第5章 数据思维——数据的组织、管理与挖掘

We are entering a new world in which data may be more important than software.

——Tim O'Reilly

(O'Reilly 媒体公司创始人兼 CEO，预言了开源软件、Web 2.0 等数次互联网潮流)

5.1 数据的组织和管理

第 2 章中，我们了解了数值、西文字符、汉字和多媒体信息在计算机中的数据表示和编码。本章主要讲述相互关联的数据的组织、管理和挖掘及面向数据组织和数据处理时的基本思维框架。本章内容的相关视频，读者可以参考中国大学视频公开课官方网站“爱课程”网(http://www.icourses.cn)河北工程大学“心连‘芯’的思维之旅”课程中的第五讲。

信息是对客观世界中各种事物的运动状态和变化的反映。数据是信息的一种载体，是信息的一种表达方式。在计算机中，信息是使用二进制进行编码的。数据是描述客观事物的数值、字符以及能输入机器且能被处理的各种符号集合。简而言之，数据就是计算机化的信息。

计算机程序是对信息(数据)进行加工处理。可以说，程序＝算法＋数据组织和管理，程序的效率取决于两者的综合效果。随着信息量的增大，数据的组织和管理变得非常重要，它直接影响程序的效率。

5.1.1 数据结构

数据结构是计算机存储、组织数据的方式。数据结构是指相互之间存在一种或多种特定关系的数据元素的集合。通常情况下，精心选择的数据结构可以带来更高的运行或者存储效率。数据结构往往与高效的检索算法和索引技术有关。

比如，一幅图像是由简单的数值组成的矩阵，一个图形中的几何坐标可以组成表，语言编译程序中使用的栈、符号表和语法树，操作系统中所用到的队列、树形目录等都是有结构的数据。

数据结构通常包括以下几方面。

(1) 数据的逻辑结构：由数据元素之间的逻辑关系构成。

(2) 数据的存储结构：数据元素及其关系在计算机存储器中的存储表示，也称为数据结构的物理结构。

(3) 数据的运算：施加在该数据上的操作。

1. 逻辑结构

数据的逻辑结构是从逻辑关系上描述数据，它与数据的存储无关，是独立于计算机的，因此数据的逻辑结构可以看作是从具体问题抽象出来的数学模型。数据的逻辑结构有两个要素，一是数据元素，二是关系。根据数据元素之间关系的不同，通常有四类基本结构：集合结构、线性结构、树形结构和图结构。

(1) 集合结构。

集合结构是指比较简单的数据，即少量、相互间没有太大关系的数据。比如，在进行计算某方程组的解时，中间的计算结果数据可以存放在内存中以便以后调用。在程序设计语言中，往往用变量来实现。

(2) 线性结构。

线性结构是指该结构中的数据元素之间存在一对一的关系。其特点是开始元素和终端元素都是唯一的，除了开始元素和终端元素以外，其余元素都有且仅有一个前驱元素，有且仅有一个后继元素。典型的线性结构有线性表、栈和队列。

① 线性表。

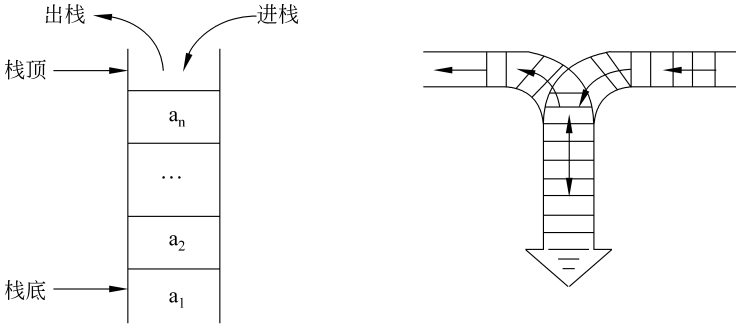
简单地说，线性数据是指同类的批量数据，也称线性表。比如，英文字母表(A,B,…,Z)，1000 个学生的学号和成绩，3000 个职工的姓名和工资，一年中的四个季节(春、夏、秋、冬)等。

② 栈。

如果对线性数据操作增加如下规定：数据的插入和删除必须在同一端进行，每次只能插入或删除一个数据元素，则这种线性数据组织方式就称为栈结构。通常将表中允许进行插入、删除操作的一端称为栈顶(Top)，表的另一端称为栈底(Bottom)。当栈中没有元素时称为空栈。

栈的插入操作被形象地称为进栈或入栈，删除操作称为出栈或退栈。

栈是先进后出的结构(First In Last Out，FILO)，如图 5.1(a)所示。日常生活中铁路调度就是栈的应用，如图 5.1(b)所示。



(a) 栈　　(b) 铁路调度

图 5.1　栈和栈的应用

比如，网络浏览器会将用户最近访问过的网址组织为一个栈。这样，用户每访问一个新页面，其地址就会被存放至栈顶；而用户每按下一次“后退”按钮，即可沿相反的次序访问此前刚访问过的页面。

类似地，主流的文本编辑器也大都支持编辑操作的历史记录功能(Ctrl＋z：撤销，Ctrl＋y：恢复)，用户的编辑操作被依次记录在一个栈中。一旦出现误操作，用户只需按下“撤销”按钮，即可取消最近一次操作并回到此前的编辑状态。

③ 队列。

如果对线性数据操作增加如下规定：只允许在表的一端插入元素，而在另一端删除元素，则这种线性数据组织方式就称为队列，如图 5.2 所示。

队列具有先进先出(Fist In Fist Out，FIFO)的特性。在队列中，允许插入的一端称为队尾，允许删除的一端则称为队头。

队列运算包括：入队运算——从队尾插入一个元素；退队运算——从队头删除一个元素。

日常生活中排队就是队列的应用。计算机及其网络自身内部的各种计算资源，无论是多进程共享的 CPU 时间，还是多用户共享的打印机，都需要借助队

列结构来实现合理和优化的分配。

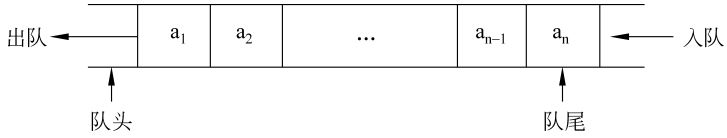


图 5.2　队

（3）树形结构。

如果要组织和处理的数据具有明显的层次特性，比如，家庭成员间辈分关系、一个学校的组织图，这时可以采用层次数据的组织方法，也形象地称为树形结构。

层次模型是数据库系统中最早出现的数据模型，是用树形结构来表示各类实体以及实体间的联系的。层次数据库是将数据组织成树形结构，并用“一对多”的关系联结不同层次的数据库。

严格地讲，满足下面两个条件的基本层次联系的集合称为树形数据模型或层次数据模型：

- 有且只有一个结点没有双亲结点，这个结点称为根结点；
- 根结点以外的其他结点有且只有一个双亲结点，如图 5.3 所示。

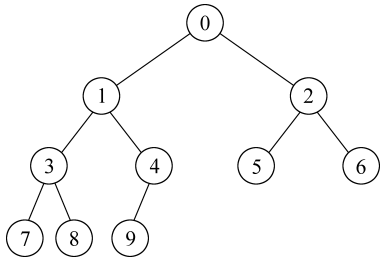


图 5.3　树形数据模型

在第 1 章的例 1-3 中，提到了国际象棋世界冠军“深蓝”。国际象棋、西洋跳棋与围棋、中国象棋一样都属于双人完备博弈。所谓双人完备博弈，就是两位选手对垒，轮流走步，其中一方完全知道另一方已经走过的棋步以及未来可能的走步，对弈的结果要么是一方赢（另一方输），要么是和局。

对于任何一种双人完备博弈，都可以用一个博弈树（与或树）来描述，并通过博弈树搜索策略寻找最佳解。博弈树类似于状态图和问题求解搜索中使用的搜索树。搜索树上的第一个结点对应一个棋局，树的分支表示棋的走步，根结点表示棋局的开始，叶结点表示棋局的结束。一个棋局的结果可以是赢、输或者和局。

树在计算机领域中也有着广泛的应用。例如，在编译程序中，用树来表示源程序的语法结构；在数据库系统中，可用树来组织信息；在分析算法的行为时，可用树来描述其执行过程。

（4）图结构。

有时，还会遇到更复杂一些的数据关系，满足下面两个条件的基本层次联系

的集合称为图状数据模型或网状数据模型：

- 允许有一个以上的结点无双亲；
- 一个结点可以有多于一个的双亲。

比如，在第 4 章的例 4-1 国际会议排座位问题中，可以将问题转化为在图 G 中找到一条哈密顿回路的问题。

【例 5-1】 哥尼斯堡七桥问题。

1736 年，29 岁的欧拉向圣彼得堡科学院递交了《哥尼斯堡的七座桥》的论文，在解答问题的同时，开创了数学的一个新的分支——图论与几何拓扑。

哥尼斯堡七桥问题是：17 世纪的东普鲁士有一座哥尼斯堡城，城中有一座奈佛夫岛，普雷格尔河的两条支流环绕其旁，并将整个城市分成北区、东区、南区和岛区 4 个区域，全城共有 7 座桥将 4 个区域相连起来。

人们常通过这 7 座桥到各城区游玩，于是产生了一个有趣的数学难题：寻找走遍这 7 座桥，且只许走过每座桥一次，最后又回到原出发点的路径。该问题就是著名的“哥尼斯堡七桥问题”，如图 5.4 所示。

欧拉抽象出问题最本质的东西，忽视问题非本质的东西（如桥的长度等），把每一块陆地考虑成一个点，连接两块陆地的桥以线表示，并由此得到了如图 5.5 所示的几何图形。

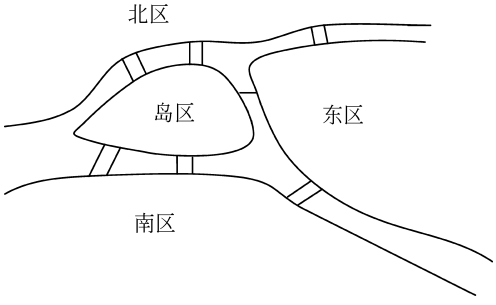


图 5.4 哥尼斯堡七桥问题

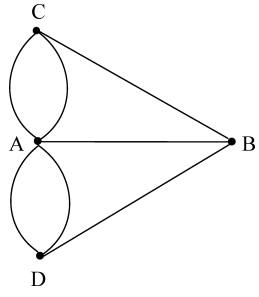


图 5.5 赋权图示例

若我们分别用点 A、B、C、D 表示为哥尼斯堡的 4 个区域，这样著名的“哥尼斯堡七桥问题”便转化为是否能够用一笔不重复地画出过此七条线的一笔画问题了。

欧拉不仅解决了此问题，且给出了一笔画问题的必需条件：图形必须是连通的；途中的奇点个数是 0 或 2（奇点是指连到一点的边的数目是奇数条）。

由于“哥尼斯堡七桥问题”中 4 个点全是奇点，因此不能一笔画出，也就是不存在不重复地通过所有桥的路径。

欧拉的论文为图论的形成奠定了基础。图论是对现实问题进行抽象的一个强有力的数学工具，已广泛地应用于计算、运筹学、信息论、控制论等学科。

在实际应用中，有时图的边或弧上往往与具有一定意义的数有关，即每一条边都有与它相关的数，称为权，这些权可以表示从一个顶点到另一个顶点的距离或耗费等信息。我们将这种带权的图称为赋权图或网，如图 5.6 所示。

可以利用算法求出图中的最短路径、关键路径等，因此图可以用来解决多类问题，如电路网络分析、线路的铺设、交通网络管理、工程项目进度安排、商业活动安排等。图是一种应用极为广泛的数据结构。

网状模型与层次模型的区别在于：网状模型允许多个结点没有双亲结点；网状模型允许结点有多个双亲结点；网状模型允许两个结点之间有多种联系（复合联系）；网状模型可以更直接地去描述现实世界；而层次模型实际上是网状模型的一个特例。

2. 存储结构

数据对象在计算机中的存储表示称为数据的存储结构，也称为物理结构。把数据对象存储到计算机时，通常要求既要存储各数据元素的数据，又要存储数据元素之间的逻辑关系，数据元素在计算机中用一个结点来表示。数据元素在计算机中有两种基本的存储结构，分别是顺序存储结构和链式存储结构。

（1）顺序存储结构。

顺序存储结构是借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系，通常借助程序设计语言的数组类型来描述，如图 5.7 所示。

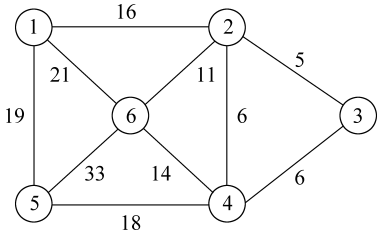


图 5.6　赋权图示例

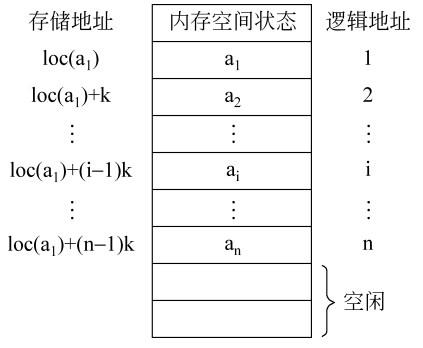


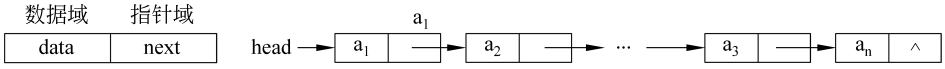
图 5.7　顺序存储

在此方式下，每当插入或删除一个数据时，该数据后面的所有数据都必须向后或向前移动。因此，这种方式比较适合于数据相对固定的情况。

（2）链表存储结构。

顺序存储结构要求所有的元素依次存放在一片连续的存储空间中，而链表存储结构，无须占用整块存储空间，但为了表示结点之间的关系，需要给每个结点附加指针字段，用于存放后继元素的存储地址，所以，链表存储结构通常借助程序设计语言的指针类型来描述。

在链表存储结构中，每个结点由两部分组成：一部分用于存储数据元素的值，称为数据域；另一部分用于存储指针，称为指针域，用于指向该结点的前一个或后一个结点（即前件或后件）。对于最后一个数据，就填上一个表示结束的特殊值，这种像链条一样的数据组织方法就称为链表存储结构。头指针 head 指向第一个结点，最后一个结点的指针域为“空”（NULL），如图 5.8 所示。



(a) 结点结构　　(b) 一个非空的线性链表示意图

图 5.8　链表存储结构

【例 5-2】　线性表（A，B，C，D，E，F，G）的单链表存储结构如图 5.9 所示，整个链表的存取需从头指针开始进行，依次顺着每个结点的指针域找到线性表的各个元素。

在此方式下，每当插入或删除一个数据时，可以方便地通过修改相关数据的位置信息来完成。因此，这种方式比较适合于数据相对不固定的情况。

头指针 Head 位置：16

存储地址	数据域	指针域
1	D	55
8	B	22
22	C	1
37	F	25
16	A	8
25	G	NULL
55	E	37

图 5.9　线性表（A，B，C，D，E，F，G）的单链表存储结构

知行合一

在链表存储结构中，每个数据都增加了存放位置信息的空间，所以是靠空间来换取数据频繁插入和删除等操作时间的设计，这种空间和时间的平衡问题是计算机的算法和方法设计经常要考虑的问题。

3. 数据结构与算法

数据结构与算法之间存在着密切的关系。可以说，不了解施加于数据上的算法需求，就无法决定数据结构；反之，算法的设计和选择又依赖于作为其基础的数据结构。即数据结构为算法提供了工具，算法是利用这些工具来解决问题的最佳方案。

（1）数据结构与算法的联系。

数据结构是算法实现的基础，算法总是要依赖于某种数据结构来实现的。算法的操作对象是数据结构。算法的设计和选择要结合数据结构，简单地说，数据结构的设计就是选择存储方式，如确定问题中的信息是用普通的变量存储还是用其他更加复杂的数据结构。算法设计的实质就是为实际问题要处理的数据选择一种恰当的存储结构，并在选定的存储结构上设计一个好的算法。不同的数据结构将导致差异很大的算法。数据结构是算法设计的基础。算法设计必须考虑到数据结构，算法设计是不可能独立于数据结构的。另外，数据结构的设计和选择需要为算法服务。如果某种数据结构不利于算法实现，它将没有太大的实际意义。知道某种数据结构的典型操作才能设计出好的算法。

总之，算法的设计同时伴有数据结构的设计，两者都是为最终解决问题服务的。

（2）数据结构与算法的区别。

数据结构关注的是数据的逻辑结构、存储结构以及基本操作，而算法关注更多的是如何在数据结构的基础上解决实际问题。算法是编程思想，数据结构则是这些思想的逻辑基础。

5.1.2 文件系统和数据库

5.1.2.1 文件系统

在较为复杂的线性表中，数据元素（Data Elements）可由若干数据项组成，由若干数据项组成的数据元素称为记录（Record），由多个记录构成的线性表称为文件（File）。

以文件方式进行数据组织和管理，一般需要进行文件建立、文件使用、文件删除、文件复制和移动等基本操作，其中文件使用必须经过打开、读、写、关闭这4个基本步骤。程序设计语言一般都提供了文件管理功能。

5.1.2.2 数据库系统

如果数据量非常大，关系也很复杂，这时可以考虑使用数据库技术来组织和

管理。

数据管理技术是在20世纪60年代后期开始的，经历了人工管理、文件管理、数据库系统三个阶段，与前两个阶段相比，数据库系统具有以下特点。

(1) 数据结构化。

在数据库系统中的数据是面向整个组织的，具有整体的结构化。同时存取数据的方式可以很灵活，可以存取数据库中的某一个数据项、一组数据项、一个记录或者一组记录。

(2) 共享性高、冗余度低、易扩充。

数据库系统中的数据不再面向某个应用，而是面向整个系统，因而可以被多个用户、多个应用共享使用。使用数据库系统管理数据可以减少数据冗余度，并且数据库系统弹性大，易于扩充，可以适用各种用户的要求。

(3) 数据独立性高。

数据独立性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。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是相互独立的，即使数据的逻辑结构改变了，用户程序也可以不变。

利用数据库系统，可以有效地保存和管理数据，并利用这些数据得到各种有用的信息。

1. 数据库系统概述

数据库系统主要包括数据库(DataBase，DB)和数据库管理系统(DataBase Management System，DBMS)等。

(1) 数据库。

数据库是长期存储在计算机内的、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、描述和存储，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。

(2) 数据库管理系统。

数据库管理系统具有建立、维护和使用数据库的功能；具有面向整个应用组织的数据结构，高度的程序与数据的独立性，数据共享性高、冗余度低、一致性好、可扩充性强、安全性和保密性好、数据管理灵活方便等特点；具有使用方便、高效的数据库编程语言的功能；能提供数据共享和安全性保障。

数据库系统包括两部分软件：应用层与数据库管理层。

应用层负责数据库与用户之间的交互，决定整个系统的外部特征，例如，可以采用问答或者填写表格的方式与用户交互，也可以采用文本或图形用户界面的方式等。

数据库管理层负责对数据进行操作，例如数据的添加、修改等，是位于用户与操作系统之间的一层数据管理软件，主要有以下几个功能。

- 数据定义功能：提供数据定义语言，以便对数据库的结构进行描述。
- 数据操纵功能：提供数据操纵语言，用户通过它实现对数据库的查询、插入、修改和删除等操作。
- 数据库的运行管理：数据库在建立、运行和维护时由 DBMS 统一管理、控制，以保证数据的安全性、完整性、系统恢复性等。
- 数据库的建立和维护功能：数据库的建立、转换，数据的转储、恢复，数据库性能监视、分析等，这些功能需要由 DBMS 完成。

(3) 数据库管理员。

数据库与人力、物力、设备、资金等有形资源一样，是整个组织的基本资源，具有全局性、共享性的特点，因此对数据库的规划、设计、协调、控制和维护等需要专门人员来统一管理，这些人员统称为数据库管理员。

2. 数据模型

各个数据以及它们相互间关系称为数据模型。在结构上数据库主要有 4 种数据模型，即层次、网状、关系和面向对象模型。

关系模型是 1970 年 IBM 公司的研究员 E.F.Codd 首次提出的，是目前最重要的一种数据模型，它建立在严格的数学概念基础上，具有严格的数学定义。20 世纪 80 年代以来推出的数据库管理系统几乎都支持关系模型，关系数据库系统采用关系模型作为数据的组织方式。关系数据模型应用最为广泛，例如 SQL Server、MySQL、Oracle、Access、Sybase、Excel 等都是常用的关系数据库管理系统。

关系模型是对关系的描述，由关系名及其所有属性名组成的集合，格式为：关系名(属性 1，属性 2，……)，比如，表 5.1 的学生成绩管理(学号，姓名，高数，英语，计算机)等。

在关系模型中，数据的逻辑结构实际上就是一个二维表，它应具备如下条件。

① 关系模型要求关系必须是规范化的。最基本的一个条件是，关系的每一个分量必须是不可分的数据项。

② 表中每一列的名称必须唯一，且每一列除列标题外，必须有相同的数据类型。

③ 表中不允许有内容完全相同的元组(行)。

④ 表中的行或列的位置可以任意排列，并不影响所表示的信息内容。