第3章

数据质量与数据管理

本章学习目标

- 了解数据质量的概念
- 了解数据质量管理的概念
- 了解数据标准的概念
- 掌握主数据的定义
- 掌握元数据的定义与特征
- 了解电子文件元数据
- 了解元数据管理的定义与实施

本章先向读者介绍数据质量与数据质量管理，再介绍数据标准，接着介绍主数据与元数据，最后介绍元数据管理的定义与实施。

3.1 数据质量与数据质量管理概述

3.1.1 数据质量

1. 数据质量介绍

数据无处不在，它贯穿整个数据生命周期，为企业决策提供了可靠的基础支撑，是企业成功的关键。在大数据时代，随着企业数据规模的不断扩大、数据数量的不断增加及数据来源的复杂性不断变化，为了能够充分地利用数据价值，企业需要对数据进行管理。

然而，大数据应用必须建立在质量可靠的数据之上才有意义，建立在低质量甚至错误数据之上的应用有可能与其初心背道而驰。数据质量就是确保组织拥有的数据完整且准确，只有完整、准确的数据才可以供企业分析、共享使用。因此，组织只有拥有强大的数据质量流程才可以确保数据的干净和清洁。

2. 数据质量术语

（1）质量。一组固有特性满足要求的程度。

（2）准确度。在一定观测条件下，观测值及其函数的估值与其真值的偏离程度。

（3）一致性。满足规定的要求。

（4）一致性质量级别。数据质量结果的一个或一组阈值，用于确定数据集符合产品规范规定或用户要求的程度。

（5）数据质量结果。数据质量测量得到的一个值或一组值，或者将获取的一个值或一组值与规定的一致性质量级别相比较得到的评价结果。

（6）数据质量范围。记录其质量信息的数据的覆盖范围或特征。

（7）数据质量值类型。记录数据质量结果的值的类型。

（8）数据质量值单位。记录数据质量结果的值的单位。

（9）完全检查。检查质量范围内的所有个体。

（10）检验单元。可被单独描述或考察的事物。

（11）要素。现实世界现象的抽象。

3. 造成数据质量的常见问题

造成数据质量的常见问题大致可以分为3种，即技术原因、业务原因和管理原因。

1）技术原因

（1）数据模型设计的质量问题。例如，数据库表结构、数据库约束条件、数据校验规则的设计开发不合理，造成数据录入无法校验或校验不当，引起数据重复、不完整、不准确。

（2）数据源存在数据质量问题。例如，有些数据是从生产系统采集过来的，在生产系统中这些数据就存在重复、不完整、不准确等问题，而采集过程中没有对这些问题做清洗处理，这种情况也比较常见。

（3）数据采集过程的质量问题。例如，采集点、采集频率、采集内容、映射关系等采集参数和流程设置不正确，数据采集接口效率低，导致数据采集失败、数据丢失、数据映射和转换失败。

（4）数据传输过程的问题。例如，数据接口本身存在问题、数据接口参数配置错误、网络不可靠等都会造成数据传输过程中发生数据质量问题。

（5）数据装载过程的问题。例如，数据清洗规则、数据转换规则、数据装载规则配置有问题。

（6）数据存储的质量问题。例如，数据存储设计不合理、数据的存储能力有限、人为后台调整数据，引起数据丢失、数据无效、数据失真、记录重复。

（7）系统原因。业务系统各自为政，烟囱式建设，系统之间的数据不一致问题严重。

2）业务原因

（1）业务需求不清晰。例如，数据的业务描述、业务规则不清晰，导致技术无法构建出合理、正确的数据模型。

（2）业务需求的变更。这个问题其实对数据质量的影响非常大，需求一变，数据模型设计、数据录入、数据采集、数据传输、数据装载、数据存储等环节都会受到影响，稍有不慎就会导致数据质量问题的发生。

（3）业务端数据输入不规范。常见的数据录入问题有大小写、全半角、特殊字符等一不小心录错。人工录入的数据质量与录入数据的人员密切相关，录入数据的人员工作严谨、认真，数据质量就相对较好，反之就较差。

（4）数据造假。某些操作人员为了提高或降低考核指标，对一些数据进行处理，使得数据的真实性无法保证。

3）管理原因

（1）认知问题。企业管理缺乏数据思维，没有认识到数据质量的重要性，重系统而轻数据，认为系统是万能的，数据质量差一些也没关系。

（2）没有明确的数据归口管理部门或岗位。企业缺乏数据认责机制，出现数据质量问题找不到负责人。

（3）缺乏数据规划。企业没有明确的数据质量目标，没有制定与数据质量相关的政策和制度。

（4）数据输入规范不统一。不同的业务部门、不同的时间甚至在处理相同业务时，由于数据输入规范不同，造成数据冲突或矛盾。

（5）缺乏有效的数据质量问题处理机制。数据质量问题从发现、指派、处理到优化没有一个统一的流程和制度支撑，数据质量问题无法闭环。

（6）缺乏有效的数据管控机制。对历史数据的质量检查、新增数据的质量校验没有明确和有效的控制措施，出现数据质量问题无法考核。

值得注意的是，数据量定义了分析所需的数据量。在数据质量计划开始时估计和评估数据量对于程序的成功是至关重要的。例如，需要的数据是太少还是太多？观察的次数是多少？没有太多数据的缺点是什么？这些问题可以帮助人们决定驱动数据质量计划所需的工具和技术。

4. 数据质量评估

数据质量一般指数据能够真实、完整地反映经营管理实际情况的程度，通常可在以下几个方面衡量和评价：

（1）准确性。准确性是指数据在系统中的值与真实值相比的符合情况，一般而言，数据应符合业务规则和统计口径。常见的数据准确性问题如下：

① 与实际情况不符。数据来源存在错误，难以通过规范进行判断与约束。

② 与业务规范不符。在数据的采集、使用、管理、维护过程中，业务规范缺乏或执行不力，导致数据缺乏准确性。

（2）完整性。完整性是指数据的完备程度。常见的数据完整性问题如下：

① 系统已设定字段，但在实际业务操作中并未完整采集该字段数据，导致数据缺失或不完整。

② 系统未设定字段；或存在数据需求，但未在系统中设定对应的取数字段。

（3）一致性。一致性是指系统内外部数据源之间的数据一致程度，数据是否遵循了统一的规范，数据集合是否保持了统一的格式。常见的一致性问题如下：

① 缺乏系统联动。系统间应该相同的数据却不一致。

② 联动出错。在系统中缺乏必要的联动和核对。

（4）可用性。可用性一般用来衡量数据项整合和应用的可用程度。常见的可用性问题如下：

① 缺乏应用功能，没有相关的数据处理、加工规则或数据模型的应用功能。

② 缺乏整合共享，数据分散，不易有效整合和共享。

另外，还有其他衡量标准。如有效性可考虑对数据格式、类型、标准的遵从程度，合理性可考虑数据符合逻辑约束的程度。

例如对国内某企业数据质量问题进行调研显示如下：常见数据质量问题中准确性问题占33%、完整性问题占28%、可用性问题占24%、一致性问题占8%，这在一定程度上代表了国

内企业面临的数据问题。表 3-1 显示了数据质量评估的常见等级(注：数据质量问题频率＝数据质量问题发生次数/存储的总数据量,指标单位为次/吉字节)。表 3-2 显示了数据质量评估的参考维度。

表 3-1　数据质量评估的常见等级

数据质量等级	描　　述	统 计 口 径
一级	数据质量差,需要重点监控	数据质量问题频率大于或等于 1 次/吉字节
二级	数据质量一般	数据质量问题频率大于或等于 0.5 次/吉字节,小于 1 次/吉字节
三级	数据质量好	数据质量问题频率小于 0.5 次/吉字节

表 3-2　数据质量评估的参考维度

维　　度	描　　述	标　　准
准确性	数据准确体现了真实情况	数据内容和定义是否一致
精确性	数据精度满足业务要求的程度	数据精度是否达到业务规则要求的位数
完整性	必需的数据项已经被记录	业务指定必需的数据是否缺失,不允许为空字符或者空值等
时效性	数据被及时更新以体现当前事实	当需要使用时,数据能否反映当前事实,即数据必须及时,能够满足系统对数据时效的要求
唯一性	数据在特定数据集中不存在重复值	每条数据是否唯一
依赖一致性	数据项的取值满足与其他数据项之间的依赖关系	数据是否有相同的依赖
可访问性	数据易于访问	数据是否便于自动化读取
业务有效性	数据符合已定义的业务规则	数据项是否按已定义的格式标准组织
技术有效性	数据符合已定义的格式规范	数据是否符合规范
可用性	数据在需要时是可用的	数据可用时间和数据需要被访问时间的比例
参照完整性	数据项在被引用的父表中有定义	数据项是否在父表中有定义

5. ISO 8000 数据质量标准

ISO 8000 数据质量标准是国际标准化组织针对数据质量制定的标准,该标准致力于管理数据质量,具体来说,包括规范和管理数据质量活动、数据质量原则、数据质量术语、数据质量特征(标准)和数据质量测试。根据 ISO 8000 数据质量标准的要求,数据质量的高低程度由系统数据与明确定义的数据要求进行对比得到。通过 ISO 8000 标准的规范,可以保证用户在满足决策需求和数据质量的基础上,在整个产品或服务的周期内高质量地交换、分享和存储数据,从而保证用户可以依托获取的数据高效地做出最优化的安全决策。

通过将 ISO 8000 标准应用于组织内部,可以对组织内的数据进行规范化整合和管理,对各个部门的数据进行统一识别和管理,从组织的整体层面进行资源与信息的协调管理,从而减少因为信息沟通不畅带来的运营成本。此外,如果在合作公司之间或整个行业采用 ISO 8000 标准,数据或信息将会更有可用性。例如,在医疗卫生领域,各个医疗机构的信息系统不能很好地兼容,导致同一病人在不同医院的信息无法快速共享和传递。通过在全国范围内应用 ISO 8000 数据质量标准,可以将病历信息与特定信息系统分离,使病历的所有信息独立于医疗信息系统存在,并可被任意一个应用 ISO 8000 数据质量标准的信息系统读取,患者可以更加自主地选择就医医院,而不用担心由于对自身的健康信息缺失导致医疗误判。

3.1.2 数据质量管理

1. 数据质量管理介绍

数据价值的成功发掘必须依托于高质量的数据，只有准确、完整、一致的数据才有使用价值。因此需要从多维度来分析数据的质量，例如偏移量、非空检查、值域检查、规范性检查、重复性检查、关联关系检查、离群值检查、波动检查等。需要注意的是，优秀的数据质量模型的设计必须依赖于对业务的深刻理解，在技术上也推荐使用大数据相关技术来保障检测性能和降低对业务系统性能的影响，例如 Hadoop、MapReduce、HBase 等。

数据质量管理是指对数据从计划、获取、存储、共享、维护、应用到消亡整个生命周期的每个阶段里可能引发的各类数据质量问题进行识别、度量、监控、预警等一系列管理活动，并通过改善和提高组织的管理水平使数据质量获得进一步提高。数据质量管理是企业数据治理的一个重要的组成部分，企业数据治理的所有工作都是围绕提升数据质量目标而开展的。

值得注意的是，在数据治理方面，不论是国际的还是国内的，人们能找到很多数据治理成熟度评估模型这样的理论框架作为企业实施的指引。说到数据质量管理的方法论，其实在业内还没有一套科学、完整的数据质量管理体系。因为数据质量管理不单纯是一个概念、一项技术、一个系统，更不单纯是一套管理流程，数据质量管理是一个集方法论、技术、业务和管理于一体的解决方案。通过有效的数据质量控制手段进行数据的管理和控制，从而消除数据质量问题，进而提升企业数据变现的能力。

2. 数据质量管理的价值

数据质量管理的目标是解决企业内部数据在使用过程中遇到的数据质量问题，提升数据的完整性、准确性和真实性，为企业的日常经营、精准营销、管理决策、风险管控等提供坚实、可靠的数据基础。

因此，数据质量管理的价值就是通过建设一个完整的数据质量管理平台对数据进行检核与统计，从制度、标准、监控、流程几个方面提升对数据信息的管理能力，解决项目面临的数据标准问题、数据质量问题，为数据治理提供准确的数据信息。通过数据质量管理能够完成从发现数据问题到最后解决数据问题的过程，从而为企业不断提高数据质量，完成从数据产生、数据交换到数据应用中数据质量的统一管理与控制。

3. 数据质量管理的主要工作

数据质量管理主要有以下几个工作：

1）组织环境

建设强有力的数据管理组织是数据治理项目成功最根本的保证。其涉及两个层面：一是在制度层面，制定企业数据治理的相关制度和流程，并在企业内推广，融入企业文化；二是在执行层面，为各项业务应用提供高可靠的数据。

2）数据质量管理的方针

为了改进和提高数据质量，必须从产生数据的源头开始抓起，从管理入手，对数据运行的全过程进行监控，强化全面数据质量管理的思想观念，把这一观念渗透到数据生命周期的全过程。数据质量问题是影响系统运行、业务效率、决策能力的重要因素，在数字化时代，数据质量问题更是影响企业降本增效、业务创新的核心要素。对于数据质量问题的管理，采用事前预防控制、事中过程控制、事后监督控制的方式进行控制，持续提升企业数据的质量水平。

3）数据质量问题的分析

对于质量问题的分析，企业可以使用经典的六西格玛(6 Sigma)。六西格玛是一种改善企业质量流程管理的技术，以“零缺陷”的完美商业追求带动质量成本的大幅度降低，它以客户为导向，以业界最佳为目标，以数据为基础，以事实为依据，以流程绩效和财务评价为结果，持续改进企业经营管理的思想方法、实践活动和文化理念。六西格玛重点强调质量的持续改进，对于数据质量问题的分析和管理，该方法依然适用。

4）数据质量监控

数据质量监控可以分为数据质量的事前预防控制、事中过程控制和事后监督控制。

(1) 事前预防控制。建立数据标准化模型，对每个数据元素的业务描述、数据结构、业务规则、质量规则、管理规则、采集规则进行清晰的定义，数据质量的校验规则、采集规则本身也是一种数据，在元数据中定义。如果没有元数据来描述这些数据，使用者无法准确地获取所需信息。正是通过元数据，使得数据可以被理解、使用，从而产生价值。构建数据分类和编码体系，形成企业数据资源目录，能够让用户轻松地查找和定位到相关的数据(关于元数据的有关内容，请参考本章的 3.3 节)。

(2) 事中过程控制。事中过程控制，即在维护和使用数据的过程中监控和处理数据质量。通过建立数据质量的流程化控制体系，可以对数据的新建、变更、采集、加工、装载、应用等环节进行流程化控制。

(3) 事后监督控制。定期开展数据质量的检查和清洗工作，并作为企业数据质量治理的常态工作来抓。监督控制工作主要包括设置数据质量规则、设置数据检查任务、出具数据质量问题报告、制定和实施数据质量改进方案、进行评估与考核等。

5）数据周期管理

数据生命周期从数据规划开始，中间是一个包括设计、创建、处理、部署、应用、监控、存档、销毁这几个阶段并不断循环的过程。企业的数据质量管理应贯穿数据生命周期的全过程，覆盖数据标准的规划设计、数据的建模、数据质量的监控、数据问题诊断、数据清洗、优化完善等各方面。

这里以典型的设备资产为例，如图 3-1 所示。其全生命周期一般包括 6 个环节，即设计、采购、安装、运行、维护和报废。从设备设计、采购开始，直至设备安装、运行、维护、报废进行全生命周期管理；将基建期图纸、采购、资料信息带到设备台账中，实现对设计数据、采购数据、施工数据、安装数据、调试数据等后期移交和设备系统生产运维所需要的完整数据的平滑过渡，实现基建、生产一体化，提升企业资产利用率，增强企业投资回报率。同时结合成本管理、财务管理，既实现对资产过程的管控，又实现对资产价值的管理。

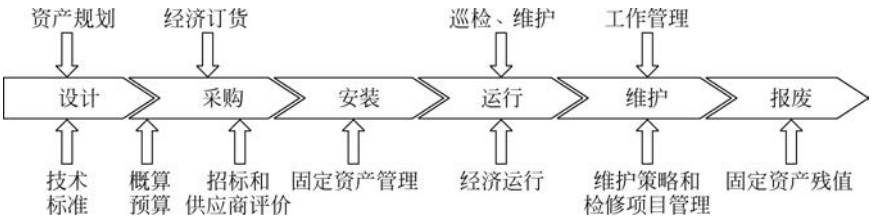


图 3-1　设备资产的全生命周期

在数据全生命周期管理中最重要的是数据规划、数据设计、数据创建和数据使用。

(1) 数据规划。从企业战略的角度不断完善企业数据模型的规划，把数据质量管理融入企业战略中，建立数据治理体系，并融入企业文化中。

（2）数据设计。推动数据标准化制定和贯彻执行，根据数据标准化要求统一建模管理，统一数据分类、数据编码、数据存储结构，为数据的集成、交换、共享、应用奠定基础。

（3）数据创建。利用数据模型保证数据结构完整、一致，执行数据标准，规范数据维护过程，加入数据质量检查，从源头系统保证数据的正确性、完整性、唯一性。

（4）数据使用。利用元数据监控数据使用；利用数据标准保证数据正确；利用数据质量检查加工正确。元数据提供各系统统一的数据模型进行使用，监控数据的来源去向，提供全息的数据地图支持；企业从技术、管理、业务三方面进行规范，严格执行数据标准，保证数据输入端的正确性；数据质量提供了事前预防、事中预警、事后补救三方面的措施，形成完整的数据治理体系。

要做好数据质量的管理，应抓住影响数据质量的关键因素，设置质量管理点或质量控制点，从数据的源头抓起，从根本上解决数据质量问题。在企业的数据治理中，进行数据质量管理必须识别相应产品规范或用户需求中的质量信息，在元数据、质量评价报告中形成正确的质量描述，并且这些规范上的质量结果均为“合格”。

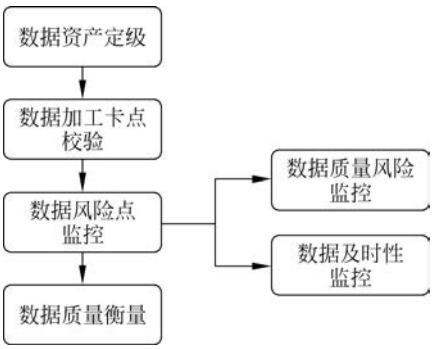


图 3-2　阿里云数据质量管理流程

4. 数据质量管理的实施

数据质量管理的方法较多，不同的企业有不同的实施方式。在这里以阿里云为例介绍数据质量管理的实施。阿里云通过划分数据资产等级和分析元数据的应用链路对不同资产等级的数据采取相应的质量管理方式，其数据质量管理流程如图 3-2 所示。

对数据质量管理流程中各环节的说明如下：

1）数据资产定级

数据是数字经济的核心，对于企业而言，数据更是企业重要的资产。数据资产是指个人或企业的照片、文档、图纸、视频、数字版权等以文件为载体的数据，相对于实物资产，它以数据形式存在。但是，并非企业拥有的所有数据都能被称为数据资产。企业的数据治理指的是企业对所拥有的数据资产的治理，只有关乎重大商业利益的数据资产才是数据治理的对象。重要的数据资产可以为企业带来显著的商业利润，因此这些数据资产也是企业资产的重要组成部分。

通常可以根据数据质量不满足完整性、准确性、一致性、及时性时对业务的影响程度来划分数据的资产等级，可以划分为以下 5 个性质的等级：

（1）毁灭性质。数据一旦出错，将会引起重大资产损失、面临重大收益损失等。标记为 A1。

（2）全局性质。数据直接或间接地用于企业级业务、效果评估和重要决策等。标记为 A2。

（3）局部性质。数据直接或间接地用于某些业务线的运营、报告等，如果出现问题，会给业务线造成一定的影响或造成工作效率降低。标记为 A3。

（4）一般性质。数据主要用于日常数据分析，出现问题带来的影响极小。标记为 A4。

（5）未知性质。无法明确数据的应用场景。标记为 Ax。

这些等级按重要性依次降低，即重要程度为 A1＞A2＞A3＞A4＞Ax。如果一份数据出现在多个应用场景汇总中，则根据其最重要程度进行标记。因此，企业需要通过对关键系统关键数据资源的梳理，形成企业数据资产目录，并通过对数据资产的盘点，不断推进企业数据整合共享及相关标准化工作。

值得注意的是，数据治理和数据资产管理是一个渐进的过程，不是所有数据都可以变成数据资产，只有数据在经过治理的二次加工达到了资产的利用要求并能够产生自身价值之后才能变成数据资产。因此，数据资产的管理过程同样不能脱离数据治理，数据治理是数据变成资产的条件，也是数据资产管理的必备功能和过程。

2）数据加工卡点校验

卡点校验在各个加工环节上根据不同资产等级对数据采取不同的质量管理方式，主要分为在线系统卡点校验和离线系统卡点校验。在线系统卡点校验要随时关注发布平台的变更和数据库的变更，而离线系统卡点校验需要关注代码的提交质量、任务发布时的线上检测及任务变更时的更新。

3）数据风险点监控

数据风险点监控分为在线数据风险点监控和离线数据风险点监控。在线业务系统的数据生成过程必须确保数据质量，根据业务规则对数据进行监控。例如对数据库表的记录进行规则校验，制定监控规则。在业务系统中，当每个业务过程进行数据入库时，对数据进行校验。在常见的交易系统中，订单拍下时间、订单完结时间、订单支付金额、订单状态流转都可以配置监控校验规则。

离线数据风险点监控则需要在离线系统加工时精准地把控数据准确性。离线数据风险点监控以数据集（可识别的数据集，数据集在物理上可以是更大的数据集的较小部分。从理论上讲，数据集可以小到更大数据集内的单个要素或要素属性。一张硬拷贝地图或图表均可以被认为是一个数据集）为监控对象，当离线数据发生变化时，会对数据进行校验，并阻塞生产链路，以避免问题数据污染扩散。系统还提供了对历史校验结果的管理，方便数据质量的分析和定级。此外，在确保数据准确性的前提下，系统还需要让数据能够及时提供服务，否则数据的价值将大幅度降低。

4）数据质量衡量

数据质量衡量是指针对每个数据质量事件，必须分析原因和处理过程，制定后续同类事件预防方案，可以将严重的数据质量事件升级为故障，并对故障进行定义、等级划分、处理和总结。

常见的数据质量处理过程如图 3-3 所示。

5. 数据质量管理的应用

数据质量管理的应用较多，下面以高校质量管理为例来讲述。

高校的各类业务较多，应用系统繁杂，在系统建设过程中往往会忽视数据质量的重要性，没有采取足够的措施，导致随着系统和数据的逐步深入应用，数据质量问题一点一点暴露出来，比如数据的有效性、准确性、一致性等。最坏的结果就是用户感觉系统和数据是不可信的，最终放弃了使用系统，这样也就失去了建设系统的意义。因此，在高校中数据质量是一个非常复杂的系统性问题，解决数据质量问题应该从数据质量管理制度、应用系统建设、数据质量监控三方面开展，并且三方面要有机结合，形成联动，单靠某一方面的努力是不够的。图 3-4 显示了高校数据质量监控平台。

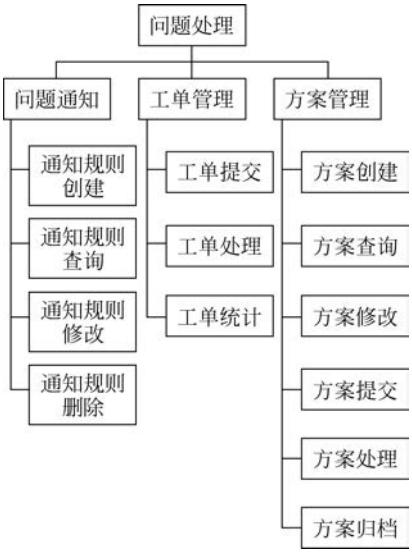


图 3-3　数据质量处理过程

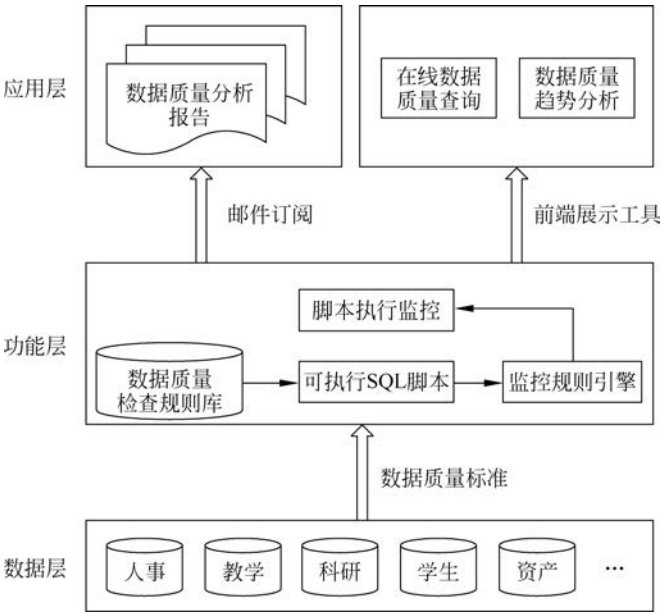


图 3-4 高校数据质量监控平台

从图 3-4 可以看出，数据质量监控平台主要包括三部分，即数据层、功能层和应用层。

数据层定义了数据质量监控的对象，主要是各核心业务系统的数据，例如人事系统、教学系统、科研系统、学生系统等。

功能层是数据质量监控平台的核心部分，包括数据质量检查规则的定义、数据质量检查规则脚本、数据质量检查规则执行引擎、数据质量检查规则执行情况的监控等。

在应用层中，数据质量检查结果可以通过两种方式访问：一种是通过邮件订阅方式将数据质量检查结果发给相关人员；另一种是利用前端展示工具（例如 MicroStrategy、Cognos、Tableau 等）开发数据质量在线分析报表、仪表盘、分析报告等。前端展示报表不仅能够查看汇总数据，而且能够通过钻取功能查看明细数据，以便业务人员能够准确定位到业务系统的错误数据。

在该平台中，数据质量检查规则库是监控平台的核心，用来存放用户根据数据质量标准定义的数据质量检查规则脚本，供监控规则引擎读取并执行，同时将检查产生的结果存放到监控结果表中。

3.2 数据标准

3.2.1 数据标准介绍

1. 认识数据标准

标准是指为了在一定的范围内获得最佳秩序，经协商一致制定并由公认机构批准，共同使用的和重复使用的一种规范性文件。数据标准是指对数据的表达、格式及定义的一致约定，包括数据业务属性、技术属性和管理属性的统一定义。其中，业务属性包括中文名称、业务定义、业务规则等，技术属性包括数据类型、数据格式等，管理属性包括数据定义者、数据管理者等。因此，对于数据标准的定义通俗地讲就是给数据一个统一的定义，让各系统的使用人员对同一指标的理解是一样的。

数据标准对于企业来说是非常重要的。因为大数据时代数据应用分析项目特别多，如果数据本身存在非常严重的问题，例如数据统计口径不统一、数据质量参差不齐、数据标准不统一等，往往会影响到项目的正常交付，甚至会影响到后续数据应用和战略决策。在整个项目实施过程中，应用系统之间需要上传下达、信息共享、集成整合、协同工作。如果没有数据标准，会严重影响企业的正常运行。因此，在大数据行业中对数据全生命周期进行规范化管理，可以从根本上解决诸多的数据问题。

2. 数据标准的分类

数据标准是进行数据标准化、消除数据业务歧义的主要参考依据。数据标准的分类是从更有利于数据标准的编制、查询、落地和维护的角度进行考虑的。数据标准一般包括 3 个要素，即标准分类、标准信息项（标准内容）和相关公共代码（例如国别代码、邮政编码）。数据标准通常可分为基础类数据标准和指标类数据标准。

1）基础类数据标准

基础类数据标准是为了统一企业所有业务活动相关数据的一致性和准确性，解决业务间的数据一致性和数据整合，按照数据标准管理过程制定的数据标准。基础类数据标准一般包括数据维度标准、主数据标准、逻辑数据模型标准、物理数据模型标准、元数据标准、公共代码标准等。表 3-3 为行业参考模型实体数据标准体系定义的内容，表 3-4 为公共代码标准体系定义的内容。

表 3-3　行业参考模型实体数据标准体系定义的内容

行业参考模型实体标准	标准体系属性说明
数据标准编码	根据数据标准编码的规则进行编写
标准主题	数据标准的归属主题
标准子类	数据标准的归属类型
中文名称	数据标准的中文名称
英文名称	数据标准的英文名称
实体编号	根据行业参考模型实体编号的规则进行编写
实体名称	根据行业参考模型实体名称的命名规则进行编写
数据版本	数据标准的版本信息
数据体系分类	根据数据分类规则对数据进行分类，以保证数据体系的易用性，以及符合用户的查找习惯
重要级别	集团规范定义的数据为一级，省公司定义的数据为二级，其他常用的数据为三级
数据提供部门	该数据标准定义数据的提供部门
数据提供部门负责人	该数据标准定义数据的提供部门负责人
数据维护部门	该数据标准定义数据的维护部门
数据维护部门负责人	该数据标准定义数据的维护部门负责人
业务主管部门	该数据标准定义数据的业务主管部门，该部门对数据口径、编码取值和相关专业术语有决定权
业务主管部门负责人	该数据标准定义数据的业务主管部门负责人
数据来源系统	例如 BOSS、CRM、ERP 等
主要依据	关于指标的解释和描述文件。例如集团规范、省公司规范、业务部门制定的规范等
业务定义	指标的业务描述口径，一般由业务部门使用业务语言制定

表 3-4 公共代码标准体系定义的内容

公共代码标准	标准体系属性说明
数据标准编码	根据数据标准编码的规则进行编写
公共标准号	引入外部公共标准号
中文标准名称	数据标准的中文名称
英文标准名称	数据标准的英文名称
标准状态	该标准的状态，例如现行、停止
公共标准机构名称	引入该公共标准的机构名称
数据体系分类	根据数据分类规则对数据进行分类，以保证数据体系的易用性，以及符合用户的查找习惯
重要级别	集团规范定义的数据为一级，省公司定义的数据为二级，其他常用的数据为三级
数据标准引入部门	该数据标准的引入和维护部门
数据标准引入部门负责人	该数据标准的引入和维护部门负责人
数据上报系统	最终对数据进行计算和发布的系统，也是各部门唯一获取指标数据的来源系统

2）指标类数据标准

指标类数据标准一般分为基础指标标准和计算指标（又称组合指标）标准。基础指标具有特定业务和经济含义，且仅能通过基础类数据加工获得，计算指标通常由两个以上的基础指标计算得出。并非所有基础类数据和指标类数据都应纳入数据标准的管辖范围，数据标准管辖的数据，通常只是需要在各业务条线、各信息系统之间实现共享和交换的数据，以及为满足监控机构、上级主管部门、各级政府部门的数据报送要求而需要的数据。

在基础类数据标准和指标类数据标准框架下，可以根据各自的业务主题进行细分。在细分时应尽可能做到涵盖企业的主要业务活动，且涵盖企业生产系统中产生的所有业务数据。

3. 数据标准管理

数据标准管理是指数据标准的制定和实施的一系列活动，关键活动如下：

（1）理解数据标准化需求。

（2）构建数据标准体系和规范。

（3）规划制定数据标准化的实施路线和方案。

（4）制定数据标准管理办法和实施流程要求。

（5）建设数据标准管理工具，推动数据标准的执行落地。

（6）评估数据标准化工作的开展情况。

数据标准管理的目标是通过统一的数据标准制定和发布，结合制度约束、系统控制等手段，实现大数据平台数据的完整性、有效性、一致性、规范性、开放性和共享性管理，为数据资产管理活动提供参考依据。

4. 建设数据标准的好处

通过数据标准的建设，可以有效消除数据跨系统的非一致性，从根源上解决数据定义和使用的不一致问题，为企业数据建设带来诸多好处。

（1）数据标准的统一制定与管理，可保证数据定义和使用的一致性，促进企业级单一数据视图的形成，促进信息资源共享。

（2）通过评估已有系统标准建设情况，可及时发现现有系统标准问题，支撑系统改造，减少数据转换，促进系统集成，提高数据质量。

（3）数据标准可作为新建系统的参考依据，为企业系统建设的整体规划打好基础，减少系统建设工作量，保障新建系统完全符合标准。

3.2.2　数据标准的建设

1. 数据标准的建设过程

数据标准建设大致分为 5 个步骤，即数据标准规划、数据标准编制、标准评审发布、标准执行落地及标准维护增强。

1）数据标准规划

从实际情况出发，结合业界经验，收集国家标准、现行标准、新系统需求标准及行业通行标准等，梳理出数据标准建设的整体范围，定义数据标准体系框架和分类，并制定数据标准的实施计划。值得注意的是，不是所有的数据都需要建立数据标准，企业实际数据模型中有上万个字段，有些模型还会经常变换更新，没有必要将这些信息全部纳入标准体系中，仅需对核心数据建立标准并落地，即可达到预期效果，同时也提升了工作效率。

在规划过程中需要注意以下几点：

（1）共享性高、使用频率高的字段需要入标。

（2）监管报送或发文涉及的业务信息需要入标。

（3）结合数据使用情况，对于关键数据的字段尽量入标。

（4）数据应用有使用需求的字段需要入标。

2）数据标准编制

数据标准管理办公室根据数据需求展开数据的编制工作、确定数据项，数据标准管理执行组根据所需数据项提供数据属性信息，例如数据项的名称、编码、类型、长度、业务含义、数据来源、质量规则、安全级别、值域范围等。数据标准管理办公室对这些数据项进行标准化定义形成初稿，并提交审核。

表 3-5～表 3-7 显示了数据标准的编制。例如企业在编制员工信息表的时候，需要把表 3-5 中的员工信息与表 3-6 中的民族编码标准及表 3-7 中的学历编码标准一一对应。

表 3-5　员工信息表

姓　名	性　别	民　族	学　历	专　业
张明	男	汉族	博士	计算机
张佳	女	汉族	硕士	机械
郑剑	男	苗族	本科	计算机
夏娟	女	汉族	本科	电子

表 3-6　民族编码表

编　码	名　称
1	汉族
2	回族
3	蒙古族
4	苗族
5	傣族
…	

表 3-7 学历编码表

编　码	名　称
1	博士
2	硕士
3	本科
4	专科
5	高中

3）标准评审发布

数据标准管理委员会对数据标准初稿进行审核，判断数据标准是否符合企业的应用和管理需求，是否符合企业数据的战略要求。如果数据标准审查不通过，则由数据标准管理办公室进行修订，直到满足企业数据标准的发布要求为止。标准通过审查后，由数据标准管理办公室面向全公司进行数据标准的发布。在该过程中数据标准管理执行组需要配合进行数据标准发布对现有应用系统、数据模型的影响的评估，并做好相应的应对策略。

4）标准执行落地

把已定义的数据标准与业务系统、应用和服务进行映射，标明标准和现状的关系及可能影响到的应用。在该过程中，对于企业新建的系统应当直接应用定义好的数据标准，对于旧系统则建议建立相应的数据映射关系，进行数据转换，逐步进行数据标准的落地。

5）标准维护增强

数据标准后续可能会随着业务的发展变化、国标行标的变化及监管要求的变化需要不断更新和完善。在数据标准维护阶段需要对标准变更建立相应的管理流程，并做好标准版本管理。

2. 数据标准的建设案例

图 3-5 所示为某银行建设的数据标准案例，在图中数据标准简称为数标。

在建设数据标准时，从软件生命周期来看，一般有以下几个步骤：

（1）需求分析。

（2）软件设计。

（3）软件开发。

（4）测试上线。

（5）运行维护。

从数据标准落地流程来看，一般有以下几个步骤：

（1）数据标准引用需求。

（2）模型设计审批。

（3）IT 开发数据标准修改流程。

（4）录入数据标准与元数据映射。

（5）运形态模型采集。

值得注意的是，在建设数据标准时应以落地实施为目的，并以国家、行业标准为基础，结合现有 IT 系统的现状，以对现有生产系统的影响最小为原则进行编制，确保标准切实可用，并最终让数据标准回归到业务中，发挥价值。

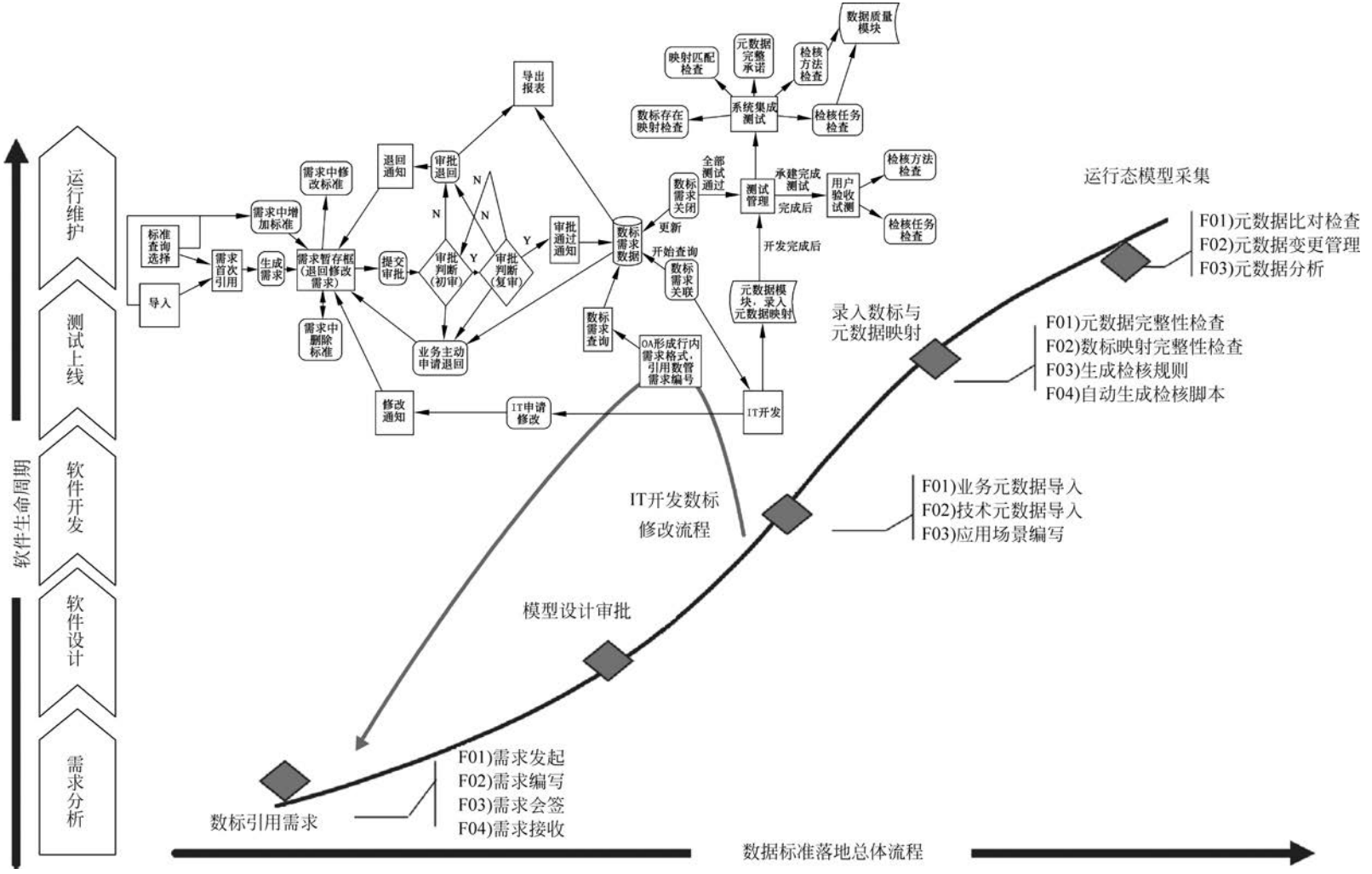

运行维护
测试上线
软件开发
软件设计
需求分析
软件生命周期
标准查询选择
导入
需求首次引用
生成需求
需求暂存柜(退回修改需求)
需求中增加标准
需求中修改标准
需求中删除标准
提交审批
审批判断(初审)
审批退回
退回通知
审批判断(复审)
审批通过通知
导出报表
业务主动申请退回
修改通知
IT申请修改
数标需求数据
更新
开始查询
数标需求关闭
数标需求关联
数标需求查询
OA形成行内需求格式，引用数管需求编号
IT开发
开发完成后
测试管理
全部测试通过
承建完成测试
完成后
系统集成测试
用户验收试测
检核方法检查
检核任务检查
映射匹配检查
元数据完整承诺
检核方法检查
数据质量模块
数标存在映射检查
元数据模块，录入元数据映射
数标引用需求
F01)需求发起
F02)需求编写
F03)需求会签
F04)需求接收
模型设计审批
IT开发数标修改流程
F01)业务元数据导入
F02)技术元数据导入
F03)应用场景编写
录入数标与元数据映射
F01)元数据完整性检查
F02)数标映射完整性检查
F03)生成检核规则
F04)自动生成检核脚本
运行态模型采集
F01)元数据比对检查
F02)元数据变更管理
F03)元数据分析
数据标准落地总体流程

图 3-5　数据标准案例

3.3 主数据与元数据

3.3.1 主数据介绍

1. 认识主数据

主数据是用来描述企业核心业务实体的数据，它是具有高业务价值的、可以在企业内跨越各个业务部门被重复使用的数据，并且存在于多个异构的应用系统中。

由于主数据是具有共享性的基础数据，可以在企业内跨越各个业务部门被重复使用，所以通常长期存在且应用于多个系统。另外，主数据是企业基准数据，数据来源单一、准确、权威，具有较高的业务价值，因此是企业执行业务操作和决策分析的数据标准。

需要注意的是，主数据不是企业内所有的业务数据，只有有必要在各个系统间共享的数据才是主数据，比如大部分的交易数据、账单数据等都不是主数据，而描述核心业务实体的数据，像客户、供应商、账户、组织单位、员工、合作伙伴、位置信息等都是主数据。因此，主数据通常是企业内能够跨业务重复使用的高价值的数据，这些主数据在进行主数据管理之前经常存在于多个异构或同构的系统中。

主数据可以包括很多方面，除了常见的客户主数据之外，不同行业的客户还可能拥有其他各种类型的主数据。例如，对于电信行业客户而言，电信运营商提供的各种服务可以形成其产品主数据；对于航空业客户而言，航线、航班是其企业主数据的一种。对于某个企业的不同业务部门，其主数据也不同，例如市场销售部门关心客户信息，产品研发部门关心产品编号、产品分类等产品信息，人事部门关心员工机构、部门层次关系等信息。

在企业数据中涉及企业经营的人、财、物的数据最有可能纳入企业主数据管理的范畴，主要包括以下内容：

(1) 企业产品及其相关信息，例如企业相关产品、服务、版本、价格、标准操作等。

(2) 企业财务信息，例如业务、预算、利润、合同、财务科目等。

(3) 企业利益相关者，例如客户、供应商、合作伙伴、竞争对手等。

(4) 企业组织架构，例如员工、部门等。

由此可见，主数据就是企业被不同运营场合反复引用的关键的状态数据，它需要在企业范围内保持高度一致。主数据可以随着企业的经营活动而改变，例如客户的增加、组织架构的调整、产品下线等，但是主数据的变化频率应该是较低的。所以，企业运营过程中产生的过程数据，例如订购记录、消费记录等，一般不会纳入主数据的范围。

2. 主数据的特征

主数据具有以下几个特征。

(1) 超越部门。主数据是组织范围内共享的、跨部门的数据，不归属于某一特定的部门而归属于整个组织，是企业的核心数据资产。

(2) 超越业务。主数据是跨越了业务界限，在多个业务领域中被广泛使用的数据，其核心属性也来自业务。主数据在各个业务流程中都是唯一识别的对象，它不会依赖于业务流程存在，但它的价值是在业务交互中体现的。

(3) 超越系统。主数据是多个系统之间的共享数据，是应用系统建设的基础，同时也是数据分析系统重要的分析对象。因此，主数据应该保持相对独立，服务于但要高于其他业务信息系统。

（4）超越技术。主数据是要解决不同异构系统之间的核心数据的共享问题，应当满足在不同业务系统架构下使用的情况，兼容多种系统架构，提供较多的数据接收及应用方式，不会局限于一种特定的技术。

3. 主数据管理概述

主数据通常需要在整个企业范围内保持一致性（consistent）、完整性（complete）、可控性（controlled），为了达成这一目标，需要进行主数据管理（Master Data Management，MDM）。集成、共享、数据质量、数据治理是主数据管理的四大要素。主数据管理要做的就是从企业的多个业务系统中整合最核心的、最需要共享的数据（主数据），集中进行数据的清洗和丰富，并且以服务的方式把统一的、完整的、准确的、具有权威性的主数据分发给企业范围内需要使用这些数据的操作型应用和分析型应用，具体包括各个业务系统、业务流程和决策支持系统等。

MDM 一方面可以保障主数据的规范性和唯一性。按规则和流程规范管理主数据，比如规定主数据名称要使用营业执照上的名称，社会统一信用代码、国别地区等必填，按姓名、信用代码等条件校验避免重复输入，系统内编码唯一，主数据要经流程审核后方能生效等。另一方面，MDM 使得主数据能够集中管理。主数据全部在 MDM 中产生或者受控，保障来源唯一从而避免歧义。同时，MDM 能够把主数据分发给相关系统，也可以接收外部系统产生的主数据，经处理后再分发出去。

在开始进行主数据管理之前，主数据管理策略应围绕以下 6 个领域构建。

（1）建立组织体系。有效的组织机构是项目成功的有力保证，为了达到项目预期目标，在项目开始之前对于组织及其责任分工做出规划是非常必要的。主数据涉及的范围很广，涉及不同的业务部门和技术部门，是企业的全局大事，如何成立和成立什么样的组织应该依据企业本身的发展战略和目标来确定。在明确了组织机构的同时还要明确主数据管理岗位，例如主数据系统管理员、主数据填报员、主数据审核员、数据质量管理员、集成技术支持员等。主数据管理岗位可以兼职，也可以全职，根据企业的实际情况而定。在整个主数据管理中安排合适的人员，包括主数据所有者、数据管理员和参与治理的人员。

（2）主数据梳理和调研。在进行主数据管理前，应当首先对所在单位信息的采集、处理、传输和使用做全面规划。其核心是运用先进的信息工程和数据管理理论及方法，通过总体数据规划，奠定资源管理的基础，促进实现集成化的应用开发，构建信息资源网，让企业能够对现有数据资源有一个全面、系统的认识。特别是通过对职能域之间交叉信息的梳理，使人们更加清晰地了解企业信息的来龙去脉，有助于人们把握各类信息的源头，有效地消除“信息孤岛”和数据冗余，控制数据的唯一性和准确性，确保所获取信息的有效性。在这个过程中，需要在既定的数据范围内摸透企业主数据的管理情况、数据标准情况、数据质量情况、数据共享情况等。这种方法适用于包含咨询的主数据项目的建设。

（3）建立主数据标准体系。主数据标准体系主要包含主数据分类和编码标准化。没有标准化就没有信息化，主数据分类和编码标准是主数据标准中最基础的标准。数据分类就是根据信息内容的属性或特征，将信息按一定的原则和方法进行区分和归类，并建立起一定的分类系统和排列顺序，以便管理和使用信息。主数据编码就是在信息分类的基础上，将信息对象赋予有一定规律性的、易于计算机和人识别与处理的符号。主数据模型标准化就是根据前期的调研、梳理和评估定义出每个主数据的元模型，明确主数据的属性组成、字段类型、长度、是否唯一、是否必填及校验规则等。

（4）建立评估与管理体系。主数据管理需建立评估体系，主要步骤是根据前期的业务调研情况和数据普查情况确定参评数据范围，准备出参评数据，并依据打分模板进行打分，识别

出企业主数据。目前对于数据管理能力的评估已经有了比较成熟的评价模型，典型的有IBM数据治理成熟度评估模型、SEI数据能力成熟度模型、EDM数据能力成熟度模型、DataFlux数据治理成熟度模型等。表3-8显示了主数据管理的考核评价指标。

表3-8 主数据管理的考核评价指标

序号	考核方向	技术指标	衡量标准
1	及时性	及时率	满足时间要求的数据总数/总数据数
2	真实性和准确性	数据真实率	1－数据中失真记录总数/数据总记录数
		有效值比率	1－超出值域的异常值记录总数/数据总记录数
		流转过程失真率	数据传输失真记录总数/总记录数
		重复数据比率	重复记录数/总记录数
3	一致性	外键无对应主键的记录比率	外键无对应主键的记录总数/总记录数
		主数据一致率	一致的主数据总数/主数据总数
4	完整性	字段的空值率	空值记录总数/总记录数
		信息完备率	能够获取的指标数/总需求指标数

(5) 建立制度与流程体系。制度和流程体系的建设是主数据成功实施的重要保障。制度章程是确保对主数据管理进行有效实施的认责制度。建立主数据管理制度和流程体系时需要明确主数据的归口部门和岗位，明确岗位职责，明确每个主数据的申请、审批、变更、共享的流程。同时做好数据运营工作，定期检查数据质量，进行数据的清洗和整合，实现企业数据质量的不断优化和提升。

(6) 建立技术体系。主数据管理技术体系的建设应从应用层面和技术层面两方面考虑。在应用层面，主数据管理平台需具备元数据(数据模型管理)、数据管理、数据清洗、数据质量、数据集成、权限控制、数据关联分析，以及数据的映射(mapping)/转换(transforming)/装载(loading)能力。在技术层面，重点考虑系统架构、接口规范、技术标准。在主数据管理工具中，IBM InfoSphere MDM是当今市场上功能最强大的主数据管理(MDM)产品，用于处理完整范围的主数据管理需求和用例。为了给客户提供其MDM解决方案需求的最佳范围，IBM InfoSphere MDM有4个版本，即Collaborative Edition、Standard Edition、Advanced Edition及Enterprise Edition，其中，Enterprise Edition版本包含了其他3个版本的所有功能。

4. 主数据管理平台的建设

主数据是企业最基础、最核心的数据，企业的一切业务基本上都是基于主数据来开展的，所以主数据管理成为企业数据治理中最核心的部分。

为了更好地管理主数据，企业经常需要建设主数据管理平台，该平台从功能上主要包括主数据模型、主数据编码、主数据管理、主数据清洗、主数据质量和主数据集成等。

(1) 主数据模型提供主数据的建模功能，管理主数据的逻辑模型和物理模型等。

(2) 主数据编码支持各种形式主数据的编码，提供数据编码申请、审批、集成等服务。编码功能是主数据产品的初级形态，也是主数据产品的核心能力。

(3) 主数据管理主要提供主数据的增/删/改/查功能。

(4) 主数据清洗主要包括主数据的采集、转换、清理、装载等功能。

(5) 主数据质量主要提供主数据从质量问题发现到质量问题处理的闭环管理功能。

(6) 主数据集成主要提供主数据采集和分发服务，完成与企业其他异构系统的对接。

图3-6显示了某公司使用主数据管理平台(MDM平台)对输入数据进行管理。图3-7显示了某公司使用主数据平台为应用服务提供数据服务。图3-8显示了某高校建设的主数据管理

平台的整体架构，主要包含数据集成层、数据存储层和接口层。

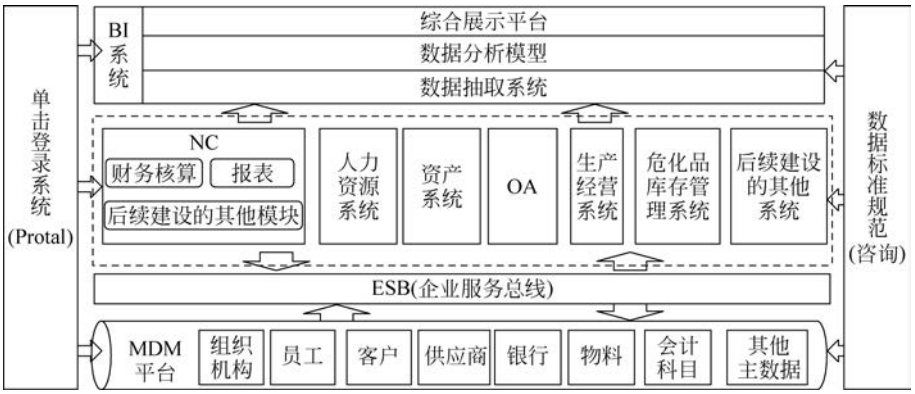


图 3-6　主数据管理平台

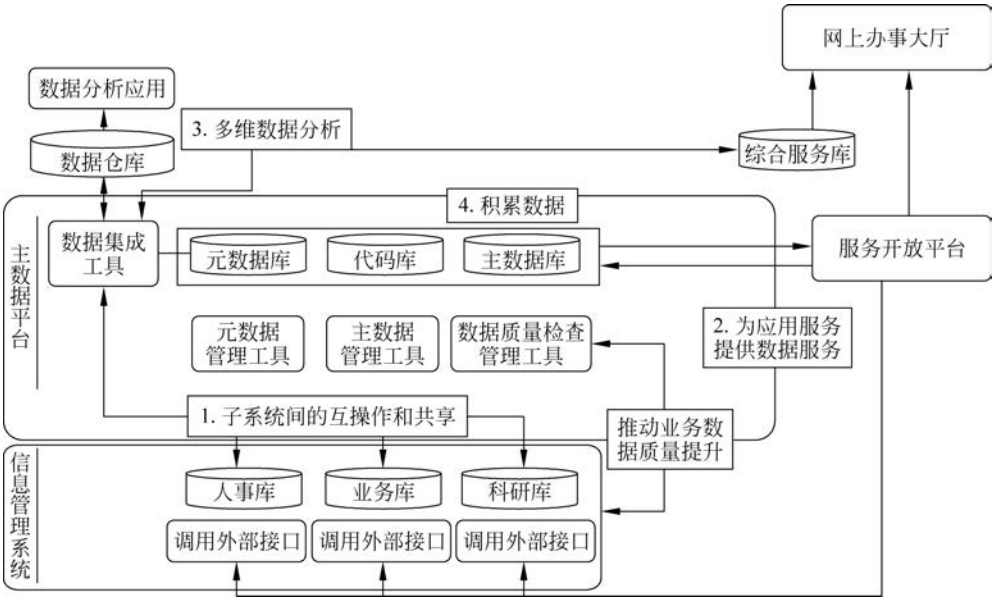


图 3-7　主数据平台

3.3.2　元数据概述

1. 认识元数据

元数据是描述企业数据的相关数据(包括对数据的业务、结构、定义、存储、安全等各方面的描述)，一般是指在 IT 系统建设过程中所产生的与数据定义、目标定义、转换规则等相关的关键数据，在数据治理中具有重要的地位。

元数据不仅仅表示数据的类型、名称、值等信息，它可以理解为一组用来描述数据的信息组/数据组，该信息组/数据组中的一切数据、信息都描述/反映了某个数据的某方面特征，则该信息组/数据组可称为一个元数据。例如，元数据可以为数据说明其元素或属性(名称、大小、数据类型等)，或其结构(长度、字段、数据列)，或其相关数据(位于何处、如何联系、拥有者)。在日常生活中，元数据无所不在。只要有一类事物，就可以定义一套元数据。

一般来讲，元数据主要用来描述数据属性的信息，例如记录数据仓库中模型的定义、各层级间的映射关系、监控数据仓库的数据状态及 ETL 的任务运行状态等。因此，元数据是对数

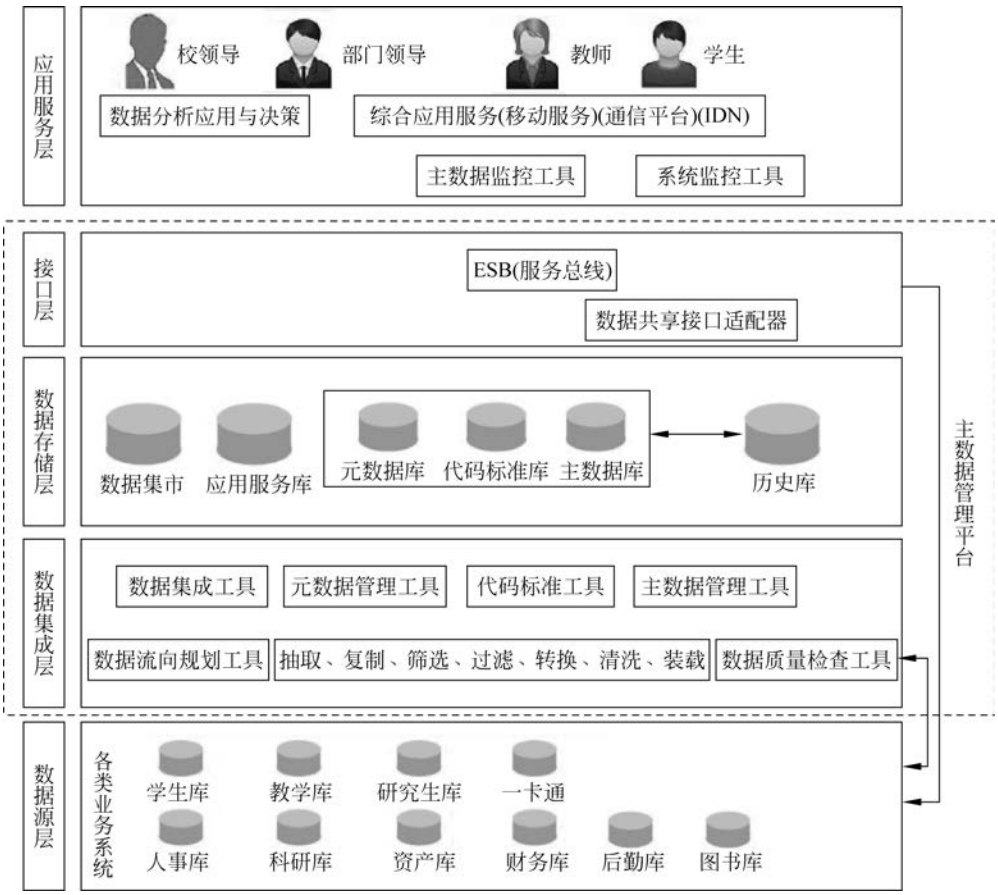


图 3-8 主数据管理平台的整体架构

据本身进行描述的数据，或者说它不是对象本身，它只描述对象的属性，就是一个对数据自身进行描绘的数据。例如，人们网购，想要买一件衣服，那么衣服就是数据，而所挑选衣服的色彩、尺寸、做工、样式等属性就是它的元数据。

又例如，有一条学生信息记录，其中包括字段姓名(name)、年龄(age)、性别(gender)、班级(class)等，那么姓名、年龄、性别、班级就是元数据。通过它们的描述，一条关于学生信息的数据记录就产生了。

再例如，在电影数据库 IMDB 上可以查到每部电影的信息。IMDB 本身也定义了一套元数据，用来描述每部电影。下面是它的元数据，可以从多方面刻画一部电影：Cast and Crew(演职人员)、Company Credits(相关公司)、Basic Data(基本情况)、Plot and Quotes(情节和引语)、Fun Stuff(趣味信息)、Links to Other Sites(外部链接)、Box Office and Business(票房和商业开发)、Technical Info(技术信息)、Literature(书面内容)、Other Data(其他信息)。

在实际应用中记录元数据时经常使用以下指标。

1) 总体概况

标题：生成它的数据集或项目的名称。

创建者：创建数据的组织或人员的姓名和地址，个人姓名的首选格式是姓氏。

标识符：用于标识数据的唯一编号，即使只是内部项目参考编号。

日期：与数据关联的关键日期，包括项目开始和结束日期、发布日期、数据涵盖的时间段，以及与数据生命周期相关的其他日期、例如维护周期、更新时间，其首选格式为 yyyy-mm-dd 或 yyyy. mm. dd-yyyy. mm. dd。

方法：如何生成数据，列出所使用的设备和软件（包括模型和版本号）、公式、算法等。

处理：数据如何被更改或处理（例如标准化）。

来源：来自其他来源的数据的引用，包括源数据的保存位置和访问方式的详细信息。

2）内容说明

主题：描述数据主题或者内容的关键字或短语。

地点：所有适用的物理位置。

语言：数据集中使用的所有语言。

变量列表：数据文件中的所有变量（如果适用）。

代码列表：文件名中使用的代码或缩写的说明或数据文件中的变量。

3）技术说明

文件清单：与项目关联的所有文件，包括扩展名。

文件格式：数据格式，例如 FITS、SPSS、HTML、JPEG 等。

文件结构：数据文件的组织和变量的布局（如果适用）。

版本：每个版本的唯一日期/时间戳和标识符。

权利：任何已知的知识产权、法定权利、许可或数据使用限制。

访问信息：可以在何处及如何访问该数据。

【例 3-1】 在 Tableau 中查看数据和元数据。

（1）下载并安装 Tableau。

（2）运行 Tableau，在已保存的数据源中选择“示例-超市”，如图 3-9 所示。

图 3-9　选择数据源

（3）选中“预览数据源”选项，查看“示例-超市”中的数据，如图 3-10 所示。

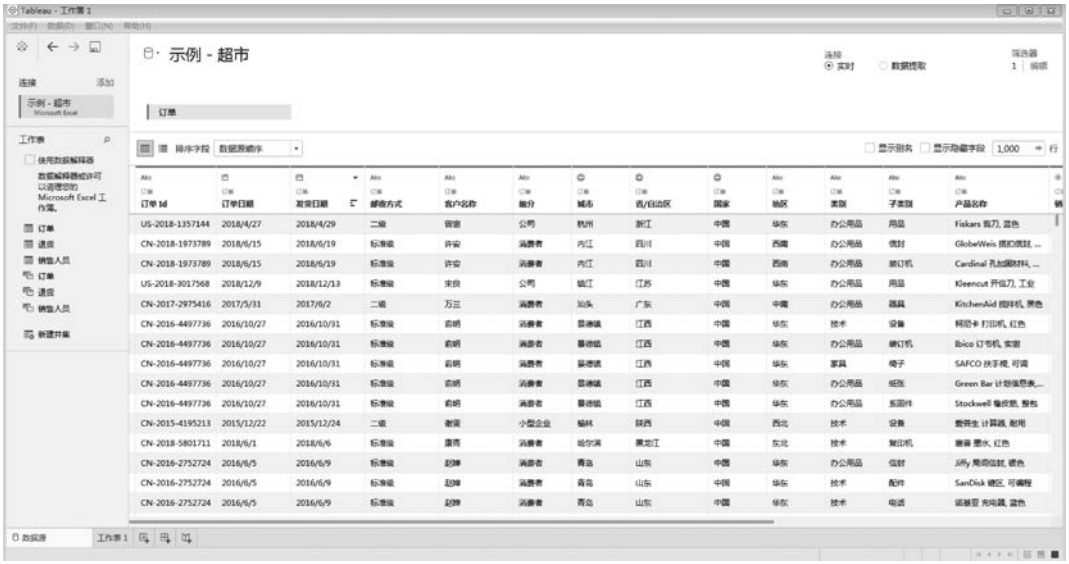


图 3-10　查看“示例-超市”中的数据

（4）选中“管理元数据”选项，查看“示例-超市”中的元数据，如图 3-11 所示。

（5）选中“订单日期”字段，在右侧的下拉菜单中选择“描述”，可查看该字段的描述信息，如图 3-12 所示。

图 3-11 查看“示例-超市”中的元数据

图 3-12 查看“订单日期”字段的描述信息

【例 3-2】 在 MySQL 中查看数据表的元数据。

(1) 运行 MySQL,输入命令“show databases;”,查看已创建好的数据库,如图 3-13 所示。

(2) 输入命令“use stu;”,选中 stu 数据库,如图 3-14 所示。

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| company            |
| hy                 |
| library            |
| mysql              |
| performance_schema |
| stu                |
| student            |
| test               |
| user               |
+--------------------+
10 rows in set (0.54 sec)
```

图 3-13　查看数据库

```
mysql> use stu;
Database changed
mysql> show tables;
+---------------+
| Tables_in_stu |
+---------------+
| st            |
| user          |
| user2         |
+---------------+
3 rows in set (0.00 sec)
```

图 3-14　选中 stu 数据库

(3) MySQL 用 show 语句获取元数据是最常用的方法。这里输入命令“show table status like 'user'\G;”查看数据表 user 的元数据(描述性信息),如图 3-15 所示。

```
mysql> show table status like 'user' \G;
*************************** 1. row ***************************
           Name: user
         Engine: InnoDB
        Version: 10
     Row_format: Compact
           Rows: 4
 Avg_row_length: 4096
    Data_length: 16384
Max_data_length: 0
   Index_length: 0
      Data_free: 0
 Auto_increment: NULL
    Create_time: 2018-12-25 22:35:55
    Update_time: NULL
     Check_time: NULL
      Collation: latin1_swedish_ci
       Checksum: NULL
 Create_options:
        Comment:
1 row in set (0.16 sec)
```

图 3-15　查看数据表 user 的元数据

【例 3-3】 在 MySQL 中查看 information_schema 数据库中的元数据。

(1) 运行 MySQL。

(2) information_schema 数据库是 MySQL 自带的信息数据库,主要用于存储数据库中的元数据(关于数据的数据),例如数据库名、表名、列的数据类型、访问权限等。这里输入命令“use information_schema;”和“show tables;”查看 information_schema 中的数据表,如图 3-16 所示。

(3) 输入命令“show character set;”查看 MySQL 中可用字符集的信息,如图 3-17 所示。

(4) 输入命令“select * from schemata;”查看当前 MySQL 实例中所有数据库的信息,如图 3-18 所示。

此外,用户也可以用代码来查看元数据,在 Kafka 中代码如下:

```
private final long metadataExpireMs;        //自动更新元数据,默认 5 分钟一次
private int version;                        //对于 Producer 端来说元数据是有版本号的,每次
                                            //更新元数据都会更新一下版本号
private long lastRefreshMs;                 //上一次更新元数据的时间
private long lastSuccessfulRefreshMs;       //上一次成功更新元数据的时间,可能有时更新不成功
private Cluster cluster;                    //Kafka 集群本身的元数据消息
private boolean needUpdate;                 //标识,用来判断是否更新元数据的标识之一
private final Map<String, Long> topics;     //存放当前所有的 topic
```

```
mysql> use information_schema;
Database changed
mysql> show tables;
+---------------------------------------+
| Tables_in_information_schema          |
+---------------------------------------+
| CHARACTER_SETS                        |
| COLLATIONS                            |
| COLLATION_CHARACTER_SET_APPLICABILITY |
| COLUMNS                               |
| COLUMN_PRIVILEGES                     |
| ENGINES                               |
| EVENTS                                |
| FILES                                 |
| GLOBAL_STATUS                         |
| GLOBAL_VARIABLES                      |
| KEY_COLUMN_USAGE                      |
| OPTIMIZER_TRACE                       |
| PARAMETERS                            |
| PARTITIONS                            |
| PLUGINS                               |
| PROCESSLIST                           |
| PROFILING                             |
| REFERENTIAL_CONSTRAINTS               |
| ROUTINES                              |
| SCHEMATA                              |
| SCHEMA_PRIVILEGES                     |
| SESSION_STATUS                        |
| SESSION_VARIABLES                     |
| STATISTICS                            |
| TABLES                                |
| TABLESPACES                           |
| TABLE_CONSTRAINTS                     |
```

图 3-16　查看 information_schema 中的数据表

```
mysq1> show character set;
+----------+-----------------------------+---------------------+--------+
| Charset  | Description                 | Default collation   | Maxlen |
+----------+-----------------------------+---------------------+--------+
| big5     | Big5 Traditional Chinese    | big5_chinese_ci     |      2 |
| dec8     | DEC West European           | dec8_swedish_ci     |      1 |
| cp850    | DOS West European           | cp850_general_ci    |      1 |
| hp8      | HP West European            | hp8_english_ci      |      1 |
| koi8r    | KOI8-R Relcom Russian       | koi8r_general_ci    |      1 |
| latin1   | cp1252 West European        | latin1_swedish_ci   |      1 |
| latin2   | ISO 8859-2 Central European | latin2_general_ci   |      1 |
| swe7     | 7bit Swedish                | swe7_swedish_ci     |      1 |
| ascii    | US ASCII                    | ascii_general_ci    |      1 |
| ujis     | EUC-JP Japanese             | ujis_japanese_ci    |      3 |
| sjis     | Shift-JIS Japanese          | sjis_japanese_ci    |      2 |
| hebrew   | ISO 8859-8 Hebrew           | hebrew_general_ci   |      1 |
| tis620   | TIS620 Thai                 | tis620_thai_ci      |      1 |
| euckr    | EUC-KR Korean               | euckr_korean_ci     |      2 |
| koi8u    | KOI8-U Ukrainian            | koi8u_general_ci    |      1 |
| gb2312   | GB2312 Simplified Chinese   | gb2312_chinese_ci   |      2 |
| greek    | ISO 8859-7 Greek            | greek_general_ci    |      1 |
| cp1250   | Windows Central European    | cp1250_general_ci   |      1 |
| gbk      | GBK Simplified Chinese      | gbk_chinese_ci      |      2 |
| latin5   | ISO 8859-9 Turkish          | latin5_turkish_ci   |      1 |
| armscii8 | ARMSCII-8 Armenian          | armscii8_general_ci |      1 |
| utf8     | UTF-8 Unicode               | utf8_general_ci     |      3 |
| ucs2     | UCS-2 Unicode               | ucs2_general_ci     |      2 |
| cp866    | DOS Russian                 | cp866_general_ci    |      1 |
| keybcs2  | DOS Kamenicky Czech-Slovak  | keybcs2_general_ci  |      1 |
| macce    | Mac Central European        | macce_general_ci    |      1 |
| macroman | Mac West European           | macroman_general_ci |      1 |
| cp852    | DOS Central European        | cp852_general_ci    |      1 |
| latin7   | ISO 8859-13 Baltic          | latin7_general_ci   |      1 |
| utf8mb4  | UTF-8 Unicode               | utf8mb4_general_ci  |      4 |
```

图 3-17　查看可用字符集的信息

2. 电子文件元数据

电子文件的形成、捕获、登记、分类、存储和保管、利用、跟踪、处置、传输、归档移交及长期保存等都需要记录在元数据中，并应保持连续、一致，以确保电子文件的真实性、完整性与有效性。因此，电子文件元数据是描述电子文件数据属性的数据，包括文件的格式、编排结构、硬件和软件环境、文件处理软件、字处理软件和图形处理软件、字符集等数据。此外，电子文件元数

```
mysql1> select * from schemata;
+--------------+--------------------+----------------------------+--------------
----------+----------+
| CATALOG_NAME | SCHEMA_NAME        | DEFAULT_CHARACTER_SET_NAME | DEFAULT_COLLA
TION_NAME | SQL_PATH |
+--------------+--------------------+----------------------------+--------------
----------+----------+
| def          | information_schema | utf8                       | utf8_general_
ci        | NULL     |
| def          | company            | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | hy                 | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | library            | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | mysql              | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | performance_schema | utf8                       | utf8_general_
ci        | NULL     |
| def          | stu                | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | student            | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | test               | latin1                     | latin1_swedis
h_ci      | NULL     |
| def          | user               | utf8                       | utf8_general_
ci        | NULL     |
+--------------+--------------------+----------------------------+--------------
----------+----------+
```

图 3-18　查看所有数据库的信息

据描述的数字对象为通用的电子文件核心元数据，主要为原生电子文件与数字化文件（文本、图像）元数据。

1）电子文件元数据模型

电子文件元数据模型的建立是以文件连续体理论为基础的。文件作为交流、传递、存储、利用信息的工具，其生成、处理、运转必然与文件责任者处理某项事务相关。对该事务的办理，形成文件的业务活动，构成了文件的来源。这种业务活动构成了文件的背景。文件管理业务系统的各个流程需要通过元数据实现对文件或档案的管理。

电子文件元数据体系由一系列元素组成，元素之间的相互关系形成了元数据的结构。元数据的结构与所描述及管理的资源对象的特性相关，并与元数据规范的设计思想与相关抽象模型相关。在电子文件元数据模型中，元数据的用途之一是用来描述业务系统中的实体。关键的实体如下：

- 文件实体。文件本身，不管是单份文件还是文件集合体。
- 责任者实体。业务环境中的人或组织结构。
- 业务实体。业务办理。

通常，可以将元数据分为下列几类：关于文件自身的元数据、关于责任者的元数据、关于业务工作或过程的元数据、关于业务规章制度与政策及法规的元数据、关于文件管理过程的元数据。

在电子文件元数据标准中，元数据元素的语义构成见表 3-9，元数据文件主体见表 3-10，元数据文件摘要见表 3-11，元数据文件日期见表 3-12。

表 3-9　元数据元素的语义构成

元 素 名 称	元 素 描 述
定义	对元素概念与内涵的说明
用途	表明元素的作用
必备性	说明元素必选、可选或条件必选
可重复性	说明元素是否可以重复出现

续表

元素名称	元素描述
取值范围	元素取值的允许范围，有可能从编码体系中获取
适用性	元素适用范围
限定元素	对现有的元素语义进行细化或者限定
默认值	一般情况下元素的取值
使用条件	使用该元素需满足的条件
来源	元素取值的信息来源
注释	对元素的补充说明

表 3-10 元数据文件主体

定义	用于表达文件主题内容的规范化词或词组。关键词是在标引和检索中取自文件和题名或正文用于表达文件主题并具有检索意义的词或词组			
用途	简略概括文件内容主题并便于检索利用，便于按主题进行文件组合			
必备性	必选			
可重复性	可重复			
取值范围	主题词编码表或自由文本。例如《公文主题词表》《档案主题词表》《中国档案分类表》			
适用性	仅限于文件和文件组合(案卷)主题词的描述，描述系列和全宗时不可选			
限定元素	限定元素名称	取值范围	必备性	可重复性
	1.5.1 主题词或关键词	主题词编码表或自由文本	必选	可重复
	1.5.2 次关键词	主题词编码表或自由文本	可选	可重复
	1.5.3 第三关键词	主题词编码表或自由文本	可选	可重复
默认值	无			
使用条件	无			
来源	在创建或处理文件实体时产生			
著录细则	由文件创建者或处理人员手工著录，或根据主题词表选择著录			
注释	主题词建议选择相关的编码体系，例如档案主题词表(CCS)、中国档案分类表(CAC)。如果选择编码体系中的主题词或分类号，则必须在属性中标明编码体系名称。 关键词则是没有经过规范的词或词组			

表 3-11 元数据文件摘要

定义	对文件或文件组合内容的摘录、解释、附录及说明，能够反映主题内容、重要数据(包括技术参数等)
用途	便于对文件的了解、检索和利用
必备性	可选
可重复性	可重复
适用性	适用于文件实体的所有类型
默认值	无
使用条件	无
来源	在文件实体生成时由处理人员著录，或由档案管理人员著录
著录细则	需要文件创建者或处理人员手工著录。 如果文件用于描述单个文件，则著录为文件提要； 如果文件用于描述文件组合(案卷)，则著录案卷描述信息； 如果文件用于描述类别(系列)，则著录类别(系列)说明； 如果文件用于描述全宗，则著录全宗指南
注释	扼要介绍文件内容要点，指出文件的价值、特点、可靠程度等，要求评述中肯、文字简洁，不是文件题名的简单重复

表 3-12　元数据文件日期

定义	与文件生命周期中某一事件相关的时间			
用途	提供对创建、登记和处理行为的系统确认； 提供文件真实性的证明； 限制或帮助对文件的获取； 提供对文件适当和可靠的管理			
必备性	必选			
可重复性	可重复			
取值范围	ISO 8601			
适用性	适用于文件实体的所有类型			
限定元素	限定元素名称	取值范围	必备性	可重复性
	1.7.1 创建日期	ISO 8601	必选	不可重复
	1.7.2 登记日期	ISO 8601	必选	不可重复
	1.7.3 传输日期	ISO 8601	可选	可重复
默认值	创建日期：文件创建时的系统日期/时间 登记日期：文件登记时的系统日期/时间			
使用条件	当文件实体为文件时，属性 Type 的值可以为创建日期、登记日期、传输日期、归档日期； 当文件实体为案卷时，属性 Type 的值可以为创建日期、归档日期； 当文件实体为系列或全宗时，属性 Type 的值只为创建日期			
来源	来源于文件实体创建、登记、传输与归档的过程。在多数情况下，文件时间元数据是由系统生成的			

2）电子文件元数据的语法

电子文件元数据的语法（句法）是一个形式化描述的问题，即将元数据规范体系的所有语义、结构及描述的内容以人可读或计算机可读的形式化方式描述出来，从标准、开放、互操作角度，采用标记语言对元数据集进行描述，其中 XML 标记语言的应用较多。

元数据形式化描述包括两方面的内容，一是有关元数据规范的定义与描述；二是有关元数据记录的描述。从系统应用的角度来说，前者如数据词典或数据库结构，后者则为数据记录。因内容与要求不同，两者可采用不同的描述方法。

从描述元数据规范来说，主要有 DTD、XML Schema 和 RDFS3 种方法。其中，DTD 是通过 SGML 应用程序来使用的，但存在描述能力不强、重用的代价相对较高等缺点；XML Schema 是对 DTD 的扩展，采用了 XML 形式来定义描述 XML 文档的结构，因此可以很方便地利用 XML 解析器与相关工具进行处理，并且通过引入数据类型，大大提高了对数据的描述能力；RDFS 采用基于 RDF 的语法来进行 RDF 规范的描述，更多地用于描述属性及它们的意义与关系等。

由于 XML 具有过多的灵活性，在格式正确的前提下，对于元数据记录的描述有多种可能性，但灵活性对不同行业不同元数据规范之间的互操作具有负面作用。RDF 不仅具有清晰的描述结构，还具有较强的描述元数据结构与语义关系的能力，更适合展现元数据的内容，但存在体积大、增加系统负载等问题。用户在实际应用中要根据需要来选择。

编码不仅是元数据长期保存、互操作的基础，同时也可以在应用中直接作为元数据挖掘与展现的技术平台。在大数据量的实际应用中，鉴于应用的复杂程度与效率之间的矛盾，在系统内部采用自行定义的高效率编码或数据库设计也是一种选择，但前提是内、外接口必须能够支持标准的标记语言编码，以保证系统的互操作能力。

【例 3-4】 电子文件元数据实例。

```
<?xml version = "1.0" encoding = "gb2312" ?>
< saac:信息总体 xmlns:saac = "http://www.gov.cn">
< saac:文件实体>
< saac:文件层级>文件层级</saac:文件层级>
</saac:文件实体>
< saac:标识>
< saac:文件标识码>唯一标识码</saac:文件标识码>
< saac:文件编号>文件编号</saac:文件编号>
</saac:标识>
< saac:题名>
< saac:正题名>正题名</saac:正题名>
< saac:并列题名>并列题名</saac:并列题名>
< saac:副题名及说明题名文字>副题名及说明题名文字</saac:副题名及说明题名文字>
< saac:缩写题名>缩写题名</saac:缩写题名>
</saac:题名>
< saac:分类>
< saac:职能分类>职能分类</saac:职能分类>
< saac:主题分类>主题分类</saac:主题分类>
</saac:分类>
< saac:主题>
< saac:主题词或关键词>主题词或关键词</saac:主题词或关键词>
< saac:次关键词>次关键词</saac:次关键词>
< saac:第三关键词>第三关键词</saac:第三关键词>
</saac:主题>
< saac:语种>中文</saac:语种>
< saac:文种>公文</saac:文种>
< saac:位置>
< saac:当前位置>二级存储</saac:当前位置>
< saac:存储位置>\data\A1\A1 - 14 - 0034 - 117 </saac:存储位置>
< saac:存储说明 />
</saac:位置>
< saac:业务描述>
< saac:业务范围 />
< saac:业务名称 />
< saac:业务说明 />
</saac:业务描述>
</saac:信息总体>
```

3.3.3 元数据管理

1. 元数据管理模型

1）元数据管理概述

元数据管理是数据治理的基础和核心，是构建企业信息单一视图的重要组成部分，元数据管理可以保证在整个企业范围内跨业务竖井协调和重用主数据。元数据管理不会创建新的数据或新的数据纵向结构，而是提供一种方法使企业能够有效地管理分布在整个信息供应链中的各种主数据（由信息供应链各业务系统产生）。

从整个企业层面来说，各种工具软件和应用程序越来越复杂，相互依存度逐年增加，相应地追踪整个信息供应链各组件之间数据的流动、了解数据元素的含义和上下文的需求越来越强烈。在从应用议程向信息议程转变的过程中，元数据管理也逐渐从局部存储和管理转向共享。从总量上来看，整个企业的元数据越来越多，仅现有的数据模型中就包含了成千上万的表，并且还有更多的模型等着上线，同时随着大数据时代的来临，企业需要处理的数据类型越来越多。因此，企业为了更高效地运转，需要明确元数据管理策略和元数据集成体系结构，依

托成熟的方法论和工具实现元数据管理，并有步骤地提升其元数据管理成熟度。

元数据管理一直比较困难，一个很重要的原因就是缺乏统一的标准。在这种情况下，各公司的元数据管理解决方案各不相同。近几年来，随着元数据联盟（Meta Data Coalition，MDC）的开放信息模型（Open Information Model，OIM）和 OMG 组织的公共仓库模型（Common Warehouse Model，CWM）标准的逐渐完善，以及 MDC 和 OMG 组织的合并，为数据仓库厂商提供了统一的标准，从而为元数据管理铺平了道路。

2）元数据管理策略

为了实现大数据治理，构建智慧的分析洞察，企业需要实现贯穿整个企业的元数据集成，建立完整且一致的元数据管理策略，该策略不仅仅针对某个数据仓库项目、业务分析项目、某个大数据项目或某个应用单独制定一个管理策略，而是针对整个企业构建完整的管理策略。元数据管理策略也不是技术标准或某个软件工具可以取代的，无论软件工具的功能多么强大都不能完全替代一个完整一致的元数据管理策略，反而在定义元数据集成体系结构及选购元数据管理工具之前需要定义元数据管理策略。

元数据管理策略需要明确企业元数据管理的愿景、目标、需求、约束和策略等，依据企业自身当前及未来的需要确定要实现的元数据管理成熟度及实现目标成熟度的路线图完成基础本体、任务本体和应用本体的构建，确定元数据管理的安全策略、版本控制及元数据的订阅和推送等。企业需要对业务术语、技术术语中的敏感数据进行标记和分类，制定相应的数据隐私保护政策，确保企业在隐私保护方面符合当地隐私方面的法律、法规，如果企业有跨国数据交换、元数据交换的需求，也要遵循所涉及国家的法律、法规要求。企业需要保证每个元数据元素在信息供应链中的每个组件中语义上保持一致，也就是语义等效。语义等效（平均）可以强也可以弱，在一个元数据集成方案中，语义等效越强则整个方案的效率越高。语义等效的强弱程度直接影响了元数据的共享和重用。

本体（ontology）是元数据管理中的核心概念，是领域概念及概念之间关系的规范化描述，并且这种描述是规范的、明确的、形式化的、可共享的。本体有时也被翻译成本体论，在人工智能和计算机科学领域中的本体最早源于 20 世纪 70 年代中期，随着人工智能的发展，人们发现知识的获取是构建强大人工智能系统的关键，于是开始将新的本体创建为计算机模型，从而实现特定类型的自动化推理。到了 20 世纪 80 年代，人工智能领域开始使用本体表示模型化时间的一种理论及知识系统的一种组件，认为本体（人工智能）是一种应用哲学。目前被人们广泛接受的一个本体定义为“本体是共享概念模型的明确形式化规范说明”。本体提供了一个共享词汇表，可以用来对一个领域建模，具体包括存在的对象或概念的类型，以及它们的属性和关系。随着时间的推移和技术的发展，本体从最开始的人工智能领域逐渐扩展到图书馆学、情报学、软件工程、信息架构、生物医学和信息学等越来越多的学科。本体（人工智能和计算机科学）依赖某种类别体系来表达实体、概念、事件及其属性和关系。一个本体可以由类（class）、关系（relations）、函数（function）、公理（axioms）和实例（instances）5 种元素组成。其中，类也称为概念。本体的核心是知识共享和重用，通过减少特定领域内概念或术语上的分歧，使不同的用户之间可以顺畅地沟通和交流并保持语义等效性，同时让不同的工具软件和应用系统之间实现互操作。

根据研究层次可以将本体划分为顶级本体（top-level ontology）、领域本体（domain ontology）、应用本体（application ontology）和任务本体（task ontology）几种类型。

（1）顶级本体。顶级本体也称为上层本体（upper ontology）或基础本体（foundation ontology），是指独立于具体的问题或领域，在所有领域都适用的共同对象或概念所构成的模

型，主要用来描述高级别且通用的概念及概念之间的关系。顶级本体是指对某个特定的领域建模，显式地实现对领域的定义，确定该领域内共同认可的词汇、词汇业务的含义和对应的信息资产等，提供对该领域知识的共同理解。

(2) 领域本体。领域本体是专业性的本体，在这类本体中被表示的知识是针对特定学科领域的。这类本体描述的词表关系到某一学科领域，例如飞机制造、化学元素周期表等。它们提供了关于某个学科领域中概念的词表及概念之间的关系，或者该学科领域的重要理论。

(3) 应用本体。应用本体描述依赖于特定领域和任务的概念及概念之间的关系，是用于特定应用或用途的本体，其范畴可以通过可测试的用例来指定。

(4) 任务本体。任务本体是针对任务元素及其之间关系的规范说明或详细说明，用来解释任务存在的条件及可以被用在哪些领域或环境中，是一个通用术语的集合，用来描述关于任务的定义和概念等。

3) 元数据集成体系结构

在明确了元数据管理策略后需要确定实现该管理策略所需的技术体系结构，即元数据集成体系结构。元数据集成体系结构涉及多个概念，例如元模型、元-元模型、公共仓库元模型(CWM)等。

值得注意的是，统一、完整的元数据管理，特别是清晰的主题域划分、完善的元模型和元-元模型有利于更好地管理主数据。

(1) 元模型。模型(model)是对特定的系统、过程、事物或概念的准确而抽象的表示，是描述数据的数据。例如软件架构师可以用概要设计的形式建立一个应用系统的模型。从本质上来说，元数据是数据的形式化模型，是数据的抽象描述，该描述准确地描述了数据。元模型(meta model)也就是模型的模型(或者元-元数据)，是用来描述元数据的模型。使用元模型的目的在于识别资源，评价资源，追踪资源在使用过程中的变化，简单、高效地管理大量网络化数据，实现信息资源的有效发现、查找、一体化组织和对所使用资源的有效管理。图 3-19 显示了数据、元数据和元模型之间的关系。

人们可以将元模型想象成某种形式语言，这样模型就是一篇用该语言描述的文章，其中元模型中的元素就是该语言的词汇，元素之间的关系就是该语言的语法。元模型与形式语言的关系如图 3-20 所示。

图 3-19　数据、元数据和元模型之间的关系

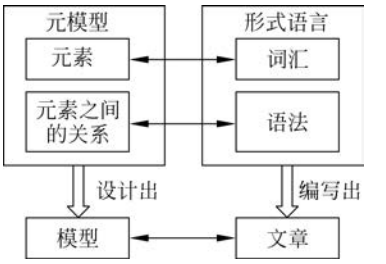


图 3-20　元模型与形式语言的关系

在具体应用中，如果要创建一个关系型表模型，基于该表元模型创建一个实例即可。比如创建一个常见的雇员表(Employees 表)模型，具体如图 3-21 所示，Employees 表中包含了 6 列，分别是编号(ID)、姓(First_name)、名字(Last_name)、部门编号(Depart_ID)、经理编号(Manager_ID)和职位编号(Job_ID)。

同样基于图 3-20 所示的简单关系型表元模型创建另一个实例——Department 表模型。Department 表中包含两列，分别是编号(ID)和部门名称(name)，具体如图 3-22 所示。由于

Department 表模型和 Employees 表模型基于相同的公共元模型，它们是同一个元模型的实例，所以其他工具和应用程序软件可以很容易地理解 Department 表和 Employees 表。

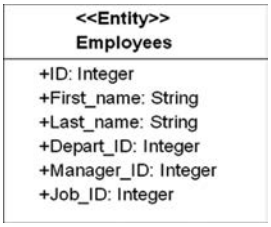


图 3-21　Employees 表模型

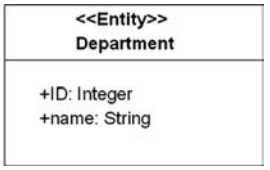


图 3-22　Department 表模型

当元模型在企业中实际应用时，例如在 Hadoop 环境下，通常会涉及大数据集群 NameNode 元数据，包括集群的运行监控信息及文件/目录元数据，表 3-13 为 NameNode 节点的元数据信息，表 3-14 为作业监控信息，表 3-15 为 DataNode 节点的元数据信息。

表 3-13　NameNode 节点的元数据信息

英　文　名	中　文　名	类　　型	备　　注
Configured Capacity	配置容量	double	
Present Capacity	当前总容量	double	
DFS Remaining	剩余容量	double	
DFS Used	已用容量	double	
DFS Used%	使用率	double	小数点后 4 位
Under replicated blocks	待复制数据块	double	
Blocks with corrupt replicas	中断复制数据块	double	
Missing blocks	丢失数据块	double	
Datanodes available	可用节点数	double	
Datanodes Non available	不可用节点数	double	

表 3-14　作业监控信息

英　文　名	中　文　名	类　　型	备　　注
Name	名称	text	
Description	描述	text	
LastModified	更新时间	date	
Steps	步骤	double	
Status	状态	text	
Owner	拥有人	text	

表 3-15　DataNode 节点的元数据信息

英　文　名	中　文　名	类　　型	备　　注
Name	节点名称	text	
Hostname	主机名	text	
Rack	所属机架	text	
Decommission Status	可用状态	text	
Configured Capacity	配置容量	double	
DFS Used	已用容量	double	
Non DFS Used	非 DFS 使用容量	double	
DFS Remaining	剩余容量	double	

续表

英 文 名	中 文 名	类 型	备 注
DFS Used%	使用率	double	小数点后 4 位
DFS Remaining%	剩余率	double	小数点后 4 位
Configured Cache Capacity	配置缓存容量	double	
Cache Used	缓存使用量	double	
Cache Remaining	缓存剩余量	double	
Cache Used%	缓存使用率	double	小数点后 4 位
Cache Remaining%	缓存剩余率	double	小数点后 4 位
Last contact	最近检查时间	date	

(2) 元-元模型。元-元模型就是元模型的模型,有时也被称为本体,是模型驱动的元数据集成体系结构的基础,其定义了描述元模型的语言,规定元模型必须依照一定的形式化规则来建立,以便所有的软件工具都能够对其进行理解。

元-元模型比元模型具有更高的抽象级别,一个元模型是一个元-元模型的实例,元模型比元-元模型更加精细,而元-元模型比元模型更加抽象。元数据(模型)是一个元模型的实例,遵守元模型的规定和约束。用户对象(或用户数据)是元数据(或者称为模型)的实例。元数据的层次结构如表 3-16 所示,共分为 4 层,最高层 L3 是元-元模型,之下是 L2(元模型)和 L1(模型/元数据),最底层是 L0(用户对象/用户数据)。

表 3-16 元数据的层次结构

元层次	名 称	示 例
L3	元-元模型	元类、元属性、元操作
L2	元模型	类、属性、操作、构件
L1	模型/元数据	实体-关系(E-R)图
L0	用户对象/用户数据	交易数据、ODS 数据、数据仓库数据、数据集市数据、数据中心数据等

(3) 公共仓库元模型(CWM)。公共仓库元模型是被对象管理组织(Object Management Group,OMG)采纳的数据仓库和业务分析领域元数据交换开放式行业标准,在数据仓库和业务分析领域为元数据定义公共的元模型和基于 XML 的元数据交换(XMI)。CWM 作为一个标准的接口,可以使处于分布式、异构环境下的数据仓库元数据和商业智能元数据能方便地在不同的数据仓库工具、数据仓库平台和元数据仓库之间进行交换。CWM 提供一个框架为数据源、数据目标、转换、分析、流程和操作等创建和管理元数据,并提供元数据使用的世系信息。因此,CWM 实际上就是一个元数据交换的标准,是为各种数据仓库产品提出的一个标准。CWM 主要包含以下三方面的规范:

- CWM 元模型。CWM 元模型是描述数据仓库系统的模型。为了降低复杂度并达到重用,CWM 元模型采用分层的方式组织它所包含的包。CWM 元模型主要包括 4 层,即基础包 Foundation、资源包 Resource、分析包 Analysis 和管理包 Management。
- CWM XML 和 CWM DTD。DTD 和 XML 是对应于 CWM 中所有包的 DTD 和 XML,它们都遵循 XMI 规范。定义 CWM DTD 和 CWM XML 的主要目的是基于 XML 进行元数据交换,因为 XML 在各个领域的应用越来越广泛,CWM 提供元模型到 XML 的转换,无疑大大增加了自己的通用性,各种分析工具和元数据库可以利用这些模板为自己的元模型生成 DTD 和 XML 文档,这样就可以和其他的工具进行元数据交换。

- CWM IDL。CWM IDL 是共享元数据的应用程序访问接口(API)。CWM IDL 为上面所有的包定义了符合 MOF 1.3 的 IDL 接口,这样就可以利用 CORBA 进行元数据交换。用户可以创建一些具有分析功能的软件包,例如数据挖掘组件等。提供 CWM 中规定的 IDL 接口,就可以被其他支持 CWM 的工具和数据仓库调用,这样大大增强了 CWM 的灵活性和适用性。

CWM 1.1 是在 2003 年 3 月发布的,与之相关的 OMG 组织规范还有 MOF(元对象设施)、UML 和 XMI。这 3 个标准是 OMG 元数据库体系结构的核心,MOF 为构建模型和元模型提供了可扩展的框架,并提供了存取元数据的程序接口;UML 定义了表示模型和元模型的语法和语义;而利用 XMI 可以将元数据转换为标准的 XML 数据流或文件的格式,以便进行交换,这大大增强了 CWM 的通用性。

图 3-23 显示了 OMG 的元数据仓库体系结构,其中 UML 表示对 CWM 模型进行建模,而 MOF 则是 OMG 元模型和元数据的存储标准,它提供在异构环境下对元数据知识库的访问接口。

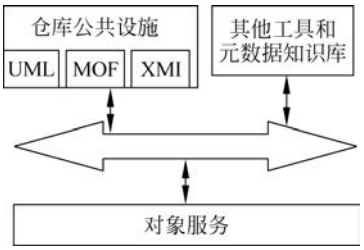


图 3-23　OMG 的元数据仓库体系结构

2. 元数据管理功能

元数据管理功能主要包含数据地图、元数据分析、辅助应用优化、辅助安全管理及基于元数据的开发管理。

1) 数据地图

数据地图是一种图形化的数据资产管理工具,数据地图以拓扑图的形式对数据系统中的各类数据实体、数据处理过程元数据进行分层次的图形化展现,并通过不同层次的图形展现粒度控制,满足开发、运维或者业务上不同应用场景的图形查询和辅助分析需要。数据地图提供的数据服务主要有以下几点:

(1) 快速进行搜索定位,找到企业的各种数据资产,形成有效的数据交汇。

(2) 提供各种数据资产快速展现的个性化形式,方便使用者获取所需要的关键信息。

(3) 在数据搜寻结果之上直接配备方便的分析工具。

(4) 建立数据资产分布及综合评估的入口,以便更好地了解数据资产的各方面信息。

数据地图包含数据的基本信息和统计信息两部分。其中,基本信息主要包含字段信息、存储信息和描述信息;统计信息主要包含数据表的大小、数据表的每天访问次数、数据表的更新时间等各种信息。

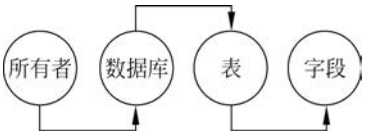


图 3-24　数据血缘关系的层次

2) 元数据分析

(1) 血缘分析。血缘分析(也称血统分析)是指从某一实体出发,往回追溯其处理过程,直到数据系统的数据源接口。图 3-24 描述了数据血缘关系的层次。

图 3-24 描述的是存储在数据库中的结构化数据血缘关系的层次结构,这是最典型的一种血缘关系的层次结构。一般来说,数据所有者是指数据归属于某个组织或者某个人;数据可以在不同的所有者之间流转、融合,形成所有者之间通过数据联系起来的一种关系,这种关系能够清楚地表明数据的提供者和需求者。值得注意的是,在血缘关系中,不同层级数据的血缘关系体现着不同的含义。所有者层次体现了数据的提供方和需求方,其他的层次则体现了数据的来龙去脉。通过不同层级的血缘关系,可以很清楚地了

解数据的迁徙流转，为数据价值的评估、数据的管理提供依据。不过对于不同类型的数据，血缘关系的层次结构会有细微的差别。

对于不同类型的实体，在血缘关系中涉及的转换过程可能有不同类型。例如，对于底层仓库实体，涉及的是 ETL 处理过程；对于仓库汇总表，可能既涉及 ETL 处理过程，又涉及仓库汇总处理过程；而对于指标，除了上面的处理过程，还涉及指标生成的处理过程。血缘分析正是提供了这样一种功能，可以让使用者根据需要了解不同的处理过程，了解每个处理过程具体做什么，需要什么样的输入，又会产生什么样的输出。

对数据进行血缘分析对于用户来说具有重要的价值，当在数据分析中发现问题数据时，可以依赖血缘关系追根溯源，快速地定位到问题数据的来源和加工流程，减少分析的时间和难度。例如，某业务人员发现“客户资产表”中的数据存在质量问题，于是向 IT 部门提出异议，技术人员通过元数据血缘分析发现“客户资产表”受到上游基础数据层中多张不同的数据表影响，从而快速定位问题的源头，低成本地解决问题。图 3-25 显示了血缘关系图。

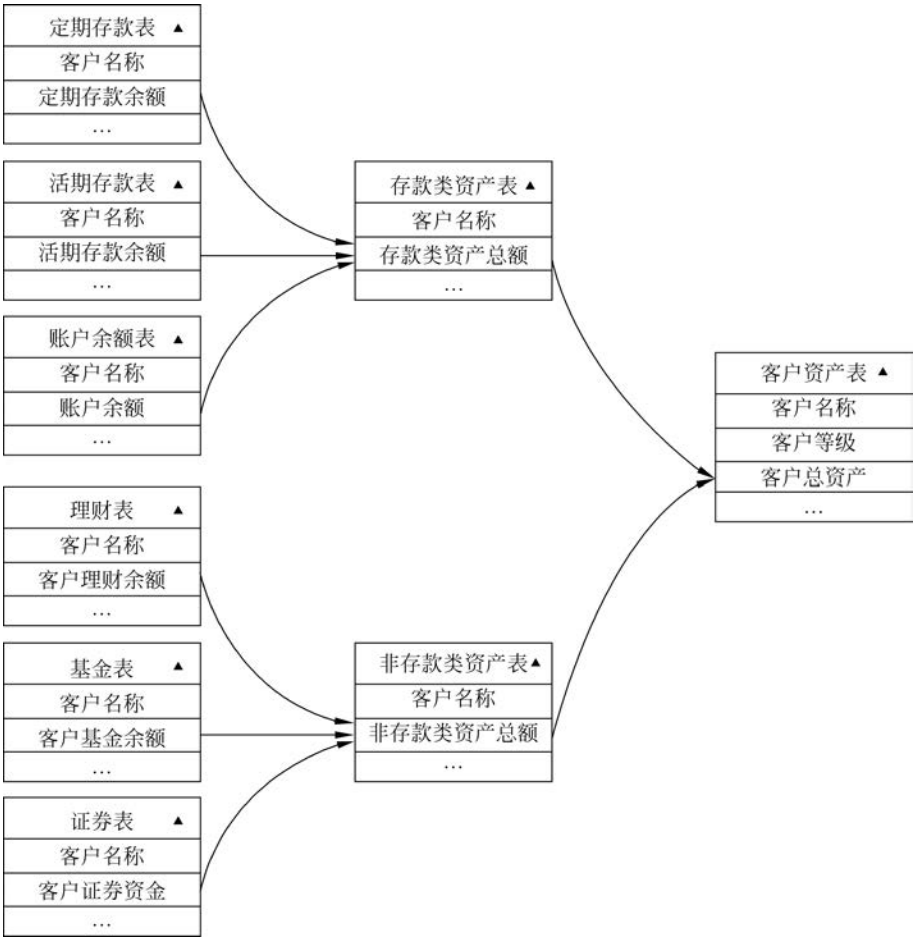


图 3-25　血缘关系图

为实现血缘分析，对于任何指定的实体，首先获得该实体的所有前驱实体，然后对这些前驱实体递归地获得各自的前驱实体，结束条件是所有实体到达数据源接口或者实体没有相应的前驱实体。

血缘分析实例见图 3-26。某数据开发工程师为了满足一次业务需求生成了该表，但是出于程序逻辑清晰或者性能优化的考虑，其中使用了很多份数据表。在这里 Table X 是最终给

业务部门的表，Table A～Table E 是原始数据表，Table F～Table I 是计算出来的中间表，Table J 是其他人处理过的结果表。过了一段时间，业务部门感觉数据开发工程师提供的数据中有个字段异常，怀疑是数据出现了问题，因此需要追踪一下这个字段的来源。首先从 Table X 中找到了异常的字段，然后定位到它来源于 Table I，再从 Table I 定位到它来源于 Table G，接着从 Table G 追溯到了 Table D，最终发现某几天的来源数据有异常来自于数据表 Table D。

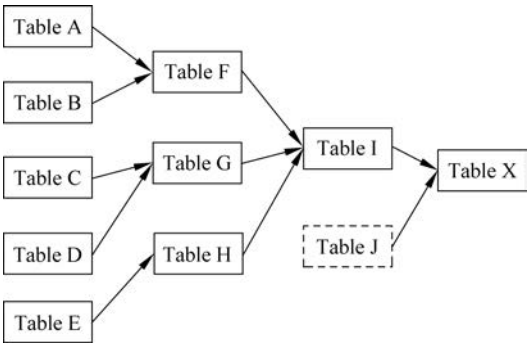


图 3-26　血缘分析实例

这就是血缘分析，它能够追根溯源，并最终找到问题数据的来源。

在 JSON 中包含的数据血缘分析的部分代码如下：

```
"relations": [
    {
        "id": "3",
        "type": "fdd",
        "effectType": "create_view",
        "target": {
            "id": "11",
            "column": "mySal",
            "parentId": "9",
            "parentName": "v_sal",
        },
        "sources": [
            {
                "id": "3",
                "column": "sal",
                "parentId": "2",
                "parentName": "emp",
            },
            {
                "id": "4",
                "column": "commission",
                "parentId": "2",
                "parentName": "emp",
            }
        ],
        "processId": "10"
    }
]
```

(2) 影响分析。影响分析是指从某一实体出发，寻找依赖该实体的处理过程实体或其他实体。如果有需要可以采用递归方式寻找所有的依赖过程实体或其他实体。该功能支持当某些实体发生变化或者需要修改时评估实体影响范围。

(3) 实体关联分析。实体关联分析是从某一实体关联的其他实体和其参与的处理过程两个角度来查看具体数据的使用情况,形成一张实体和所参与处理过程的网络,从而进一步了解该实体的重要程度。本功能可以用来支撑需求变更影响评估的应用。

(4) 实体差异分析。实体差异分析是对元数据的不同实体进行检查,用图形和表格的形式展现它们之间的差异,包括名字、属性及数据血缘和对系统其他部分影响的差异等,在数据系统中存在许多类似的实体。这些实体(例如数据表)可能只有名字或者是属性存在微小的差异,甚至有部分属性、名字都相同,但处于不同的应用中。由于各种原因,这些微小的差异直接影响了数据统计结果,数据系统需要清楚地了解这些差异。本功能有助于进一步统一统计口径,评估近似实体的差异。

(5) 指标一致性分析。指标一致性分析是指用图形化的方式来分析比较两个指标的数据流图是否一致,从而了解指标的计算过程是否一致。该功能是指标血缘分析的一种具体应用。指标一致性分析可以帮助用户清楚地了解将要比较的两个指标在经营分析数据流图中各阶段所涉及的数据对象和转换关系是否一致,帮助用户更好地了解指标的来龙去脉,清楚地理解分布在不同部门且名称相同的指标之间的差异,从而提高用户对指标值的信任。

3) 辅助应用优化

元数据对数据系统的数据、数据加工过程以及数据间的关系提供了准确的描述,利用血缘分析、影响分析和实体关联分析等元数据分析功能可以识别与系统应用相关的技术资源,结合应用生命周期管理过程,辅助进行数据系统应用的优化。

4) 辅助安全管理

企业数据平台所存储的数据和提供的各类分析应用涉及公司经营方面的各类敏感信息,因此在数据系统建设过程中必须采用全面的安全管理机制和措施来保障系统的数据安全。数据系统安全管理模块负责数据系统的数据敏感度、客户隐私信息和各环节审计日志记录管理,对数据系统的数据访问和功能使用进行有效监控。为实现数据系统对敏感数据和客户隐私信息的访问控制,进一步实现权限细化,安全管理模块应以元数据为依据,由元数据管理模块提供敏感数据定义和客户隐私信息定义,辅助安全管理模块完成相关安全管控操作。

5) 基于元数据的开发管理

数据系统项目开发的主要环节包括需求分析、设计、开发、测试和上线。开发管理应用可以提供相应的功能,对以上各环节的工作流程、相关资源、规则约束、输入/输出信息等提供管理和支持。

3. 元数据管理的实施

在明确了元数据管理策略和元数据集成体系结构之后,企业可以根据需要选择合适的业务元数据和技术元数据管理工具,并制定相应的元数据管理制度进行全面的元数据管理。

大数据扩大了数据的容量、提高了速度、增加了多样性,给元数据管理带来了新的挑战。在构建关系型数据仓库、动态数据仓库和关系型数据中心时进行元数据管理,有助于保证数据被正确地使用、重用并满足各种规定。通常,大数据分析是受用例驱动的,企业可以通过梳理大数据用例的方式逐步完善大数据的元数据管理。针对大数据的业务元数据,依旧可以通过构建基础本体、领域本体、任务本体和应用本体等方式来实现。通过构建基础本体,实现对高级别且通用的概念以及概念之间关系的描述;通过构建领域本体,实现对领域的定义,并确定该领域内共同认可的词汇、词汇业务含义和对应的信息资产等,提供对该领域知识的共同理解;通过构建任务本体,实现任务元素及其之间关系的规范说明或详细说明;通过构建应用本体,实现对特定应用的概念描述,其是依赖于特定领域和任务的。这样就通过构建各种本

体，在整个企业范围内提供一个完整的共享词汇表，保证每个元数据元素在信息供应链中的每个组件中语义上保持一致，实现语义等效。

简单来说，企业可以尝试以下步骤进行大数据的元数据管理：

（1）考虑到企业可以获取数据的容量和多样性，应该创建一个体现关键大数据业务术语的业务定义词库（本体），该业务定义词库不仅包含结构化数据，还可以将半结构化和非结构化数据纳入其中。

（2）及时跟进和理解各种大数据技术中的元数据，提供对其连续、及时的支持，比如 MPP 数据库、流计算引擎、Apache Hadoop/企业级 Hadoop、NoSQL 数据库以及各种数据治理工具（如审计/安全工具、信息生命周期管理工具等）。

（3）对业务术语中的敏感大数据进行标记和分类，并执行相应的大数据隐私政策。

（4）将业务元数据和技术元数据进行链接，可以通过操作元数据（例如流计算或 ETL 工具所生成的数据）监测大数据的流动；可以通过数据世系分析（血缘分析）在整个信息供应链中实现数据的正向追溯或逆向追溯，了解数据经历了哪些变化，查看字段在信息供应链中各组件间的转换是否正确等；可以通过影响分析了解某个字段的变更会对信息供应链中其他组件的字段造成哪些影响等。

（5）扩展企业现有的元数据管理角色，以适应大数据治理的需要，例如可以扩充数据治理管理者、元数据管理者、数据主管、数据架构师以及数据科学家的职责，加入大数据治理的相关内容。

元数据管理的实施通常用元数据管理模块来实现，如图 3-27 所示。

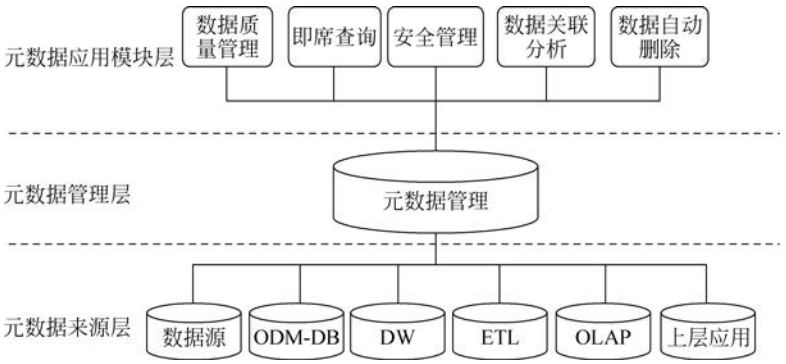


图 3-27　元数据管理模块

模块的常用功能如下：

（1）元数据管理从数据源、ODM-DB（数据挖掘）、DW（数据仓库）、ETL（数据仓库工具）、OLAP（联机分析处理）、上层应用等模块中获取元数据信息。

（2）元数据管理系统作为数据质量管理系统的依据，指导数据质量管理系统评价数据质量，主要体现在数据的完整性、准确性和关联一致性等方面。

（3）元数据管理系统提供指标库数据供页面呈现。

（4）元数据管理为综合分析系统的即席查询功能提供了基础。即席查询功能利用元数据中存储的业务元数据和技术元数据生成后台数据查询所需的 SQL 语句，得到最终的查询结果。

（5）元数据管理系统通过 API 接口调用向外部暴露数据。

（6）安全模块获取元数据的指标敏感度描述，为安全管理模块提供数据支持。

（7）元数据为 DW 数据的有效期管理提供指导，为实现数据自动删除提供数据支持。

此外，企业还可以考虑使用元数据平台来进行元数据管理，如图 3-28 所示。

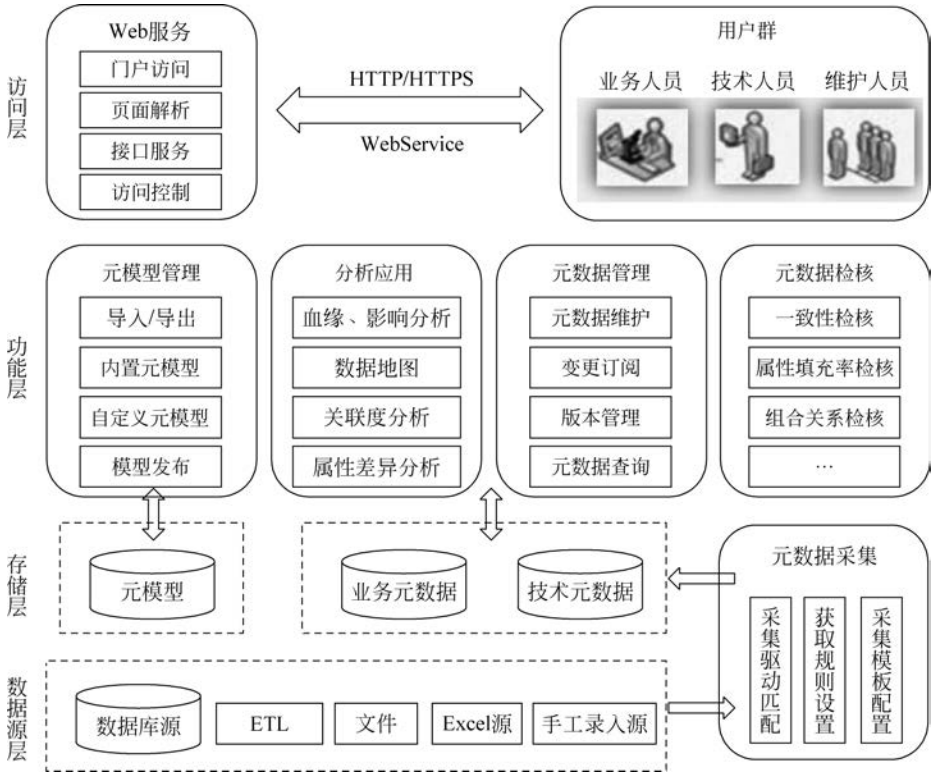


图 3-28　元数据平台

4. 元数据治理工具 Apache Atlas

目前企业中常用的元数据治理工具是 Apache Atlas，下面将对该工具做简单的介绍。

1）Apache Atlas 简介

Atlas 最早由 Hortonworks 公司开发，用来管理 Hadoop 项目里面的元数据，进而设计为数据治理的框架。后来其开源出来给 Apache 社区进行孵化，得到 Aetna、Merck、Target、SAS、IBM 等公司的支持并发展演进。因其支持横向海量扩展，并具有良好的集成能力和开源的特点，国内大部分厂家选择使用 Atlas 或对其进行二次开发。

ApacheAtlas 是 Hadoop 社区为解决 Hadoop 生态系统的元数据治理问题而产生的开源项目，它为 Hadoop 集群提供了包括数据分类、集中策略引擎、数据血缘、安全和生命周期管理在内的元数据治理核心能力，支持对 Hive、Storm、Kafka、HBase、Sqoop 等进行元数据管理以及以图库的形式展示数据的血缘关系。

2）Apache Atlas 的原理

在内部，Atlas 通过使用图形模型管理元数据对象，以实现元数据对象之间的灵活性和丰富的关系。图形引擎是负责在类型系统的类型和实体之间进行转换的组件，以及基础图形模型。除了管理图形对象之外，图形引擎还为元数据对象创建适当的索引，以便有效地搜索它们。在存储方面，目前 Atlas 使用 Titan 图数据库来存储元数据对象（使用 Titan 来存储它管理的元数据）。Titan 使用两种存储，默认情况下元数据存储配置为 HBase，索引存储配置为 Solr。另外，也可以通过构建相应的配置文件使用 BerkeleyDB 进行元数据存储和 Index 使用 ElasticSearch 存储 Index。Atlas 还定义了一套 Apache Atlas Api（Apache Atlas Api 主要是

对 Type、Entity、Attribute 这 3 个构件增/删/改/查操作），允许采用不同的图数据库引擎来实现 API，便于切换底层存储，所以 Atlas 读/写数据的过程可以看作将图数据库对象映射成 Java 类的过程。

（1）Type（类型）。Atlas 中的“类型”是一个定义，说明如何存储并访问特定类型的元数据对象。类型表示一个特征或一个特性集合，这些属性定义了元数据对象。

（2）Entity（实体）。Atlas 中的一个“实体”是类型“Type”的特定值或实例，因此表示特定的现实世界中的元数据对象。

（3）Attribute（属性）。Atlas 中的“属性”定义了与类型系统相关的概念，例如是否复合、是否索引、是否唯一等。

Apache Atlas 为 Hadoop 的元数据治理提供了以下特性：

（1）数据分类。Apache Atlas 为元数据导入或定义业务导向的分类注释，并自动捕获数据集和底层元素之间的关系。

（2）集中审计。Apache Atlas 能够捕获所有应用、过程以及与数据交互的安全访问信息。

（3）搜索与血缘。Apache Atlas 对数据集血缘关系的可视化浏览使用户可以下钻到操作、安全以及与数据起源相关的信息。

图 3-29 显示了用 Apache Atlas 展示数据血缘关系。

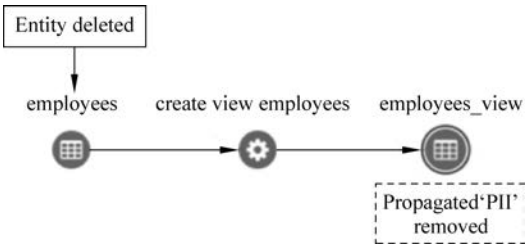


图 3-29　用 Apache Atlas 展示数据血缘关系

3.4　本章小结

（1）数据质量正是企业应用数据的瓶颈，高质量的数据可以决定数据应用的上限，而低质量的数据必然拉低数据应用的下限。

（2）数据质量管理是指对数据从计划、获取、存储、共享、维护、应用到消亡的生命周期的每个阶段里可能引发的各类数据质量问题进行识别、度量、监控、预警等一系列管理活动，并通过改善和提高组织的管理水平使得数据质量获得进一步提高。

（3）数据是数字经济的核心，对企业而言，数据更是企业重要的资产。数据资产是指个人或企业的照片、文档、图纸、视频、数字版权等以文件为载体的数据，是相对于实物资产以数据形式存在的一类资产。

（4）数据标准就是企业建立的一套符合自身实际，涵盖定义、操作、应用多层次数据的标准化体系。

（5）主数据是用来描述企业核心业务实体的数据，它是具有高业务价值的、可以在企业内跨越各个业务部门被重复使用的数据，并且存在于多个异构的应用系统中。

(6) 主数据通常需要在整个企业范围内保持一致性、完整性、可控性,为了达成这一目标,需要进行主数据管理。

(7) 元数据是描述企业数据的相关数据,一般是指在 IT 系统建设过程中所产生的与数据定义、目标定义、转换规则等相关的关键数据,包括在数据的业务、结构、定义、存储、安全等各方面对数据的描述。

(8) 电子文件元数据是描述电子文件数据属性的数据,包括文件的格式、编排结构、硬件和软件环境、文件处理软件、字处理软件和图形处理软件、字符集等数据。

(9) 为了能够更高效地运转,企业需要明确元数据管理策略和元数据集成体系结构,依托成熟的方法论和工具实现元数据管理,并有步骤地提升其元数据管理成熟度。

3.5 实训

1. 实训目的

通过本章实训了解数据质量与管理的特点,能进行简单的有关操作。

2. 实训内容

1) 目前在企业中进行数据治理时经常使用数据治理管理平台(如图 3-30 所示)。

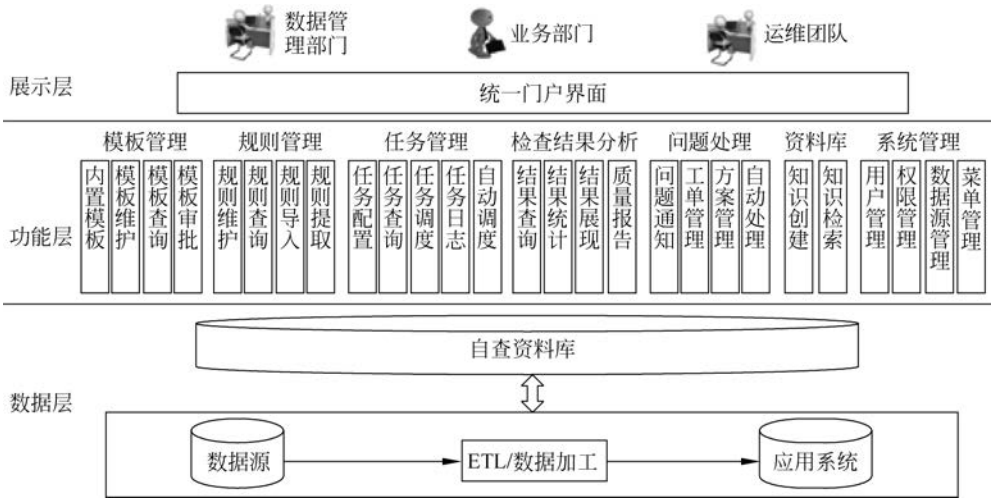


图 3-30 数据治理管理平台

该平台的主要功能如下:

(1) 模板管理。

(2) 规则管理。

(3) 任务管理。

(4) 检查结果分析。

(5) 问题处理。

(6) 资料库。

(7) 系统管理。

请根据本章内容描述各模块应具备哪些基本功能(提示:模板管理是数据质量管理平台数据展现功能、数据录入功能的基础,内置模板可以通过模板创建功能进行扩展)。

2）找出表 3-17～表 3-19 中的数据和元数据

表 3-17　student 表

s_no(学号)	s_name(姓名)	s_sex(性别)	s_department(系部)	s_age(年龄)
0501001	刘洋	男	计算机	21

表 3-18　score 表

s_no(学号)	c_course(课程号)	c_score(分数)
0501001	c1003	89

表 3-19　course 表

c_no(课程号)	c_name(课程名)
c1003	Python 程序设计

3）使用 DataCleaner 进行数据质量监控管理

DataCleaner 是一个简单、易于使用的数据质量应用工具，旨在分析、比较、验证和监控数据。它能够将凌乱的半结构化数据集转换为数据可视化软件可以读取的干净可读的数据集。此外，DataCleaner 还提供了数据仓库和数据管理服务。

（1）在网上下载该软件，本书使用的版本是 5.1.5（本书配有 DataCleaner 软件），然后直接解压运行即可。如果已经安装成功，则在安装目录下直接双击 DataCleaner 图标即可运行，如图 3-31 所示。

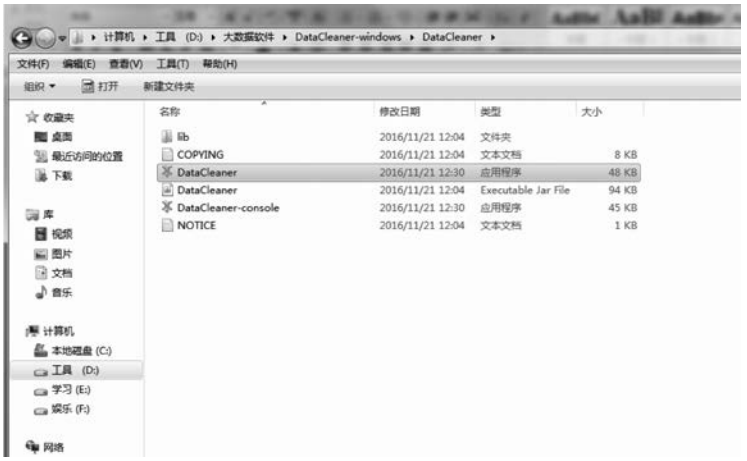


图 3-31　运行 DataCleaner

图 3-32 显示了 DataCleaner 的运行界面。

（2）打开 DataCleaner，在运行界面中选中 Build new job 选项，进入 Select datastore 界面，并选中 Customers 选项，如图 3-33 所示，表示使用 DataCleaner 自带的 customers.csv 数据集，除此以外使用者也可以导入外部数据文件。

customers.csv 数据集的部分内容如图 3-34 所示，该数据集共有 5115 行数据。

（3）在弹出的 Customers| Analysis Job 界面中左边显示的是 customers.csv 数据集的基本数据情况和每个字段的情况，中间工作区显示的是该数据集的名称，如图 3-35 所示。

（4）右击 customers.csv 图标，在弹出的快捷菜单中选择 Quick analysis 命令，该命令用于对所有的数据字段进行分析并查看，如图 3-36 所示。

（5）在弹出的分析数据对话框中可以清楚地看见该数据集中所有字段的情况，如图 3-37 所示。

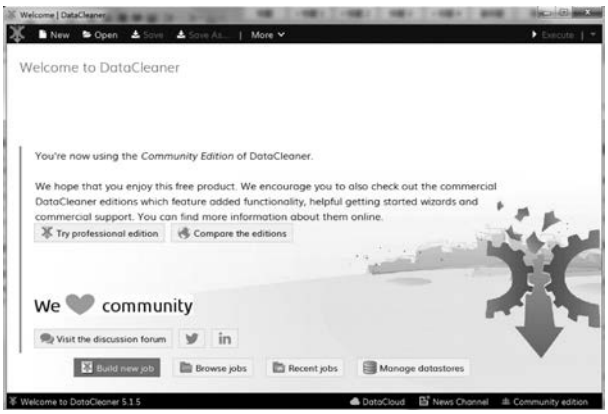


图 3-32　DataCleaner 的运行界面

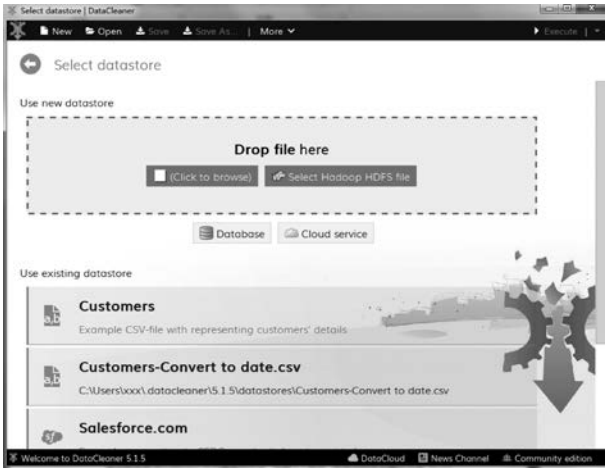
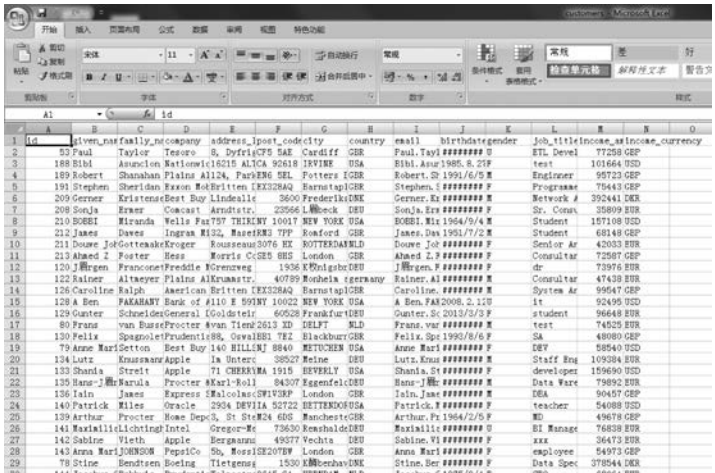


图 3-33　运行 DataCleaner 并选择自带的数据集



id	given_nar	family_nc	company	address_1	post_code	city	country	email	birthdate	gender	job_title	income_as	income_currency
53	Paul	Taylor	Tesoro	8, Dyfrig	CF5 5AE	Cardiff	GBR	Paul.Tayl	########	U	ETL Devel	77258	GBP
188	Bibi	Asuncion	Nationwic	16215	ALICA	92618	IRVINE	USA	Bibi.Asur	1985. 8. 21	F	test	101664	USD
189	Robert	Shanahan	Plains	41124,	Park	ENG 5EL	Potters	GBR	Robert.Sh	1991/6/5	M	Engineer	95723	GBP
191	Stephen	Sheridan	Exxon Mo	Britten	EX32 8AQ	Barnstap	GBR	Stephen.	########	F	Programmer	75443	GBP
209	Gerner	Kristensen	Best Buy	Lindealle	3600	Frederik	DNK	Gerner.Kr	########	M	Network A	392441	DKR
208	SonJa	Erner	Comcast	Arndtstr.	23566	L[illegible]beck	DEU	SonJa.Ern	########	F	Sr. Consu	35809	EUR
210	BOBBI	Miranda	Wells Far	757 THIR	NY 10017	NEW YORK	USA	BOBBI.Mir	1964/9/4	M	Student	157108	USD
212	James	Dawes	Ingram M	132, Hazel	RM3 7PP	Romford	GBR	James.Daw	1951/7/2	M	Student	68148	GBP
211	Douwe Jo	Gottemake	Kroger	Rousseau	3076 EX	ROTTERDA	NLD	Douwe Jo	########	F	Senior Ar	42033	EUR
213	Ahmed Z	Foster	Hess	Morris Cr	SE5 0HS	London	GBR	Ahmed Z.	########	F	Consultar	72587	GBP
120	J[illegible]rgen	Franconet	Freddie M	Grenzweg	1936 K	[illegible]nigsbr	DEU	J[illegible]rgen. F	########	F	dr	73976	EUR
122	Rainer	Altaeyer	Plains Al	Krumastr.	40789	Monheim	germany	Rainer. Al	########	M	Consultar	47438	EUR
126	Caroline	Ralph	American	Britten	EX32 8AQ	Barnstap	GBR	Caroline.	########	M	System Ar	99547	GBP
128	A Ben	FAKAHANY	Bank of	110 E 59	1NY	10022	NEW YORK	USA	A Ben.FAK	2008. 2. 12	U	it	92495	USD
129	Gunter	Schneider	General I	Goldsteir	60528	Frankfurt	DEU	Gunter.Sc	2013/3/3	F	student	96648	EUR
80	Frans	van Russ	Procter &	van Tiem	2613 XD	DELFT	NLD	Frans. van	########	M	test	74525	EUR
130	Felix	Spagnolet	Prudentie	88, Oswal	8EB 7EZ	Blackburn	GBR	Felix. Spc	1993/8/6	F	SA	48080	GBP
79	Anne Mari	Setton	Best Buy	140 HILLS	NJ 8840	METUCHEN	USA	Anne Mari	########	F	DEV	58540	USD
134	Lutz	Krauss	Apple	In Unterc	38527	Meine	DEU	Lutz. Krau	########	M	Staff Eng	109384	EUR
133	Shanda	Streit	Apple	71 CHERR	MA 1915	BEVERLY	USA	Shanda. St	########	F	developer	159690	USD
135	Hans-J[illegible]r	Warula	Procter &	Karl-Roll	84307	Eggenfel	DEU	Hans-J[illegible]r.	########	M	Data Ware	79892	EUR
136	Iain	James	Express S	Malcolms	CW1 3RP	London	GBR	Iain. Jame	########	M	DBA	90457	GBP
140	Patrick	Miles	Oracle	2934 DEV	IA 52722	BETTENDO	USA	Patrick. M	########	F	teacher	54088	USD
139	Arthur	Procter	Home Dep	3, St Ste	M24 6DS	Manchest	GBR	Arthur. Pr	1964/2/5	F	MD	49678	GBP
141	Maximilli	Lichtingh	Intel	Greger-Re	73630	Remshalde	DEU	Maximilli	########	U	BI Manage	76838	EUR
142	Sabine	Vieth	Apple	Bergsanns	49377	Vechta	DEU	Sabine. Vi	########	F	xxx	36473	EUR
143	Anna Mari	JOHNSON	PepsiCo	Sb, Morsi	SE20 7BW	London	GBR	Anna Mari	########	F	employee	54973	GBP
78	Stine	Bendtsen	Boeing	Tietgensg	1530 K	[illegible]benhav	DNK	Stine. Ben	########	F	Data Spec	378544	DKR
144	Jacobus C	Bekhuis	Prudentie	Talmastr	9645 GA	VEENDAM	NLD	Jacobus C	1975/9/4	F	CEO	68064	EUR
55	Namhoha	Bradbury	Johnson &	274b, Br	SE4 2SP	London	GBR	Namhoham.	1993/9/2	M	IT Specia	38239	GBP
31	ASHOK	HARRAP	Caterpill	EWR-203C	MJ 8807	BRIDGEWA	USA	ASHOK. HAR	########	F	Developer	193266	USD
138	Branch	Roy	Nationwic	969 SOUTH	CA 91724	COVINA	USA	Branch. Rc	########	M	SQL Devel	133852	USD
58	Dick	Lightfoot	Freddie M	41, Parka	DA1 4HG	Erith	GBR	Dick.Ligh	########	U	engineer	30528	GBP
217	Manuel	Rupp	Intel	Dorfstr.	93098	Mintraching	DEU	Manuel. Rt	########	M	engineer	85535	EUR
16	Steve	Ebbrell	Apple	7, Merrie	SO532FQ	Eastleigh	GBR	Steve. Ebb	########	M	pm	63807	GBP
59	Erhard	Schollen	Kroger	Heidenhei	73312	Geislingen	DEU	Erhard. Sc	1958/7/3	F	.	72800	EUR
63	Carolyn	Robinson	Comcast	1100 WIN	MA 2451	WALTHAM	USA	Carolyn. R	1958/1/8	M	Senior As	103400	USD
190	Cary H	THOMPSON	UnitedHea	CASPER	CT 6810	DANBURY	USA	Cary H. TH	########	M	Systems A	71936	USD
64	Wilfred	Hosking	Exxon Mo	Riversmea	SW6 3SF	London	Great Bri	Wilfred. H	1953/9/7	M	Digital P	61196	GBP
65	GILBERT	Starzyk	Enterprise	5801 BRI	CO 80022-	COMMERCE	United st	GILBERT. S	########	F	Computer	139645	USD
66	Alton	Caplan	American	12911 SUI	MD 20720	DOWIE	USA	Alton. Cap	1973/7/1	M	Applicati	195380	USD
67	Hugh	Molloy	General E	30, Will	NN1 5SLN	Kettering	GBR	Hugh. Moll	########	F	Business	70370	GBP
68	Klaus-Die	Jenneweir	American	Geschenv	48161 M	[illegible]nster	DEU	Klaus-Die	1966/7/2	M	Database	35177	EUR

图 3-34　customers. csv 数据集的部分内容

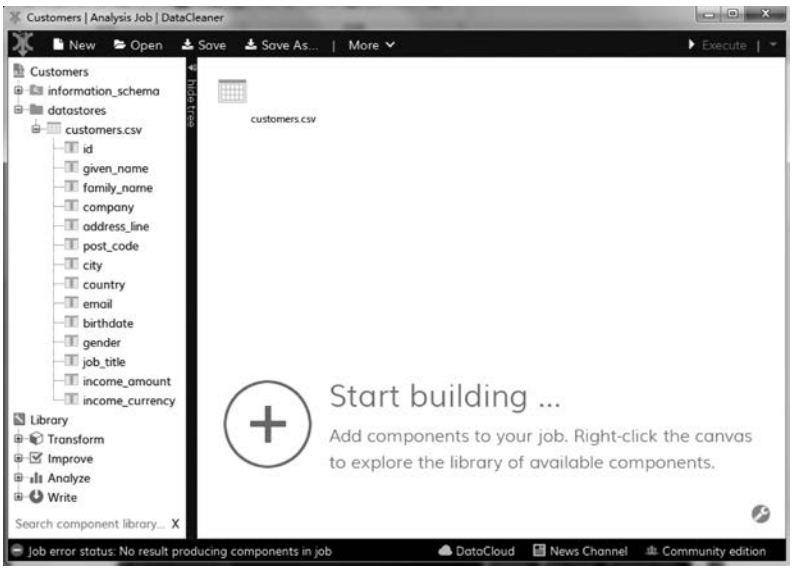


图 3-35　显示数据集情况

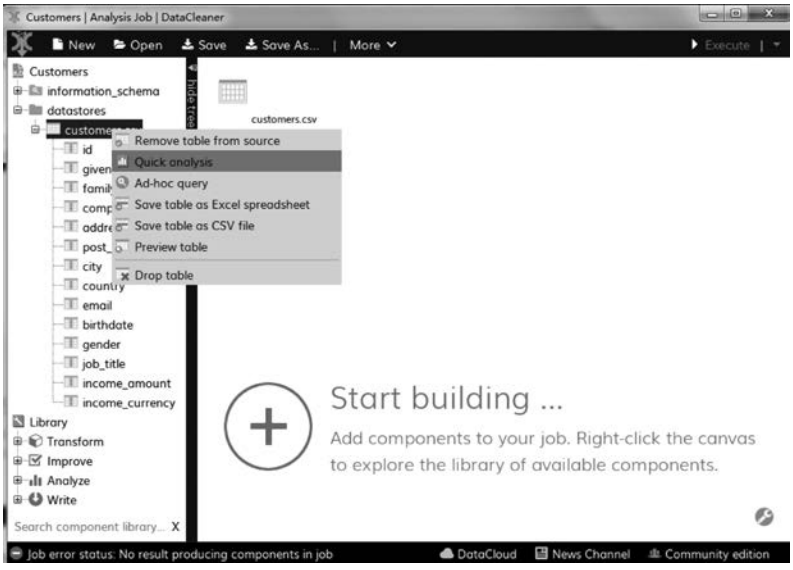


图 3-36　分析数据

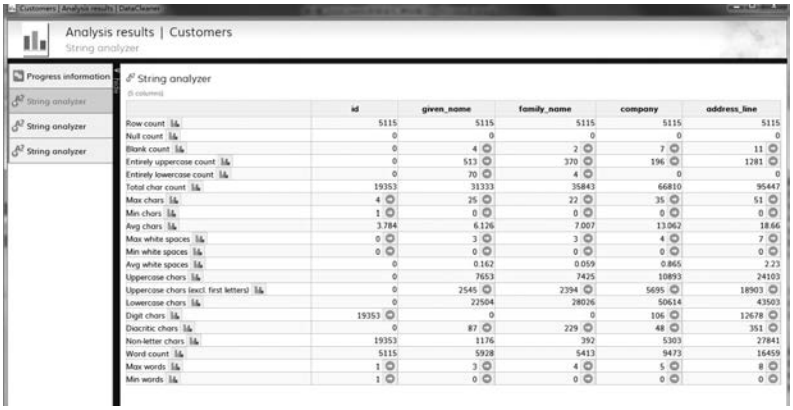


图 3-37　查看数据的分析结果

(6) 如果查看某一个字段的分析情况,也可以执行同样的操作,如图 3-38 所示为选中 id 字段的情况。

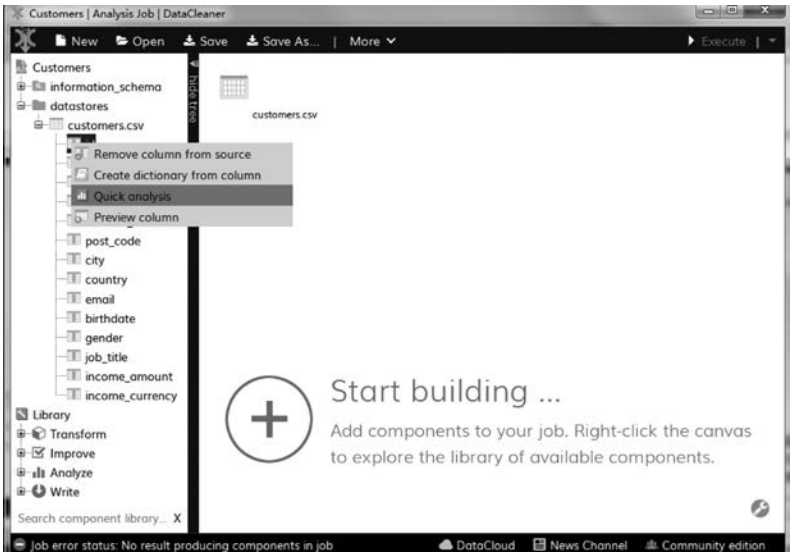


图 3-38 选中 id 字段

(7) 在弹出的对话框中可以查看 id 字段的所有情况,如图 3-39 所示。

图 3-39 查看 id 字段

(8) 返回到 Customers| Analysis Job 界面,在工作区中右击 customers. csv 图标,在弹出的快捷菜单中选择 Preview data 命令,查看该数据集中的所有数据情况,如图 3-40 所示。

(9) 在弹出的对话框中显示的数据如图 3-41 所示。

(10) 在 Customers| Analysis Job 界面中选中 Analyze,然后在展开的列表中选中 Unique key check,以查看字段的数据重复率,如图 3-42 所示。

(11) 在工作区中右击 customers. csv 图标,在弹出的快捷菜单中选择 Link to 命令,并建立 customers. csv 图标和 Unique key check 图标的联系,如图 3-43 和图 3-44 所示。

(12) 双击 Unique key check 图标,在弹出的对话框中选中 id 选项,以查看 id 字段中数据的重复率,如图 3-45 所示。

(13) 返回到 Customers| Analysis Job 界面,选中右上角的 Execute,执行本次操作,并查看运行结果,如图 3-46 所示。

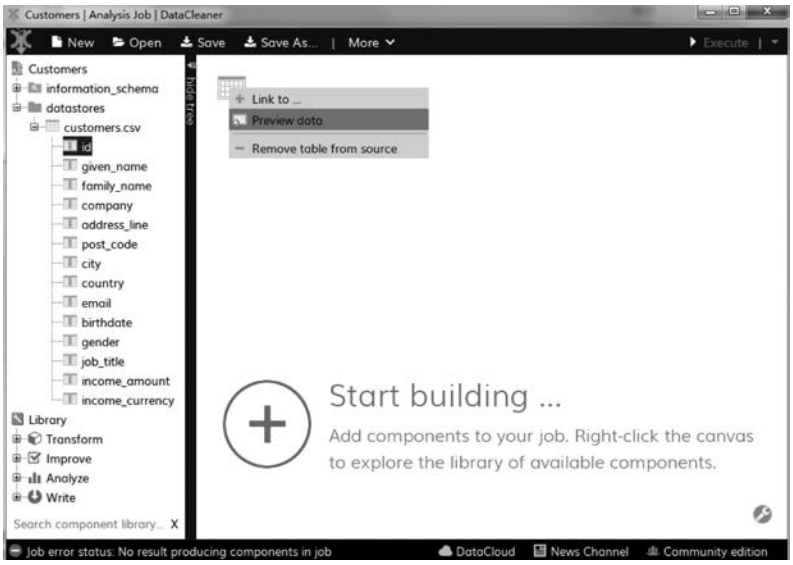


图 3-40　查看 customers. csv 的数据

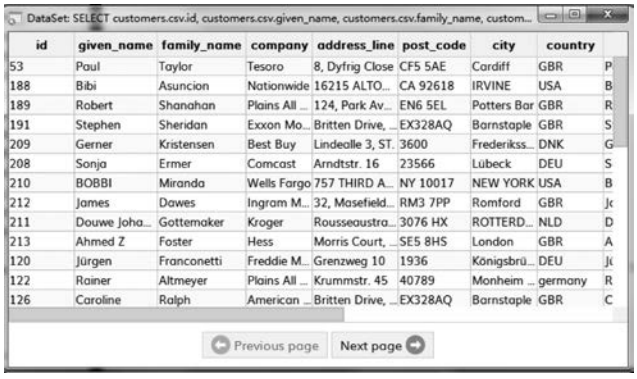


图 3-41　customers. csv 的数据

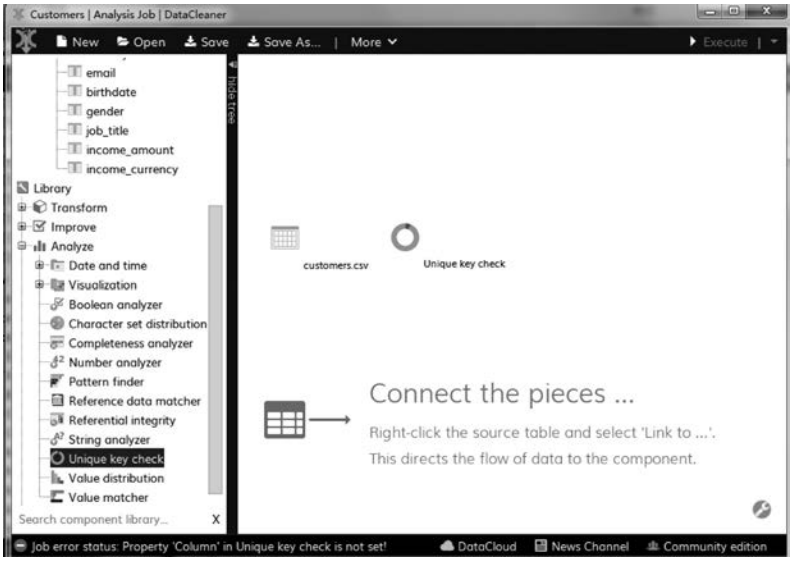


图 3-42　选中 Unique key check

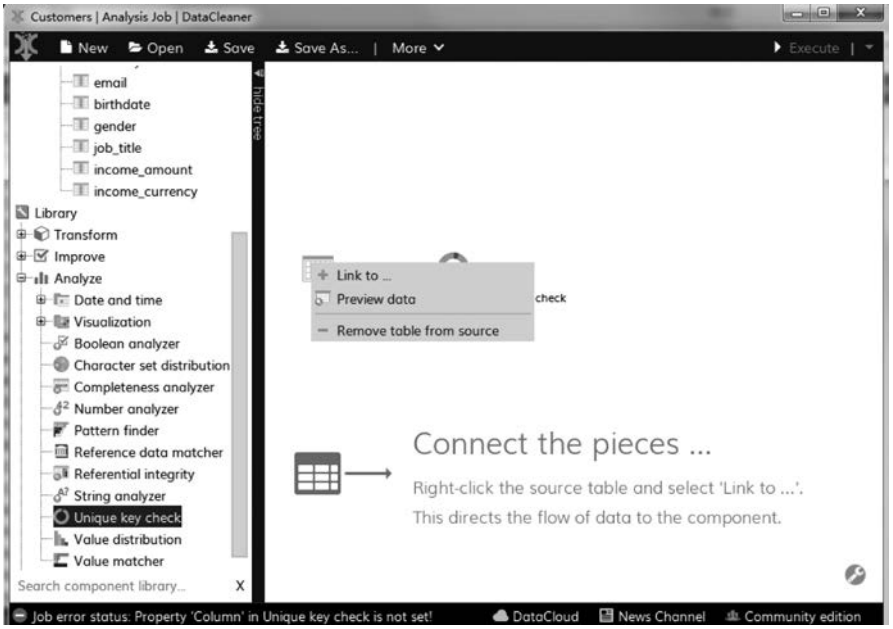


图 3-43　选择 Link to 命令

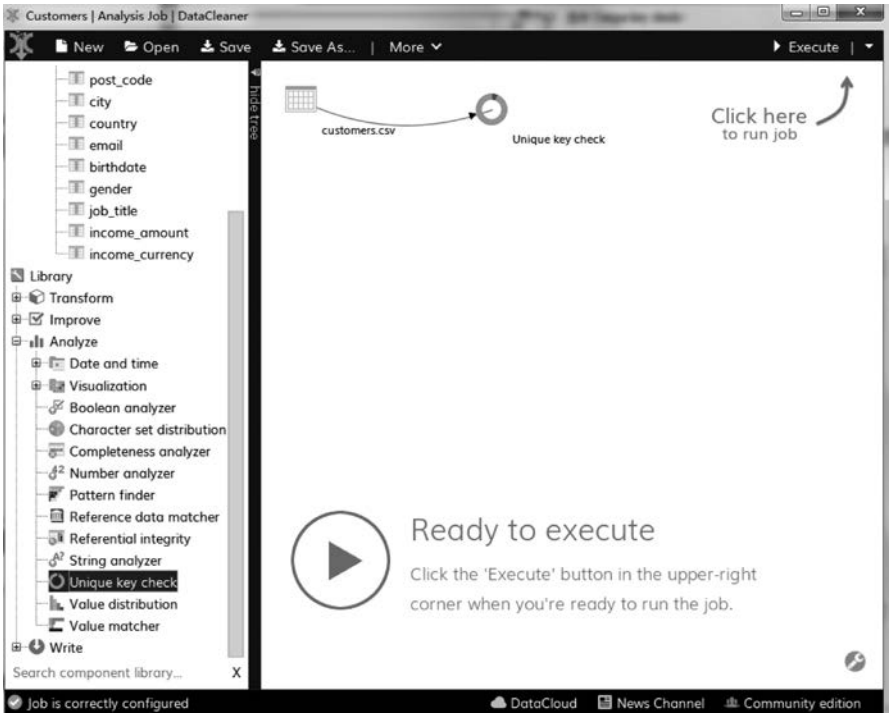


图 3-44　建立图标的联系

图 3-46 显示了 customers.csv 数据集中 id 字段的数据重复率，从图中可以看出该字段存在 15 个重复数据。

4）使用 Python 绘制桑基图描述数据管理过程

在实施数据治理时，人们需要使用编程工具来进行代码的编写。在本书中需要读者掌握的编程工具主要有 Python、MySQL（或其他数据库）、Kettle、Linux 以及 Hadoop 框架等。

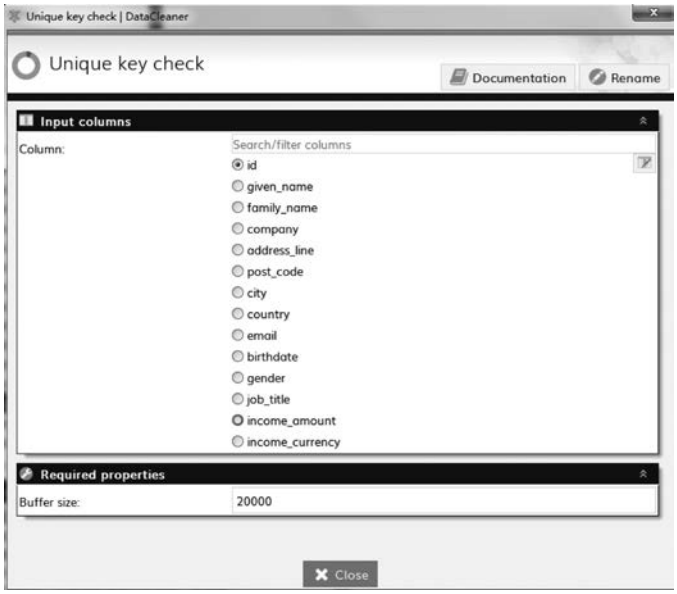


图 3-45　选中 id 选项

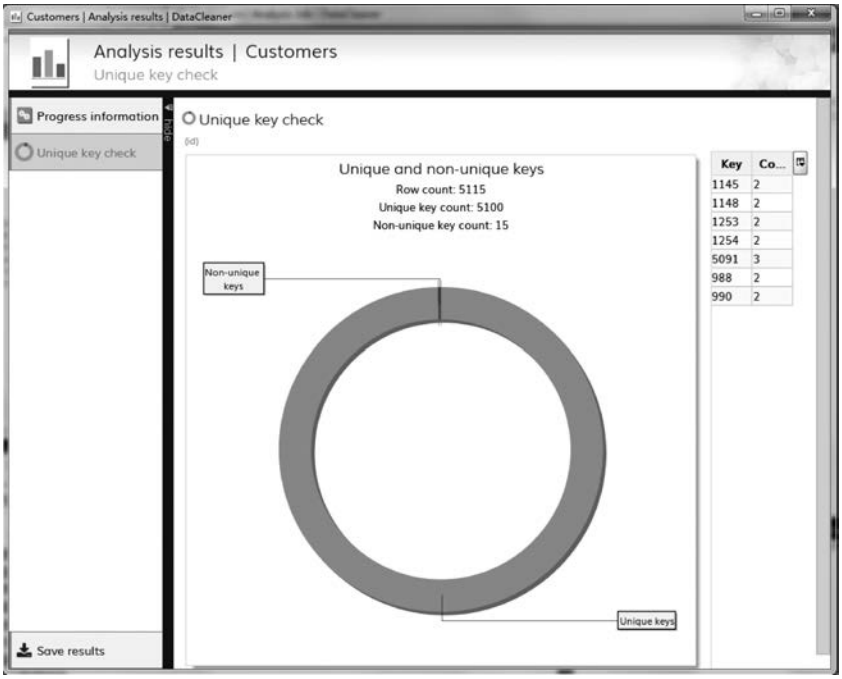


图 3-46　查看运行结果

桑基图也叫桑基能量分流图或者桑基能量平衡图。它是一种特定类型的流程图，主要由边、流量和节点组成，其中边代表了流动的数据，流量代表了流动数据的具体数值，节点则代表了不同分类。桑基图中延伸的分支的宽度对应数据流量的大小，所有主支宽度的总和应与所有分出去的分支宽度的总和相等，保持能量的平衡，因此其非常适用于用户流量等数据的可视化分析。

绘制项目管理的桑基图，Python 代码如下：

```
from pyecharts import options as opts
```

```
from pyecharts.charts import Sankey
nodes = [
    {"name": "项目 1"},
    {"name": "项目 2"},
    {"name": "项目 3"},
    {"name": "项目 4"},
    {"name": "项目 5"},
    {"name": "项目 6"},
]
links = [
    {"source": "项目 1", "target": "项目 2", "value": 10},
    {"source": "项目 2", "target": "项目 3", "value": 15},
    {"source": "项目 3", "target": "项目 4", "value": 20},
    {"source": "项目 5", "target": "项目 6", "value": 25},
]
c = (
    Sankey()
    .add(
        "桑基图",
        nodes,
        links,
        linestyle_opt = opts.LineStyleOpts(opacity = 0.2, curve = 0.5, color = "source"),
        label_opts = opts.LabelOpts(position = "right"),
    )
    .set_global_opts(title_opts = opts.TitleOpts(title = "标题"))
    .render("桑基图.html")
)
```

该实训使用 Pyecharts 库来实现，Pyecharts 是一个用于生成 Echarts 图表的类库。Echarts 是百度开源的一个数据可视化 JS 库。在本段代码中 Sankey 表示桑基图。

运行程序生成的是一个 HTML 页面，如图 3-47 所示。

尝试绘制桑基图来进行数据血缘分析，可使用代码或 Tableau 实现，如图 3-48 所示。

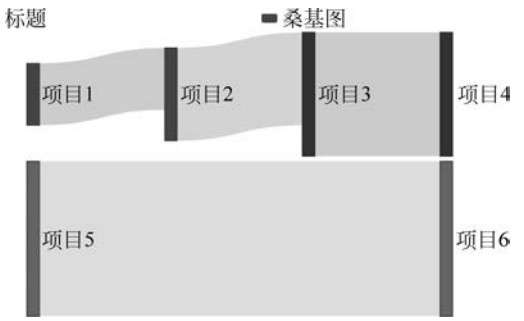


图 3-47 运行结果

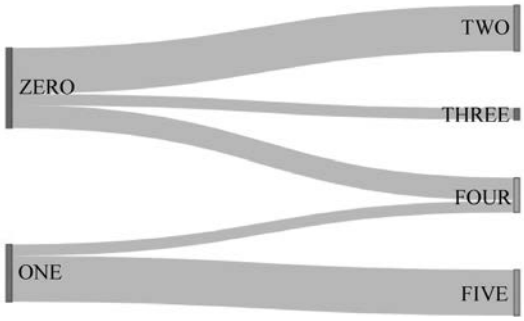


图 3-48 绘制桑基图进行数据血缘分析

5）使用 Tableau 绘制甘特图查看项目的活动情况

（1）运行 Tableau，在已保存的数据源中选择“示例-超市”。

（2）将“维度”中的“发货日期”拖到列中，并将“维度”中的“客户名称”和“细分”拖到行中，如图 3-49 所示。

图 3-49 设置行和列

（3）右击“发货日期”选项，在弹出的快捷菜单中选择“月”，如图 3-50 所示；并右击“细分”选项，在“筛选器”中选择“小型企业”，如图 3-51 所示。

（4）在“标记”选项中选择图形为“甘特条形图”，如图 3-52 所示。

图 3-50　设置月份

图 3-51　设置筛选器内容

图 3-52　选择图形为甘特条形图

（5）查看最终显示结果，如图 3-53 所示。

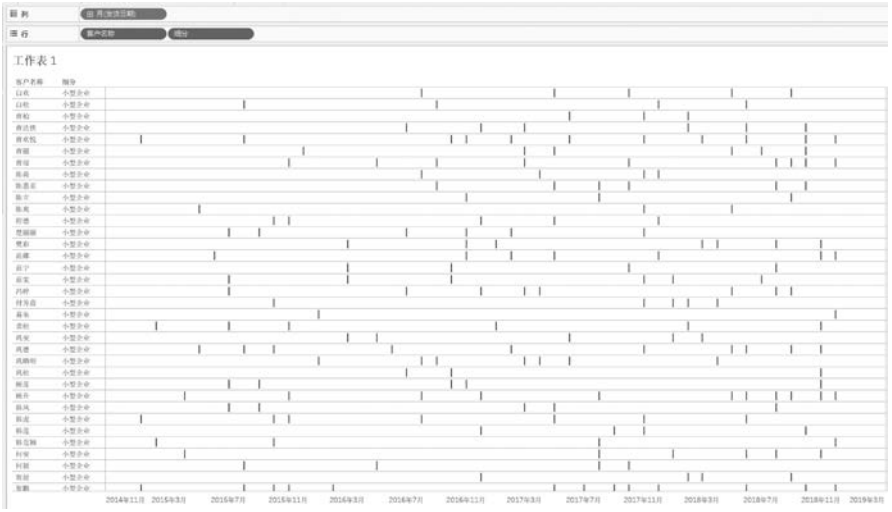


图 3-53　甘特条形图

6）使用 Kettle 查看数据质量。

（1）从官网上下载 JDK。

（2）配置 path 变量。下载 JDK 之后进行安装，安装完毕后要进行环境配置。首先右击

“我的电脑”，选择“属性”，然后在弹出的“系统属性”对话框中选择“高级”选项卡，单击“环境变量”按钮，在弹出的对话框中找到 path 变量，并把 Java 的 bin 路径添加进去，用分号隔开，注意要找到自己安装的对应路径，例如“D:\Program Files\Java\jdk1.8.0_181\bin”。

（3）配置 classpath 变量。在环境变量中新建一个 classpath 变量，里面的内容要填 Java 文件夹中 lib 下 dt.jar 和 tools.jar 的路径，例如“D:\Program Files\Java\jdk1.8.0_181\lib\dt.jar”和“D:\Program Files\Java\jdk1.8.0_181\lib\tools.jar”。

（4）在配置完后运行 cmd 命令，输入命令 java，如果配置成功会出现如图 3-54 所示的界面。

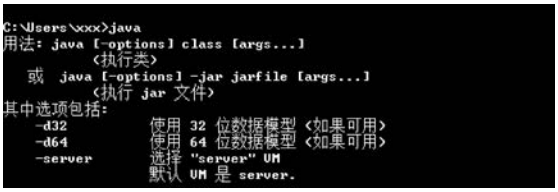


图 3-54 配置 JDK

（5）从官网上下载 Kettle 软件，由于 Kettle 是绿色软件，所以在下载后可以解压到任意目录。其网址是“http://kettle.pentaho.org”。本书下载的是 8.2 版本。

（6）运行 Kettle。在安装完成之后双击目录下面的 Spoon.bat 批处理程序即可启动 Kettle，如图 3-55 所示。

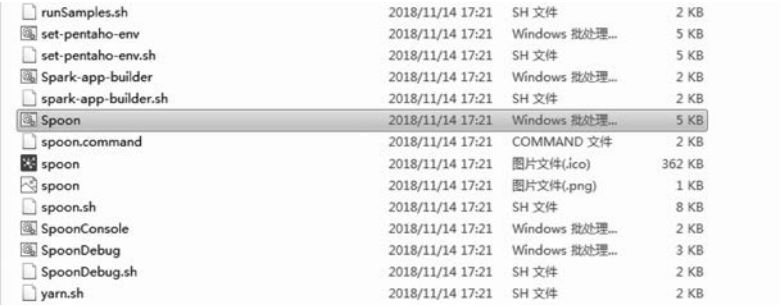


图 3-55 启动 Kettle

（7）Kettle 8.2 的运行界面如图 3-56 所示。

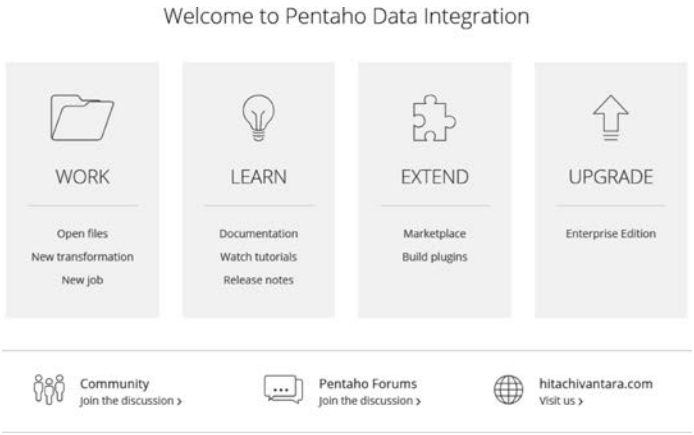


图 3-56 Kettle 8.2 的运行界面

(8) 成功运行 Kettle 后在菜单栏中单击“文件”,在“新建”中选择“转换”,在“输入”中选择“Excel 输入”,在“脚本”中选择“JavaScript 代码”,在“统计”中选择“分组”,将其分别拖动到右侧工作区中,并建立彼此之间的节点连接关系,最终生成的工作流程如图 3-57 所示。

(9) 准备好 Excel 数据表 file3-13,内容如图 3-58 所示。首先双击“Excel 输入”图标,在“文件”选项卡中将 file3-13 添加到 Kettle 中,如图 3-59 所示。然后在“工作表”选项卡中将要读取的工作表的名称选中,如图 3-60 所示。接着切换到“字段”选项卡,获取工作表中的字段名称,如图 3-61 所示。

图 3-57　工作流程

图 3-58　数据表内容

图 3-59　选中文件

(10) 双击“JavaScript 代码”图标,输入如下代码:

```
var 成绩为空 = 1;
if(成绩!= null){
成绩为空 = 0;
}
```

在“字段”中将“字段名称”设置为“成绩为空”,并设置“类型”为“Number”、“长度”为“16”、“精度”为“2”,如图 3-62 所示。

图 3-60　设置工作表名称

图 3-61　获取字段

图 3-62　设置 JavaScript 代码

（11）双击“分组”图标，在“聚合”中设置“名称”为“成绩为空”、“类型”为“求和”，如图 3-63 所示。

图 3-63　设置分组

（12）保存该文件，选择“运行这个转换”选项，可以在“执行结果”的 Preview data 选项卡中查看该程序的执行状况，如图 3-64 和图 3-65 所示。

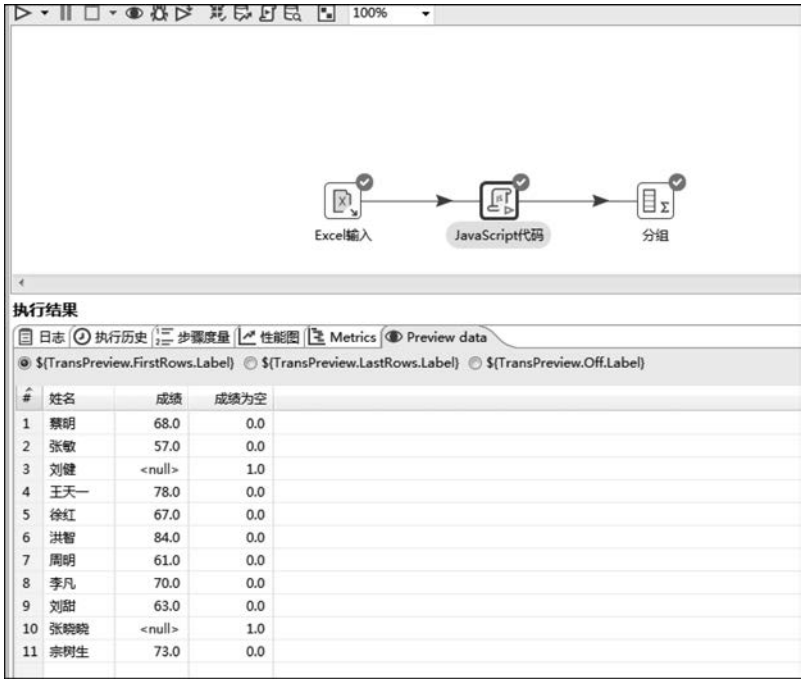


图 3-64　查看 JavaScript 代码结果

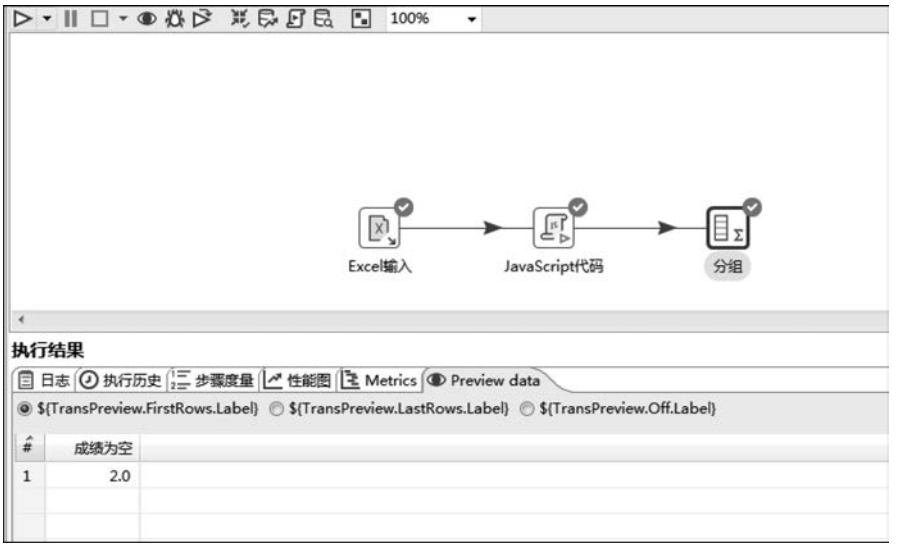


图 3-65　查看分组结果

最后在“成绩为空”中可以看到结果为 2.0,这表示有两个成绩值为空值。

习题 3

(1) 请阐述什么是数据质量。
(2) 请阐述什么是主数据。
(3) 请阐述什么是元数据。
(4) 请阐述元数据的特征。
(5) 请阐述元数据管理功能有哪些。
(6) 请阐述什么是数据血缘分析,如何实现?