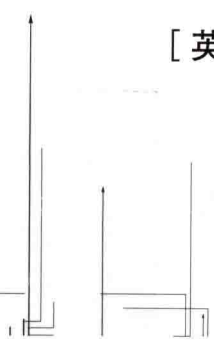
数据质量管理基础

Foundations of Data Quality Management

[英] 樊文飞　[英] 弗洛里斯·吉尔茨　著
刘瑞虹　贾西贝　译



国防工业出版社
National Defense Industry Press

装备科技译著出版基金

数据质量管理基础

Foundations of Data Quality Management

[英]樊文飞　[英]弗洛里斯·吉尔茨　著
刘瑞虹　贾西贝　译

国防工业出版社
·北京·

著作权合同登记　图字:军-2014-061号

图书在版编目(CIP)数据

数据质量管理基础/(英)樊文飞,(英)吉尔茨(Geerts,F.)著;刘瑞虹,贾西贝译. —北京:国防工业出版社,2016.1

书名原文:Foundations of Data Quality Management

ISBN 978-7-118-10138-6

Ⅰ.①数… Ⅱ.①樊… ②吉… ③刘… ④贾… Ⅲ.①数据管理-质量管理-研究 Ⅳ.①TP274

中国版本图书馆 CIP 数据核字(2015)第 298133 号

※

国防工业出版社出版发行

(北京市海淀区紫竹院南路 23 号　邮政编码 100048)

北京嘉恒彩色印刷责任有限公司印刷

新华书店经售

*

开本 880×1230　1/32　印张 7 ½　字数 214 千字

2016 年 1 月第 1 版第 1 次印刷　印数 1—2000 册　定价 69.00 元

(本书如有印装错误,我社负责调换)

国防书店:(010)88540777　　发行邮购:(010)88540776

发行传真:(010)88540755　　发行业务:(010)88540717

目　录

第1章　数据质量概述

传统数据库教科书上往往教人们如何设计数据库、如何构造查询语句以及如何提高查询速度，以至于人们相信只要查询方式正确，通过数据库管理系统（DBMS）就可以找到正确答案。然而，实际情况并非如此，在现实世界中，数据往往是“脏”的。如果基于数据库中的“脏”数据，不管是我们的查询语句如何完美，不管我们的 DBMS 如何高效，也不能保证得到精确、完整、实时甚至正确的查询结果，这些问题的存在凸显了提高数据质量的必要性。

本章首先描述与数据质量相关的重要核心问题，然后给出基于数据质量规则处理这些问题的统一逻辑框架。

1.1　数据质量管理

传统数据库系统通常只专注于数据的量，支持创建、维护和使用大量的数据。但如果数据库中被查询的数据是“脏”的，即数据不代表真实世界的实体所指，这样的数据库系统就无法找到正确的答案。

例如，下面是某公司数据库中的一个员工关系表：

employee(FN, LN, CC, AC, phn, street, city, zip, salary, status)，

这里每个元组描述了一个员工的姓名（名 FN、姓 LN）、办公电话（国家编码 CC、区域编码 AC、电话号码 phn）、办公地址（街道 street，城市 city，邮编 zip）和婚姻状态（status）。图 1.1 中给出员工关系表 employee 的一个实例 $\boldsymbol{D}_0$。

基于关系表实例 $\boldsymbol{D}_0$，考虑以下查询情况：

（1）查询 Q_1：查找在纽约办公室（NYC）工作的员工数量。通过计数求和元组 t_1、t_2、t_3，查询 Q_1 的答案是 3。但此答案可能不正确。首

	FN	LN	CC	AC	phn	street	city	zip	salary	status
t_1:	Mike	Clark	44	131	null	Mayfield	NYC	EH40 8LE	60k	single
t_2:	Rick	Stark	44	131	3456789	Crichton	NYC	EH40 8LE	96k	married
t_3:	Joe	Brady	01	908	7966899	Mtn Ave	NYC	NJ0 7974	90k	married
t_4:	Mary	Smith	01	908	7966899	Mtn Ave	MH	NJ 07974	50k	single
t_5:	Mary	Luth	01	908	7966899	Mtn Ave	MH	NJ 07974	50k	married
t_6:	Mary	Luth	44	131	3456789	Mayfield	EDI	EH4 8LE	80k	married

图 1.1　employee 实例

先，$\boldsymbol{D}_0$中的数据不一致，t_1、t_2、t_3 的属性 CC 和 AC 的取值与属性 city 的取值矛盾。当 CC = 44 且 AC = 131，city 应为英国的爱丁堡(EDI)，而非美国的纽约(NYC)。类似地，当 CC = 01 且 AC = 908 时，city 应为美国的默里山(MH)。因此，很可能 NYC 不是元组 t_1、t_2 和 t_3 中属性 city 的实值。其次，$\boldsymbol{D}_0$中的信息可能对于在纽约工作的员工来说不完备，也就是说，有些表示在纽约工作的员工的元组可能不在 $\boldsymbol{D}_0$中，因此，查询 Q_1的结果是 3 也是不可靠的。

(2) 查询 Q_2：查找 FN = Mary 的不同员工数量。通过枚举元组 t_4、t_5 和 t_6，可知查询 Q_2的答案是 3。但是元组 t_4、t_5 和 t_6 可能分别描述了 Mary 在不同时期的信息，因此实际上代表同一个人，分别描述了 Mavy 在不同时期的信息。这样，查询 Q_2的正确答案也许是 1 而非 3。

(3) 查询 Q_3：假定已知元组 t_4、t_5 和 t_6 代表同一个人，查找 Mary 目前的薪水和姓氏。简单评估就可以知道在 $\boldsymbol{D}_0$中查询 Q_3的 salary 取值为 50k 或 80k，而 LN 的取值为 Smith 或 Luth。但如果缺少可靠的时间戳，将无法确定目标的薪水和姓氏。

上述查询例子说明，当存在“脏”数据时，无论是所提供适应大量数据的能力还是如何高效处理的能力多么强大，都不能指望数据库管理系统对查询给出正确答案。然而，现实社会中的数据往往是“脏”的，包括不一致、重复、不精确、不完整和过时的现象。数据表明，企业通常会发现自己的数据错误率在 1% ~5% 之间，一些企业的数据错误率甚至达 30% 以上[Redman, 1998]。在绝大多数数据仓库项目中，数据清洗占据 30% ~80% 的开发时间和预算经费[Shilakes, Tylman,

1998]，用于提高数据质量而不是开发系统。对于信息的不完整性，据估计，当时用于医疗决策的信息的缺失率达到 13.6% ~81% [Miller Jr. 等, 2005]。对于数据的时效性，数据表明“一个月内 2% 的客户文件记录将过时”[Eckerson, 2002]。这意味着，一个拥有 500000 客户记录的数据库，每月有 10000 个记录失效，每年有 120000 记录失效，两年内有超过 50% 的记录将过时。

如此关注“脏”数据，是因为数据质量已经成为数据管理领域最严峻的挑战之一。据相关报道，“脏”数据每年使美国业务损失 6000 亿美元[Eckerson, 2002]，仅因为零售数据库中标错的价格数据，使美国消费者每年损失 25 亿美元[English, 2000]。虽然这些只是表明美国的“脏”数据带来很大的成本，但仍相信，其他依赖于信息技术国家，其产生“脏”数据的规模不会有什么不同。

这些凸显了数据质量管理的必要性，即通过提高数据库中数据质量，用一致、准确、全面、合时、唯一的数据代表现实世界的实体所指是非常必要的。

数据质量管理应至少与传统数据管理任务对数据量的考虑一样重要。各个行业对开发数据质量管理系统的需求不断增加，希望能高效探查和纠正数据中的错误，增加业务过程的精度和价值。事实上，数据质量工具的市场每年以 16% 速率增长，平均比其他 IT 领域高 7% [Gartner, 2011]。例如，在英国，电信每年数据质量工具带来“超过 6 亿英镑的整体业务价值”[Otto, Weber, 2009]。数据质量管理同样是大数据管理、主数据管理(MDM) [Loshin, 2009]、客户关系管理(CRM)、企业资源计划(ERP)和供应链管理(SCM)等系统的重要组成部分。

1.2 数据质量的核心问题

首先重点给出数据质量相关的 5 个核心问题，即数据一致性、数据去重、数据精确性、信息完整性和数据时效性。

1.2.1 数据一致性

数据一致性是指代表数据(语义)的正确性。其目的是检测数据

中的不一致或冲突。在关系型数据库中，不一致性可能存在于单个元组中，同一关系（表）的不同元组之间，或者不同关系（表）的元组之间。

例如，对于图 1.1 中给出的元组 t_1、t_2 和 t_3，元组内部存在矛盾和冲突，不同的元组之间也存在不一致性。

（1）在元组 t_1 中，知道国家是英国（CC = 44），如果区域码（AC）是 131，城市 city 应为爱丁堡（EDI），但实际上 t_1 中 CC = 44，AC = 131，而 city ≠ EDI。这说明，在元组 t_1 属性 CC、AC、city 取值之间存在不一致性。同样，元组 t_2 存在类似问题。这说明元组 t_1 和 t_2 是错误的。

（2）在元组 t_3 中，同样知道国家是美国（CC = 01），如果区域码（AC）为 908，则城市 city 应为默里山（MH），但实际上 t_3 中 CC = 01，AC = 908，而 city ≠ MH。这说明元组 t_3 同样存在错误。

（3）在英国，邮政编码（zip）对应唯一街名（street），即任何代表英国员工的元组，如果具有相同的邮政编码，那么这两个元组的 street 属性应具有相同的取值。而 $t_1[\mathrm{CC}] = t_2[\mathrm{CC}] = 44$，$t_1[\mathrm{zip}] = t_2[\mathrm{zip}]$，$t_1[\mathrm{street}] \neq t_2[\mathrm{street}]$，这说明元组 t_1 和 t_2 之间存在冲突。

是否违反数据依赖（也称完整性约束[Abiteboul 等，1995]）通常被认为是判断数据是否不一致的标准。在第 2 章条件依赖中将看到，单个关系中的错误可通过关系内约束（如扩展函数依赖）进行检测，多个关系之间的错误通过跨关系约束（如扩展包含依赖）进行识别。

1.2.2 数据（去重）

数据去重是指识别一个或多个关系中代表现实世界中相同实体的不同元组，也称实体统一、重复检测、记录匹配、记录链接、合并清除。处理复杂结构数据时，也称对象识别。

例如：对于图 1.1 中的三个元组 t_4、t_5 和 t_6，为了回答前面的查询 Q_2，首先希望了解这三个元组是否指向同一员工。如果还有一个关系表指明 Mary Smith 和 Mary Luth 具有相同的 E - mail，那么这两个人是同一个人。

研究数据去重的必要性显而易见：在数据清洗过程中需要去掉重复的记录；数据集成也需要整理和融合来自多个数据源相同实体的信息；在主数据管理中，数据去重还可以帮助人们识别输入元组和主数据

之间的关联。这种需求还体现在支付卡欺诈,2006 年全世界此类欺诈造成的损失达 48.4 亿美元[SAS, 2006]。在欺诈检测中,交叉检查一个信用卡使用者是否为合法持卡人是一种常规过程。另一个防欺诈的例子是通过比对有执照飞行员的相关记录和从美国社会安全局领取伤残福利的个人记录,惊奇地发现,有 40 名飞行员的记录同时出现在两个数据库中[Herzog 等,2009]。

数据去重不是一个简单的问题。在第 4 章将看到属于同一对象的元组在具有不同模式的不同数据源中有不同的表示,而且数据源可能存在错误。这使得通过简单检查来判断属性是否彼此相等的两个元组匹配是非常困难的,更糟糕的是从庞大数据源中比较和检查每一对元组的过程往往非常复杂。

1.2.3 数据精确性

数据精确性是指数据库中数据值与数据库中数据所表示的实体的真实值之间的接近程度。例如一个人的关系模型如下:

person(FN, LN, age, height, status)

元组描述了一个人的名字(FN,LN)、年龄(age)、身高(height)和婚姻状态(status)。图 1.2 给出 person 实例,其中 s_0 代表 Mike 的"真实"信息。可以得出结论:s_1[age, height]比 s_2[age, height]更精确。因为 s_1[age, height]更接近 Mike 的真实值。同样,s_2[FN, status]比 s_1[FN, status]更精确。

	FN	LN	age	height	status
s_0:	Mike	Clark	14	1.70	single
s_1:	M	Clark	14	1.69	married
s_2:	Mike	Clark	45	1.60	single

图 1.2 person 实例

在实际应用中,通常不存在可以参照的真实值。如果元组 s_0 未知,确定 s_1 和 s_2 的相对精确性更加困难。然而对于某些属性,仍有可能通过分析数据的语义,判断其在一个元组的值是否比其他元组的值更精确。

(1) 如果已知 Mike 目前还上中学,那么可以得出 s_1[age]比 s_2[age]更精确,即 s_1[age]比 s_2[age]更接近 Mike 的真实年龄。虽然不知道 Mike 的真实年龄,但是一个中学生不可能是 45 岁。另外,从 s_1[age]年龄值可以推论出 s_2[status]比 s_1[status]更精确。

(2) 如果 s_1[height]和 s_2[height]曾经是正确的,那么 s_1[height]比 s_2[height]更精确。因为典型的一个人身高年轻时只能单向增长。

1.2.4 信息完整性

信息完整性关注数据库是否具有完整的信息来进行查询。给定数据库 D 和查询 Q,我们希望知道 Q 是否可以仅使用数据库 D 中的数据给予完整回答。如果数据库 D 中的信息不完整,很难指望 Q 的查询结果是精确和正确的。

在实际工作中,针对人们用来进行查询的数据库往往不具有足够的信息,对于数据库中数据所表示的实体来说,不管是属性值还是元组可能都有缺失。如图 1.1 所示,关系 D_0 的 t_1[phn]值缺失显示为 null,更糟糕的是有些员工元组可能在关系 D_0 缺失。从前面可以看到,对于查询 Q_1,如果在纽约办公室工作的员工元组在关系 D_0 缺失,则其查询结果不正确。不完整的信息将给企业带来严重问题:通常导致企业误导性的分析结果、具有偏差的决策,造成企业收入、信誉和客户的损失。

那么,如何应对信息不完整问题呢? 信息完整性的传统处理借助两种假定[Abiteboul 等, 1995],即封闭世界假定(the Closed World Assumption,CWA)和开放世界假定(the Open World Assumption,OWA)。

(1) CWA 假定一个数据库中收集了代表现实世界实体的全部元组,但是这些元组的某些属性值可能缺失。

(2) OWA 除假定数据值缺失外,还假定代表现实世界实体的某些元组缺失,即数据库只代表现实世界实体全部元组集的真子集。

目前的数据库理论通常是在 CWA 下发展起来的,然而现实中数据库不仅属性值缺失,元组也缺失。也就是说,CWA 假定不成立。另外,在 OWA 下,期待可以找到完整答案的合理查询。

在第 5 章将看到,CWA 和 OWA 对于一些新出现的应用不合适,如主数据管理。换句话说,现实世界的数据库既不是完全封闭的世界也

不是完全开放的世界，这些数据库实际上属于“半封闭”。已经发现，半封闭数据库能给出回答手头查询所需的完整信息。

1.2.5 数据时效性

数据时效性也称数据合时性，其目标是识别数据库中元组表示实体的当前值，以便用当前值回答有关查询。

如果数据值具有有效的时间戳，保证数据时效性就不是大问题，但现实中经常发现时间戳不可用或不准确[Zhang 等，2010]，以及数据值的复制和从其他数据库导入[Berti - Equille 等，2009；Dong 等，2010，2009a，b]，难以形成统一的时间戳模式，造成从数据库数据中识别出实体的“最新”值非常困难和具有挑战性。

回忆图 1.1 中 employee 关系 D_0 的查询 Q_3，假定通过数据去重技术[Elmagarmid 等，2007]，发现元组 t_4、t_5、t_6 属于同一员工 Mary。如前所述，由于缺乏可靠的时间戳，对于关系 D_0 的查询 Q_3 所给出的答案并不能告诉 Mary 目前的薪水是 50k 还是 80k，也不能告诉她目前的姓氏是 Smith 还是 Luth。

这不意味着全部信息的丢失，第 6 章将看到，通过数据语义可以推断出数据的时效性：

（1）当确实没有可靠的时间戳关联 Mary 的薪水，我们则可通过常识判断：员工的薪水往往是递增的，这样就可以知道 t_6[salary]要比 t_4[salary]和 t_5[salary]更具有时效性，因此 Mary 的薪水为 80k。

（2）婚姻状态只能从单身到结婚，从结婚到离婚，而不能从结婚到单身。另外，employee 具有最新婚姻状态的元组中应包含最新的姓氏数据，因此，t_6[LN] = t_5[LN]比 t_4[LN]更具有时效性，即 Mary 目前的姓氏是 Luth。

1.2.6 数据质量问题之间的相互作用

为了提升数据质量，往往需要处理上述 5 个核心问题。这些数据质量问题相互作用，如下所示。

由前面已经看到，图 1.1 关系 D_0 中元组 t_1、t_2、t_3 是不一致的，下面介绍如何利用数据去重技术帮助解决不一致问题。假定公司为办公室

维护了一个主数据关系，包括每个办公室的地址和电话号码的一致、完整、最新时效信息。主数据关系表示为 D_m（图 1.3），其模式描述为

office(CC, AC, phn, street, city, zip)

	CC	AC	phn	street	city	zip
t_{m1}:	44	131	9083456789	Mayfield	EDI	EH4 8LE
t_{m2}:	01		7966899	Mtn Ave	MHNJ	07974

图 1.3　office 主数据关系举例

在第 7 章将可以看到，对元组 t_1、t_2、t_3 进行如下“修复”：

（1）如果元组属性 CC、AC 值是精确的，基于上述理由可以更改其 city 属性，$t_1[\text{city}] = t_2[\text{city}] := \text{EDI}$，$t_3[\text{city}] := \text{MH}$，从而产生不同于 t_1、t_2、t_3 的 t_1'、t_2'、t_3'。其唯一差别是 city 的属性值。

（2）如果存在 employee 元组 $t \in D_0$ 和 office 元组 $t_m \in D_m$，它们的地址(street, city, zip)保持一致，那么两个元组是匹配的，即具有相同的地址和电话。因此，可以根据 t_m 中所对应主数据值来修改 t[CC, AC, phn]。这样也允许将 t_2'[street]修改为 t_{m1}[street]。就是说，通过匹配 t_2' 和 t_{m1} 修复 t_2'[street]，将导致元组 t_2'' 的产生，其与 t_2' 的不同点仅在于 street 属性。

（3）对于 employee 元组 t_1、t_2，如果具有相同的地址，就应有相同的 phn 值。基于此，可以通过 $t_1'[\text{phn}] := t_2''[\text{phn}]$ 强化 t_1'[phn]，从而获得新的元组 t_1''。

很容易验证 t_1''、t_2''、t_3'' 是一致的，在上述处理过程中交织了解决冲突的操作(步骤(1)和步骤(3))和探查重复的操作(步骤(2))。一方面解决冲突可以帮助去重，步骤(2)只能在 t_2[city]被纠正后才能执行；另一方面去重可以帮助人解决冲突，t_1'[phn]只有在 t_2'[street]通过匹配和修复后，才能被充实。

数据质量不同问题之间存在相互作用，包括但不限于以下几个方面：

（1）在提高信息完整性的处理过程中，如果能得到更多的时间信息，数据的时效性将会得到明显提升。

（2）为了确定一个实体的当前值，需要通过数据去重操作来识别

相同实体所属元组。例如，为找到图 1.1 关系 D_0 中员工 Mary 的 LN 当前值，需要查询三个元组 t_4、t_5、t_6 是否指同一人。

(3) 为了解决代表同一实体不同元组之间的冲突问题，需要确定该实体有关信息是否完整，只有这样才能从已有数据库可用数据中发现实体的真正数值。

由此表明，一个实用的数据质量管理系统：首先提供针对以上每一个核心问题的处理功能；其次利用这些问题之间的相互作用改进数据质量。

1.3 基于规则的数据质量改进

现实生活中的数据经常是“脏”的，而处理“脏”数据代价高昂。基于此，必须形成有效的技术来改进数据质量。

1. 现实生活数据错误种类

现实数据库中常见错误可分为如下两类：

(1) 语法错误：数据库中的值违背域约束条件。例如，如属性 name 的类型是 string，那么 name = 1.23 是语法错误；又如，属性 age 的取值范围是[0, 120]，而 age = 250。

(2) 语义错误：数据库中数值与数据所代表实体真正数值之间的不一致。语义错误例子涉及数据的一致性、重复性、精确性、信息完整性和时效性。

由于语法错误相对来说比较容易处理，而语义错误的检测和纠正将更具挑战性，因此本书中专注于语义错误。

2. 使用依赖作为数据质量规则

一个核心问题是如何知道数据是否有语义错误，即数据是“脏”的还是“干净”的。因此，需要数据质量规则检测数据的语义错误。更好的是，通过使用规则修正这些错误。

一个很自然的想法是采用数据依赖(完整性约束)。依赖理论几乎与关系数据库理论本身一样古老。自 1972 年 Codd 给出函数依赖后，提出和研究了不同类别一阶逻辑的各种依赖语言，因此有依赖在数据质量管理系统中扮演着重要角色。从本质上说，依赖是以一种声明

方式描述数据语义上的构成，即错误是对依赖的违反。此外，基于依赖的推理系统，因素分析和剖析方法表明，其可以作为对数据语义进行推理的系统方法，这些系统方法帮助人们推断与发现提升数据质量和其他方面的规则。另外，在后面将介绍数据质量的5个核心方面，即数据一致性、数据去重、数据精确性、数据时效性和信息完整性可以通过数据依赖方式描述。这样允许将数据质量问题归结成统一的逻辑框架，在统一的逻辑框架下研究数据质量问题的相互作用和影响。

将依赖应用于数据质量管理过程中，需要对经典的依赖理论进行扩展。传统的依赖理论提出达到目的：①通过标准化提升模式的质量；②优化查询和防止无效修改（[Abiteboul 等，1995]）。为了提高数据质量，需要一种新形式的依赖，这种新形式的依赖既可以描述捕获数据不一致语义相关数据值的模式，也可以支持数据去重过程中相似谓词容错，同时强化主数据中核心业务实体有关信息的包含性，以便对信息完整性进行推理，以及通过时间序列来确定数据的时效性。

在开发提升数据质量的依赖中，首先需要在表达能力和表达复杂性两者之间掌握平衡，并重温依赖的经典问题，如可满足性、蕴含性和有限可公理化性分析。

3. 利用规则提升数据质量

在提出“正确”用于描述数据质量规则的依赖语言以后，接下来的问题是如何有效利用这些规则提升数据质量。简而言之，一个基于规则的数据质量管理系统应提供以下功能：

（1）发现数据质量规则。为了使用依赖作为数据质量规则，需要高效的技术自动从数据中发现依赖关系。事实上，单纯依靠个人专家通过昂贵冗长的手工处理来设计数据质量规则往往是不切实际的，依赖积累的业务规则通常也不够。这表明，需要从数据（可能是“脏”的）中学习有价值的和感兴趣的数据质量规则，并基于用户设置的阈值删除琐碎、微不足道的规则。

（2）验证数据质量规则。给定一个依赖集合 Σ，其或自动发现或由领域专家手工设计，本身可能是“脏”的，因此必须从集合 Σ 中识别出“一致”的依赖，即可以用于数据质量规则的规则。另外，通过对集