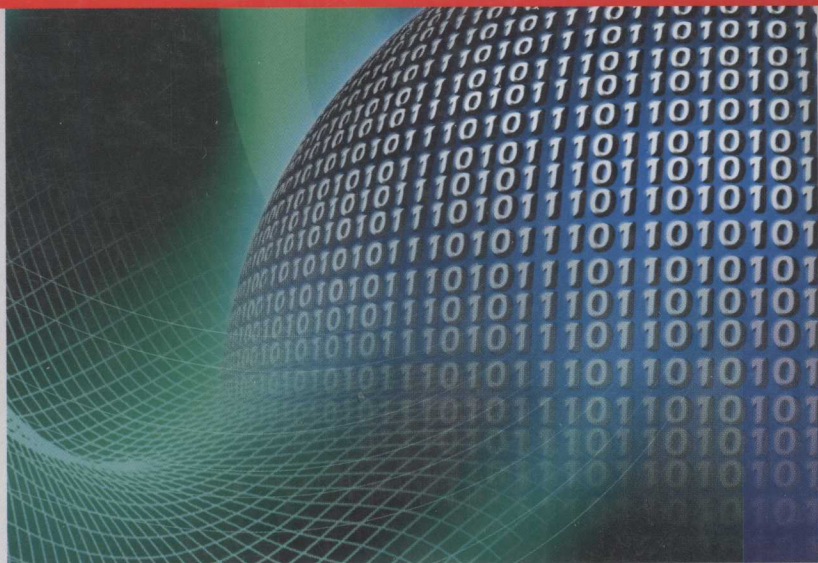


主动数据库系统 理论基础

郝忠孝 著



科学出版社
www.sciencep.com

主动数据库系统理论基础

郝忠孝 著

科学出版社

北 京

内 容 简 介

本书是在作者三十余年来对主动数据库系统理论研究的基础上撰写的。书中系统论述和分析了主动数据库系统理论以及若干新的概念、方法和算法。

本书共分二十章。主要内容包括：主动数据库管理系统的体系结构、主动规则的相关模型及说明语言、事件监测，特别重点讨论了基于触发图、活化图、惰化图、事务、规则优先级、活化路径、代数等方法对主动规则集终止性、汇流性的静态和动态分析，较详细地给出了相关的定理、方法和算法及算法证明，深入地讨论了规则执行和监测、主动数据库完整性等。

本书可作为计算机科学与技术学科、控制理论与控制工程学科等相关专业的高年级本科生教材或硕士生选修课教材，也可供从事上述领域研究的博士生、科研人员及工程技术人员等参考。

图书在版编目 (CIP) 数据

主动数据库系统理论基础/郝忠孝 著. —北京：科学出版社，2009
ISBN 978-7-03-023364-6

I. 主… II. 郝… III. 数据库管理系统-理论 IV. TP311.13

中国版本图书馆 CIP 数据核字 (2008) 第 175218 号

责任编辑：耿建业/责任校对：陈玉凤

责任印制：赵 博/封面设计：耕者设计工作室

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100717

<http://www.sciencep.com>

新蕾印刷厂印刷

科学出版社发行 各地新华书店经销

*

2009 年 1 月第 一 版 开本：B5 (720×1000)

2009 年 1 月第一次印刷 印张：24

印数：1—3 000 字数：481 000

定价：68.00 元

(如有印装质量问题，我社负责调换<新蕾>)

前 言

数据库技术是在 20 世纪 60 年代末作为数据管理的最新技术登上数据处理舞台的。随着计算机应用的不断扩大,计算机硬件快速发展,数据库技术也得到了迅速的发展。数据库技术和计算机网络技术已成为当今世界计算机应用中两个最重要的基础领域。经过四十多年的发展,以数据模型的进展、变化为主线,出现了以层次模型和网状模型为代表的层次数据库和网状数据库的第一代数据库。70 年代末出现了以关系数据模型为代表的第二代数据库——关系数据库。80 年代以来,由于非传统应用领域的不断扩大,针对一些特殊领域的应用提出了许多新的数据模型和许多新的数据管理要求功能,由于传统数据库不具备这种能力,因此出现了以面向对象数据库为代表的新一代数据库系统。工程数据库、空间数据库、时空数据库、多媒体数据库、时态数据库、空值数据库、无环数据库等支持这些数据库的数据模型都是基于关系数据模型的扩充或者是面向对象模型。

传统数据库系统只能被动地按照用户明确给出的请求执行相应的操作,完成某个事务,因此传统数据库系统是被动的。数据库状态的改变是外界或用户程序影响的结果,也就是所有的查询和数据处理操作必须通过人工操作完成。为了实现数据完整性和一致性的自动维护以及满足实时信息处理的需要,要求数据库系统通过主动规则或触发器的形式扩充传统数据库,作出实时响应,出现了主动数据模型和主动数据库管理系统。1976 年,美国的 Eswaran 在“Specifications, Implementations and Interactions of a Trigger Subsystem in an Integrated Database System”一文中首先提出了触发的概念。1983 年, Morgenstern 在“Active Databases as a Paradigm for Enhanced Computing Environments”一文中首次提出了“主动数据库”这个术语。80 年代后期,有关主动数据库的大量论文开始出现,主动数据库开始成为数据库系统领域研究的一个热点。

数据库的应用分为两大类:①面向外部的应用。面向外部的一些应用领域,如航空航天、军事系统、重要的生产过程控制系统等严格要求实时性、安全性、可靠性、交互性和监测、报警等功能,提供主动性功能的主动数据库系统则提供了这种功能。除此之外,诸如工业控制系统、工作流管理、仓库、生物信息管理等众多领域都充分利用了主动规则机制。②面向数据库本身的应用主要有:数据库的完整性控制、安全性控制、导出数据处理、面向对象数据库模型中继承机制的定义与应用、性能测度等。不仅如此,就是数据库系统本身系统功能的监测与恢复也要求主动性功能以便及时得到监测和系统功能的恢复。

主动数据库系统从任意一个状态开始,经过有限步规则的执行过程是否可以终止?这就是主动数据库的可终止性问题。

规则执行时,若有多个规则同时被触发,哪个规则首先被选择执行?数据库的最终状态是否取决于规则被选择执行的先后顺序?这就是汇流性问题。

当一个规则的动作是数据检索或事务回退操作,我们就称规则的这个动作是可观察的。如果规则执行时有多个规则同时被触发,当多个规则被选择执行的顺序对可观察动作的结果不产生影响,就称这个规则集是可观察确定的,这就是可观察确定性。

具有可终止性、汇流性、可观察确定性是保证主动规则集具有良好行为特性的三个重要特征,因为规则集的无限循环执行会导致系统大量资源的浪费,从而导致系统性能恶化,而对一些要求严格的控制系统则会对系统产生致命的损害。主动规则集的可终止性判定是一个直接影响主动数据库设计和应用的关键问题之一。另外一个对系统造成致命损害的关键问题是主动规则集的执行汇流性。一般来说,在确保汇流性的情况下,可观察确定性可能是确定的。由于主动规则集描述上的无结构性导致它的可终止性成为数据库界的一个著名的不确定问题。而具有可终止性、汇流性是保证主动规则集具有良好行为特性的重要特征。因此,本书重点和较为深入地介绍规则集的可终止性判定和汇流性分析的理论问题。

本书出版的目的之一是通过主动数据库中主动规则集的可终止性、汇流性分析理论的论述,提供进一步研究的理论基础。打破主动规则集行为难以预测的瓶颈,使主动规则的语义表达更丰富、更充分。促进主动数据库在当前各个应用领域中得到更充分、更广泛、更可靠的应用。另一个目的是作者从事数据库理论研究三十余年,作为一种责任想把研究的一些结果留给年轻的同志,使有兴趣的同行和读者进入到这一领域。

本书以主动数据库管理系统的体系结构、特征为主线,力求用通俗易懂的语言来较为全面、系统地介绍以下内容:主动规则集的可终止性的静态、动态分析、判定方法;规则执行时的汇流性分析、判定方法;主动数据库中的依赖关系;自依赖规则的判定理论;各种规则执行模式和事务、规则调度方法;主动数据库的完整性等的相关概念、定义、引理、定理、算法及相应的证明。本书力求做到条理清晰、逻辑性强、易于理解。但是,由于本书是国内首部系统阐述主动数据库系统理论方面的著作,有些理论尚未来得及去实践,有些还有待于进一步的发展,加之作者水平有限,书中难免有不妥之处,敬请同行和读者批评指正,以便今后改正和不断发展。

本书可作为计算机科学与技术学科、控制理论与控制工程学科等相关专业的高年级本科生或硕士生选修课教材,也可供从事上述领域研究的博士生、科研人员及工程技术人员等参考。

本书的出版得到了科学出版社的大力支持和帮助，在此表示衷心的感谢。

在本书的写作过程中，我的博士生熊中敏、任超等为本书的出版做了大量有益的工作，特别是博士生李博涵、李松对全书进行了校对，并由李博涵绘制了本书的插图，在此表示真挚的谢意。

作 者

2008 年 10 月于哈尔滨

目 录

前言

第 1 章 主动数据库系统概论	1
1.1 基础知识	1
1.1.1 主动规则	2
1.1.2 事务	3
1.1.3 事务历史查询	4
1.1.4 事件和事件表达式	5
1.1.5 事件的消耗模式和事件的组合	7
1.1.6 规则库与事件库	8
1.1.7 规则的粒度和耦合模式	9
1.2 几个主动数据库原型系统	11
1.2.1 基于关系数据模型的主动数据库系统	11
1.2.2 基于面向对象数据模型的主动数据库系统	15
小结	17
第 2 章 主动数据库管理系统的体系结构	19
2.1 主动数据库管理系统的特性	19
2.1.1 E-C-A 规则定义特性	19
2.1.2 E-C-A 规则执行特性	21
2.1.3 ADBMS 可用性和应用特性	22
2.2 主动数据库管理系统的体系结构	23
2.2.1 辅助工具集成环境	24
2.2.2 执行主动功能的部件	25
2.2.3 数据存储部件	26
2.3 主动数据库管理系统的实现途径	27
小结	28
第 3 章 主动规则的相关模型和规则说明语言	29
3.1 主动规则的知识模型	29
3.1.1 事件描述范畴	30
3.1.2 条件描述范畴	33
3.1.3 动作描述范畴	34

3.2	主动规则运行模型	35
3.3	主动规则管理模型	38
3.3.1	规则描述方式和操作	38
3.3.2	主动数据模型	39
3.4	主动规则管理器和事件管理器结构	40
3.4.1	主动规则管理器	40
3.4.2	事件管理器	41
	小结	43
第4章	规则说明语言	44
4.1	规则说明语法	44
4.1.1	词法的约定	44
4.1.2	E-C-A 规则语法	45
4.2	规则说明	45
4.2.1	时间说明和操作模式说明	45
4.2.2	规则优先级说明	46
4.2.3	事件说明和条件说明	47
4.2.4	事件参数和系统参数说明	47
4.2.5	逻辑表达式和方法调用说明	48
4.2.6	动作说明	49
4.3	逻辑事件说明	51
4.3.1	逻辑事件的语义和逻辑条件说明	52
4.3.2	逻辑事件在 E-C-A 规则中的语义和规范说明	53
4.3.3	逻辑事件的上下文和参数中的上下文信息	54
	小结	55
第5章	复合事件监测	56
5.1	事件的复合操作	56
5.2	利用事件图监测复合事件	60
5.2.1	事件图	60
5.2.2	事件图复合事件的监测算法	62
5.3	约束环境下事件监测	66
5.3.1	约束环境	66
5.3.2	顺序环境下复合事件监测算法	67
5.3.3	最近环境下复合事件监测算法	71
5.4	约束环境下事件监测举例	75
5.4.1	Sequence 操作符和 AND 操作符	76

5.4.2	OR 操作符和 NOT 操作符	76
5.4.3	非周期操作符和 Plus 操作符	77
5.4.4	周期操作符	78
5.5	复杂条件及其评价	79
小结		84
第 6 章	基于图的主动规则集终止性静态分析	85
6.1	规则分析主动规则的三个特性	85
6.1.1	主动规则集分析	86
6.1.2	在编译阶段执行的主动规则集可终止性静态分析	87
6.2	有向图环路检测算法	89
6.3	规则执行图	91
6.4	基于触发图和活化图的终止性分析	93
6.4.1	TG 的建立方法	94
6.4.2	触发图的终止性分析定理	95
6.4.3	基于活化图的终止性分析	98
6.5	基于触发图和活化图的规则基本归约算法	100
6.6	基于关联图 G 的终止性分析	102
小结		108
第 7 章	基于事务的规则终止性分析	109
7.1	基于进化图 EG 的规则终止性分析	111
7.1.1	主动规则与程序和事务执行语义	111
7.1.2	抽象状态	112
7.1.3	进化图 EG 和创建算法	112
7.1.4	进化图 EG 的规则终止性分析	118
7.2	利用事务进行规则终止性分析	119
7.2.1	创建精确进化图 REG 算法	120
7.2.2	检验终止性	122
7.2.3	两种分析方法之间的关系	123
小结		125
第 8 章	带有规则优先级的终止性分析	127
8.1	数据模型和核心规则	127
8.1.1	主动数据库的语义维度	127
8.1.2	数据模型和核心规则	129
8.1.3	规则的执行语义	133
8.2	主动/演绎的基本转换	135

8.2.1	Datalog 及其扩展	135
8.2.2	核心规则到逻辑规则的转换	136
8.2.3	转换图	137
8.3	终止性分析	138
8.3.1	CORE ⁺ 向 Datalog 的转换	138
8.3.2	CORE ⁺ * 向 Datalog ⁺ * 的转换	139
小结		144
第 9 章	基于代数法的规则终止性分析	145
9.1	代数传播算法	145
9.1.1	代数运算符	145
9.1.2	代数传播算法	146
9.2	传播算法的传播规则	149
9.3	E-C-A 规则和 C-A 规则的代数语言	154
9.3.1	E-C-A 规则的代数语言	154
9.3.2	C-A 规则的代数语言	156
9.4	C-A 规则的活化关系分析	156
小结		160
第 10 章	基于活化路径的分析方法	161
10.1	分析的基础	161
10.1.1	可达概念的分析	161
10.1.2	活化路径和活化路径集	163
10.2	基于活化路径和同步关系的分析方法	164
10.2.1	活化路径同步执行对 TG 环执行的影响	164
10.2.2	有效活化路径	170
10.2.3	算法描述及分析	171
10.3	相关条件公式的建立	173
10.3.1	TG 环的执行序列建立条件公式	173
10.3.2	基于活化路径的条件公式	178
10.4	基于活化路径和条件公式的分析方法	181
10.4.1	禁止活化规则的判定定理	181
10.4.2	终止性判定算法描述及分析	183
小结		185
第 11 章	计算不可归约规则集的算法	186
11.1	在运行阶段执行的主动规则集可终止性动态分析	186
11.2	归约算法的分析	187

11.3	只含独立型触发环的主动规则集的归约算法	190
11.4	含有非独立型触发环的主动规则集的归约算法	192
11.4.1	算法的理论基础	192
11.4.2	算法描述及分析	193
小结		197
第12章	监测规则集的优化算法	198
12.1	监测规则集的相关知识	198
12.1.1	不可归约规则集中非终止规则子集的格结构	198
12.1.2	运行阶段规则集不可终止的监测方法	201
12.2	环监测程序	203
12.3	计算监测规则集的现有算法的分析	205
12.4	计算监测规则集的优化算法	207
小结		210
第13章	最小环的结构和监测的执行状态的化简	212
13.1	最小环的结构分析	212
13.2	最小环所监测的执行状态的表示方法	213
13.2.1	已有表示方法的分析	213
13.2.2	一种新的表示方法及其正确性证明	214
小结		219
第14章	主动规则集汇流性分析和可观察的确定性	220
14.1	基于执行图的汇流性分析	220
14.1.1	规则可交换性	220
14.1.2	汇流性分析	221
14.1.3	规则集汇流性判定算法	227
14.2	局部汇流	228
14.2.1	局部汇流	228
14.2.2	局部汇流分析算法	232
14.3	基于代数法的汇流性分析	234
14.3.1	C-A 规则的可交换性分析	236
14.3.2	E-C-A 规则的可交换性分析	236
14.4	可观察的确定性	237
小结		238
第15章	主动数据库中的依赖关系	239
15.1	主动数据库中的依赖关系定义及分类	239
15.2	属性依赖	240

15.2.1	逻辑型属性依赖	242
15.2.2	计算型属性依赖	243
15.2.3	匹配过程	244
15.2.4	PATH 路径及匹配执行算法	245
15.3	路径定义的冲突与终止性	250
15.4	存在型属性依赖	255
15.4.1	表示方法和终止性	255
15.4.2	汇流性	258
小结		263
第 16 章	规则依赖和事务依赖	264
16.1	规则依赖的分类	264
16.2	自依赖规则的判定算法	265
16.2.1	自依赖规则和规则依赖图的关系	265
16.2.2	规则依赖图中规则结点二叉树的构造过程	267
16.2.3	自依赖规则的判定理论	268
16.2.4	自依赖规则的生成树判定算法	273
16.3	事务依赖	277
16.3.1	依赖事务集	278
16.3.2	嵌套事务的结构依赖	279
16.3.3	在事务闭包中可能存在的事务依赖关系	280
16.3.4	隐含的事务依赖关系	282
小结		283
第 17 章	规则执行	285
17.1	规则的执行冲突	285
17.1.1	规则冲突的消解	285
17.1.2	规则冲突的种类	288
17.2	冲突图和规则执行全序序列	290
17.3	优先级图和执行图的关系	294
17.4	扩展执行图	299
17.5	规则行为的并行调度算法	301
17.6	调度算法的并行性能讨论	307
17.7	规则库不一致性检测	308
小结		312
第 18 章	基于嵌套事务的规则并行执行模型	313
18.1	嵌套事务模型	313

18.2 事件历史及其投影	314
18.2.1 实体事件	314
18.2.2 事件历史及投影	315
18.3 事务的可见性	317
18.3.1 事务对数据对象的影响	317
18.3.2 事务的可见性	317
18.4 嵌套事务耦合方式	320
小结	322
第 19 章 嵌套事务规则的并行控制和死锁检测	324
19.1 一般事务处理	324
19.2 规则事务结构	330
19.3 并行控制算法	332
19.4 死锁检测恢复	336
小结	340
第 20 章 主动数据库的完整性	341
20.1 完整性约束	341
20.1.1 完整性约束条件	341
20.1.2 完整性控制	343
20.1.3 完整性控制部件的产生的过程	344
20.2 约束表达式与语言	345
20.2.1 约束表达式	345
20.2.2 约束语言	347
20.3 约束集分解	350
20.3.1 约束简化	350
20.3.2 集合简化	354
20.4 冗余约束与非一致约束	357
20.4.1 冗余约束和非一致约束的描述	357
20.4.2 冗余约束和非一致约束的检测算法	358
20.5 约束规则的生成	362
20.5.1 完整性约束规则表示	362
20.5.2 校正处理	364
小结	365
参考文献	367

第 1 章 主动数据库系统概论

在主动数据库的研究中,已经出现了多种不同的系统和模型。但是,什么是主动数据库系统?什么是主动数据库管理系统?在什么情况下,我们可以说一个管理系统是“主动”的。抽象地说,一个主动数据库系统是由主动数据库管理系统与一个具体的数据库构成。主动数据库系统能对数据库的情形自动地进行反应,并能指定系统的反应行为。但这种定义不够精确,而且,对“主动”一词的解释也没有广泛的一致看法。简单地说,主动数据库系统(ADBS)就是将“被动的”数据库系统扩展成具有反应行为(reactive behavior)功能的数据库系统。从功能的角度来讲,一个主动数据库系统是由一个传统的数据库系统和一个事件驱动的知识库以及相应的事件监测模块组成,形式化地描述为

$$\text{ADBS} = \text{DBS} + \text{EB} + \text{EM}$$

其中,DBS是用来存储、维护、管理数据的传统数据库系统;EB是一个由事件驱动的知识库,其中每一项知识表示在相应的事件发生时,如何(何时、何地)来主动地执行用户预先定义的动作;EM是在数据库应用程序运行的过程中,监测数据库的状态变化,一旦EB中定义的事件发生时就主动地触发系统,按EB中指定的相应知识执行其中预先定义的动作,从而实现主动功能。由此可见,主动数据库的知识库(或规则库)是实现主动功能的关键,EB中知识表示不同,也就决定了不同的主动功能的实现。主动数据库的主要设计思想是要用一种统一而方便的机制来实现对应用主动性功能的需求,即使得系统能用统一的方法把各种主动服务功能与数据库系统集成在一起,利于软件的模块化和软件重用,同时也增强了数据库系统的自我支持能力。

在当前的主动数据库中,知识大多数都采用由事件驱动的“事件—条件—动作”形式的规则来表示,所以又简称E-C-A规则。

1.1 基础知识

本书将在第2章中描述一个主动数据库管理系统应有的特性。为了更好地理解和掌握这些特性和本书的后面的内容,为此在介绍主动数据库管理系统的特性之前,先简单介绍一些相关的概念。

1.1.1 主动规则

E-C-A 规则 (event-condition-action) 是主动数据库系统中的核心概念。

定义 1.1 主动规则由事件、条件和动作组成。记为 E-C-A 或 ECA。

一个主动规则具有三个组成成分：事件 (event) 既可以是数据操作事件 (数据库系统内部的事件) 也可以是系统外部反馈给系统的事件。条件 (condition) 就是对当前数据库状态的一个请求，通常表达为谓词、数据库查询语句。动作 (action) 通常表示为一组数据库更新操作或包含一组数据库更新操作的过程。

规则的基本运作方式是，一旦系统检测到相应规则事件发生，就在特定时刻检查规则的条件，若条件满足，则执行相应的动作。除了三要素之外，主动规则还包含一些基本语义说明，如优先级、规则耦合方式等，合称为规则属性。规则属性决定着系统对规则三要素的不同处理方式，如何时检查规则条件等。一般情况下规则定义如下：

```
define rule <rule_name>
  event   <event_clause>
  condition <condition>
  action   <action>
  coupling mode (<coupling>, <coupling>)
  priorities (before | after) <rule_name>
  interrupt <interrupt>
  interruptible (<interruptible, interruptible>)
```

在定义了规则后，主动数据库系统监视相关的事件。当监测到相关的事件发生时，系统通知负责处理规则执行的组件，来处理规则条件的评价和规则动作的执行。主动数据库管理系统提供规则定义语言 (rule definition language) 来定义 E-C-A 规则，用户可以用该语言来指定规则的事件、条件和动作。

规则触发后，系统需要确定规则在何时开始执行以及规则执行时应当有什么样的属性，我们称为规则的执行模型 (execution model)。

一般情况下，事件发生在事务内，规则也在事务内执行。若一个事件在事务内发生并且触发了规则，则该事务称为触发事务 (triggering transaction)；负责规则执行的事务称为被触发事务 (triggered transaction)。

执行模型确定触发事务和被触发事务的提交和夭折依赖关系，以及规则执行的并发控制和恢复。常用来描述触发事务和被触发事务间关系的框架结构是嵌套事务模型。

1.1.2 事务

定义 1.2 事务是用户定义的一个数据库操作序列，这些操作要么全做要么全不做，是一个不可分割的工作单位。

例如，在关系数据库中，一个事务可以是一条 SQL 语句、一组 SQL 语句或整个程序。

事务的开始和结束可以由用户显式控制。如果用户没有显式地定义事务，则由数据库关系系统按缺省规定自动划分事务。在 SQL 语言中，定义事务的语句有三条：

begin transaction

commit

rollback

事务通常是以 begin transaction 开始，以 commit 或 rollback 结束。

commit 表示提交，即提交事务的所有操作。具体地说就是将事务中所有对数据库的更新写回到磁盘上的物理数据库中去，事务正常结束。rollback 表示回滚，即在事务运行的过程中发生了某种故障，事务不能继续执行，系统将事务中对数据库的所有已完成的操作全部撤销，回滚到事务开始时的状态。这里的操作指对数据库的更新操作。

对于一个事务具有下列性质：原子性、持久性、一致性、隔离性。

1. 原子性

所谓原子性就是不可分割的意思。在事务管理中，我们强调最多的就是原子性。因为原子性是用来描述事务的不可分割性，换句话说，事务的操作要么全部执行要么全部不执行而不存在部分被执行的问题。而这种原子性的要求就会产生下面的情况，当一个事务由于故障而中断时，它的部分结果同时也被取消。

一般来说，导致事务无法执行的原因一般有两个，即事务中止和系统故障。当一个事务的输入有错误时，该程序将无法继续运行，因此就会提出中止请求，这样可以产生一个事务的中断；如果事务本身是有错误的，那么在有些执行过程中，它是无法完成的，这样也会产生事务的中断。事务的中止也可以因与系统有关的原因而由系统来强迫中止，典型例子为系统过载和死锁。在出现事务中止时保证其原子性的措施叫做事务恢复，而在系统故障时保证原子性的措施叫做故障恢复。

完成一个事务叫做事务提交。如果要修改数据库，唯一的条件是出现事务提交。假定每个事务用“开始事务”原语来开始的，用“中止”原语来结束的，再考虑到系统强迫的中止。

2. 持久性

在数据库中，要保证当完成了事务提交以后，系统必须使其操作的结果永远不会丢失，而不管这之后有无故障以及发生何种故障。这就要求系统必须把保留的事务结果存放在数据库中，所以提供事务持久性的活动称之为数据库的恢复。

3. 一致性

事务执行的结果必须是使数据库从一个一致性状态到另一个一致性状态。因此当数据库只包含成功事务提交的结果时，就说数据库处于一致性状态。如果数据库系统运行中发生故障，有些事务尚未完成就被迫中断，这些未完成事务对数据库所做的修改有一部分已写入物理数据库，这时数据库就处于一种不正确的状态，或者说不一致的状态。例如某公司在银行中有 A, B 两个账号，现在公司想从账号 A 中取出一万元，存入 B 账号。那么就可以定义一个事务，该事务包括两个操作，第一个操作是从账号 A 中减去一万元，第二个操作是向账号 B 中加入一万元。这两个操作要么全做，要么全不做。全做或者全不做，数据库都处于一致性状态。如果只做一个操作则用户逻辑上就会发生错误，少了一万元，这时数据库就处于不一致状态。可见一致性和原子性是密切相关的。

4. 隔离性

当一个事务在没有完成时，是不能在其托付之前把结果暴露给其他事务，这种信息的屏蔽作用就是隔离性。换句话说，如果一个事务（或者说是部分结果）能被观察到以后将被中止，则它必须被中止，如果这些事务中有几个已被托付，我们也将不得不取消它们，因为它们违反了事务持久性的性质。所以，为了有效地实现事务，事务隔离是极其重要的。

1.1.3 事务历史查询

在多数主动数据库系统中，规则可以查询事务执行的历史信息。可查询的信息包括两种：数据库的历史状态（历史数据）和历史事件。

1. 历史数据

主动数据库中存取（access）所发生事件影响数据的历史状态的一般方法是扩展规则定义语言的查询语法，在规则定义语言中使用特定的关键字（keyword）表示所查询的数据是历史数据。原则上从事务开始执行到事件发生以及所有这中间的数据库状态都可以成为查询的目标，但是在实际的系统中，只有某些特定的状态可以被查询：