显。然而, 数据共享问题要等到第二编中才能讨论。

2.3 多重表的例子

酒窖存酒的例子特别简单, 因为它只包括一个表格, 大多数数据库是由多个互相联系的表格组成的。这里举一例(它仍很简单):

FLIGHTS 表:

FLIGHT	FROM-CODE	TO-CODE	DEP-TIME	ARR-TIME	CODE	CITY
TW750	SJC	STL	0745	1321	SJC	San Jose
DL834	STL	ORD	1725	1822	STL	St. Louis
AA160	SFO	ORD	1203	1750	ORD	Chicago
UA095	BOS	SFO	1005	1310	SFO	San Francisco
UA122	SFO	ORD	0730	1330	BOS	Boston
PA072	LAX	JFK	0830	1700	LAX	Los Angeles
AA214	SFO	ORD	1215	1820	JFK	New York
TW066	LAX	BOS	0845	1658		
EA224	LAX	JFK	0900	1649		

CITIES 表

FLIGHT	FROM-CODE	TO-CODE	DEP-TIME	ARR-TIME	CODE	CITY
TW750	SJC	STL	0745	1321	SJC	San Jose
DL834	STL	ORD	1725	1822	STL	St. Louis
AA160	SFO	ORD	1203	1750	ORD	Chicago
UA095	BOS	SFO	1005	1310	SFO	San Francisco
UA122	SFO	ORD	0730	1330	BOS	Boston
PA072	LAX	JFK	0830	1700	LAX	Los Angeles
AA214	SFO	ORD	1215	1820	JFK	New York
TW066	LAX	BOS	0845	1658		
EA224	LAX	JFK	0900	1649		

这里两个表格之间的互相联系是: 作为FROM-CODE或TO-CODE中的值出现在FLIGHT表中的每一城市的代码(SJC, STL等)也必定作为CODE的值出现在CITIES城市表中。假若我想找到所有的从San Francisco到Chicago的航班, 首先, 我可以在CITIES表中查找这两个城市的代码, 例如:

```
SELECT CODE
FROM CITIES
WHERE CITY='San Francisco'
```

结果(在荧光屏上)

CODE

SFO

现在查Chicago:

```
SELECT CODE
FROM CITIES
WHERE CITY='Chicago'
```

结果是:

CODE

ORD

现在我可以提出我真正的问题:

```
SELECT .
FROM FLIGHTS
WHERE FROM-CODE='SFO'
AND TO-CODE='ORD'
```

注意在WHERE子句中AND的使用中, 把两个条件结合起