(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 111712809 B
(45)授权公告日 2024.12.24

(21)申请号 201980013060.8
(22)申请日 2019.04.11
(65)同一申请的已公布的文献号
申请公布号 CN 111712809 A
(43)申请公布日 2020.09.25
(30)优先权数据
15/953,873 2018.04.16 US
(85)PCT国际申请进入国家阶段日
2020.08.12
(86)PCT国际申请的申请数据
PCT/US2019/026891 2019.04.11
(87)PCT国际申请的公布数据
WO2019/204106 EN 2019.10.24
(73)专利权人 甲骨文国际公司
地址 美国加利福尼亚
(72)发明人 M·莎森
(74)专利代理机构 中国贸促会专利商标事务所有限公司 11038
专利代理师 周衡威
(51)Int.Cl.
G06F 16/84(2006.01)
G06N 20/00(2006.01)
(56)对比文件
AnHai Doan等.Reconciling Schemas of Disparate Data Sources:A Machine-Learning Approach.SIGMOD '01: Proceedings of the 2001 ACM SIGMOD international conference on Management of data.2001,第509-520页.
审查员 刘文卓

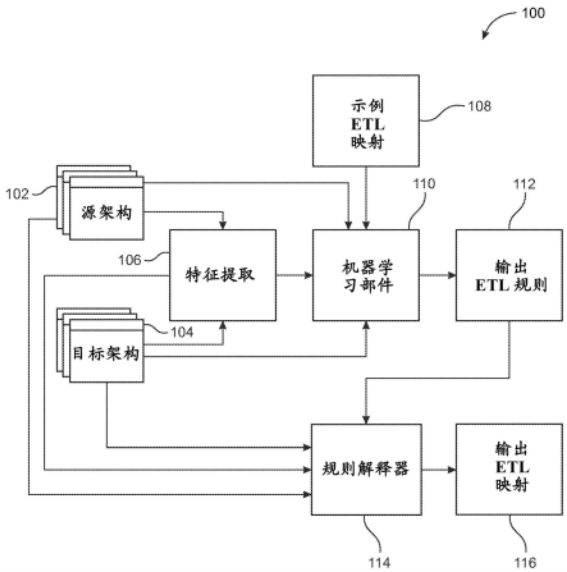
权利要求书3页 说明书27页 附图9页

(54)发明名称

通过示例来学习ETL规则

(57)摘要

实施例提供了用于通过示例来学习提取、变换和加载映射的系统和方法。可以从源架构和目标架构中提取多个特征。可以将示例ETL映射提供给机器学习算法,该示例ETL映射包括用于从源表提取数据并将提取出的数据加载到目标表中的定义。使用机器学习算法并基于源架构、目标架构和提取出的特征,可以预测一个或多个ETL规则,这些规则定义用于从源架构提取数据并将提取出的数据加载到目标架构中的逻辑。可以基于预测的ETL规则生成附加ETL映射,该附加ETL映射提供用于从源架构提取数据并将提取出的数据加载到目标架构中的附加定义。



CN 111712809 B

1.一种通过示例来学习提取、变换和加载ETL映射的方法,所述方法包括:

从源架构和目标架构中提取多个特征,所述特征至少包括所述源架构和所述目标架构的多个表的列;

向机器学习算法提供示例ETL映射,其中所述示例ETL映射包括用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的定义;

使用所述机器学习算法并基于所述源架构、目标架构和提取出的特征,预测一个或多个ETL规则,所述一个或多个ETL规则定义用于从所述源架构提取数据并将提取出的数据加载到所述目标架构中的逻辑,其中至少一个预测的ETL规则包括条件逻辑,所述条件逻辑选择性地将函数应用于从所述源架构提取出的数据值,所述函数对所述数据值进行变换;以及

基于预测的ETL规则、所述源架构、所述目标架构和提取出的特征生成附加ETL映射,所述附加ETL映射提供用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的附加定义,其中,在将提取出的数据加载到所述目标架构的目标表中时,用于至少一个附加ETL映射的定义选择性地将所述函数应用于从所述源架构的源表提取出的数据。

2.如权利要求1所述的方法,其中,所述示例ETL映射和附加ETL映射包括定义源表的一个或多个源列与目标表的目标列之间的关系的映射表达式。

3.如权利要求1所述的方法,其中,所述示例ETL映射包括实现列名后缀或列名前缀中的至少一者的映射表达式。

4.如权利要求1所述的方法,其中,所述至少一个预测的ETL规则将所述函数应用于源列。

5.如权利要求1所述的方法,其中

所述示例ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第一列,以及

所述附加ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第二列。

6.如权利要求1所述的方法,其中,所述预测的ETL规则定义由所述示例ETL映射表示的逻辑。

7.如权利要求6所述的方法,其中,

所述示例ETL映射表示所述源架构与所述目标架构的第一列集合之间的关系,并且

所述预测的ETL规则定义所述源架构与所述目标架构的第二列集合之间的关系的逻辑,并且所述第二列集合与所述第一列集合不同。

8.如权利要求7所述的方法,其中,所述附加ETL映射包括基于所述预测的ETL规则的用于所述目标架构的所述第二列集合的映射表达式。

9.如权利要求1所述的方法,其中,所述多个特征包括用于所述源架构和所述目标架构的表的外键。

10.如权利要求1所述的方法,其中,所述多个特征包括用于所述源架构的列的元数据,所述元数据指示所述源架构的列之间的关系。

11.一种其上存储有指令的非暂态计算机可读介质,所述指令在由处理器执行时使所述处理器通过示例来学习提取、变换和加载ETL映射,所述学习包括:

从源架构和目标架构中提取多个特征,所述特征至少包括所述源架构和所述目标架构的多个表的列;

向机器学习算法提供示例ETL映射,其中所述示例ETL映射包括用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的定义;

使用所述机器学习算法并基于所述源架构、目标架构和提取出的特征,预测一个或多个ETL规则,所述一个或多个ETL规则定义用于从所述源架构提取数据并将提取出的数据加载到所述目标架构中的逻辑,其中至少一个预测的ETL规则包括条件逻辑,所述条件逻辑选择性地将函数应用于从所述源架构提取出的数据值,所述函数对所述数据值进行变换;以及

基于预测的ETL规则、所述源架构、所述目标架构和提取出的特征生成附加ETL映射,所述附加ETL映射提供用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的附加定义,其中,在将提取出的数据加载到所述目标架构的目标表中时,用于至少一个附加ETL映射的定义选择性地将所述函数应用于从所述源架构的源表提取出的数据。

12.如权利要求11所述的计算机可读介质,其中,所述示例ETL映射包括实现列名后缀或列名前缀中的至少一者的映射表达式。

13.如权利要求11所述的计算机可读介质,其中,所述至少一个预测的ETL规则将所述函数应用于源列。

14.如权利要求11所述的计算机可读介质,其中

所述示例ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第一列,以及

所述附加ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第二列。

15.如权利要求11所述的计算机可读介质,其中,所述预测的ETL规则定义由所述示例ETL映射表示的逻辑。

16.如权利要求15所述的计算机可读介质,其中,

所述示例ETL映射表示所述源架构与所述目标架构的第一列集合之间的关系,并且

所述预测的ETL规则定义所述源架构与所述目标架构的第二列集合之间的关系的逻辑,并且所述第二列集合与所述第一列集合不同。

17.如权利要求16所述的计算机可读介质,其中,所述附加ETL映射包括基于所述预测的ETL规则的用于所述目标架构的所述第二列集合的映射表达式。

18.一种通过示例来学习提取、变换和加载ETL映射的系统,包括:

与存储器设备通信的处理设备,所述处理设备被配置为通过示例来学习提取、变换和加载ETL映射,所述学习包括:

从源架构和目标架构中提取多个特征,所述特征至少包括所述源架构和所述目标架构的多个表的列;

向机器学习算法提供示例ETL映射,其中所述示例ETL映射包括用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的定义;

使用所述机器学习算法并基于所述源架构、目标架构和提取出的特征,预测一个或多个ETL规则,所述一个或多个ETL规则定义用于从所述源架构提取数据并将提取出的数据加

载到所述目标架构中的逻辑,其中至少一个预测的ETL规则包括条件逻辑,所述条件逻辑选择性地将函数应用于从所述源架构提取出的数据值,所述函数对所述数据值进行变换;以及

基于预测的ETL规则、所述源架构、所述目标架构和提取出的特征生成附加ETL映射,所述附加ETL映射提供用于从所述源架构的一个或多个表提取数据并将提取出的数据加载到所述目标架构的一个或多个表中的附加定义,其中,在将提取出的数据加载到所述目标架构的目标表中时,用于至少一个附加ETL映射的定义选择性地将所述函数应用于从所述源架构的源表提取出的数据。

19.如权利要求18所述的系统,其中,所述至少一个预测的ETL规则将所述函数应用于源列。

20.如权利要求18所述的系统,其中

所述示例ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第一列,以及

所述附加ETL映射包括实现所述函数的映射表达式,用于加载所述目标架构的第二列。

通过示例来学习ETL规则

[0001] 相关申请的交叉引用

[0002] 本申请要求于2018年4月16日提交的美国专利申请No.15/953,873的优先权,其公开内容通过引用并入本文。

技术领域

[0003] 本公开的实施例一般而言涉及通过示例来学习提取(extract)、变换(transform)和加载映射(load mapping)。

背景技术

[0004] 计算设备和连接设备的激增产生了大量需要管理的数据。通常由人类(诸如数据科学家)执行的数据管理的一个方面是提取、变换和加载(“ETL”)或提取、加载和变换(“ELT”),两者将在整个本公开中可互换地使用。一般而言,ETL是将数据从源架构(source schema)移动到目标架构(target schema)的较大数据迁移处理的一个步骤。这种迁移处理通常具有多个层,诸如表/列映射、提取和加载规则/映射(例如,联接条件(join condition)、工作单元(unit of work)确定等)、迁移后和/或迁移前变换,这只是仅举的几例。因此,虽然数据管理的其它方面已变得敏捷和高效,但在利用现代常规技术时ETL可能是繁琐的处理。

发明内容

[0005] 本公开的实施例一般而言涉及通过示例来学习来提取、变换和加载映射的系统和方法,该系统和方法对相关技术进行了实质性的改进。

[0006] 在一些实施例中,可以从源架构和目标架构中提取多个特征,这些特征至少包括源架构和目标架构的多个表的列。示例ETL映射可以被提供给机器学习算法,其中示例ETL映射包括用于从源架构的一个或多个表中提取数据并将提取出的数据加载到目标架构的一个或多个表中的定义(definition)。使用机器学习算法并基于源架构、目标架构和提取出的特征,可以预测一个或多个ETL规则,这些规则定义用于从源架构提取数据并将提取出的数据加载到目标架构中的逻辑。可以基于预测的ETL规则、源架构、目标架构和提取出的特征生成附加ETL映射,这些附加ETL映射提供了用于从源架构的一个或多个表中提取数据并将提取出的数据加载到目标架构的一个或多个表中的附加定义。

[0007] 实施例的特征和优点在下面的描述中阐述,或者将从描述中变得清楚,或者可以通过实施本公开而获知。

附图说明

[0008] 通过以下结合附图对优选实施例的详细描述,其它实施例、细节、优点和修改将变得清楚。

[0009] 图1图示了根据示例实施例的用于通过示例来学习提取、变换和加载映射的系统。

[0010] 图2图示了根据示例实施例的可操作地耦合到ETL学习系统的计算设备的框图。

[0011] 图3图示了根据示例实施例的样本星型数据架构。

[0012] 图4图示了根据示例实施例的样本ETL规范。

[0013] 图5图示了根据示例实施例的由ETL工具实现的样本ETL结构。

[0014] 图6图示了根据示例实施例的样本架构映射结构。

[0015] 图7图示了根据示例实施例的样本ETL映射表达式。

[0016] 图8图示了根据示例实施例的样本属性向量。

[0017] 图9图示了根据示例实施例的用于根据示例实施例通过示例来学习提取、变换和加载映射的示例方法。

具体实施方式

[0018] 实施例通过示例来学习提取、变换和加载映射。源架构与目标架构之间的关系通常可能很复杂,因此使得用于ETL的机器学习应用变得不简单。实施例利用针对向机器学习算法提供粒度级别的数据点的源数据架构和目标数据架构的特征提取,其使得能够改善学习。此外,源数据架构和目标数据架构之间的示例映射被提供给根据ETL框架定义的机器学习算法。作为结果,机器学习算法可以基于示例中目标架构和源架构之间所示的关系来外推趋势。这些外推的趋势可以在有利于机器预测的详细级别进行定义,因为它们是在特定于架构的粒度级别定义的。随后,机器学习算法可以预测新的ETL规则,这些规则可以用于完成数据从源到目标的迁移。然后可以利用规则解释器基于预测的新ETL规则生成ETL映射。

[0019] 现在将详细参考本公开的实施例,其示例在附图中示出。在以下详细描述中,阐述了许多具体细节以便提供对本公开的透彻理解。但是,对于本领域的普通技术人员将清楚的是,可以在没有这些具体细节的情况下实践本公开。在其它情况下,没有详细描述众所周知的方法、过程、部件和电路,以免不必要地使实施例的各方面模糊。只要有可能,相同的参考数字将用于相同的元素。

[0020] 图1图示了根据实施例的用于通过示例来学习提取、变换和加载映射的系统。系统100包括源架构102、目标架构104、特征提取106、示例ETL映射108、机器学习部件110、输出ETL规则112、规则解释器114以及输出ETL映射116。源架构102和目标架构104可以是用于存储数据的任何数据架构,诸如关系数据表、平面文件(flat files)、NoSQL数据库、消息队列或主题、事件流或它们的组合。例如,源架构102和/或目标架构104可以包括根据规则或标准集合而配置的关系表集合,诸如第三范式(“3NF”)架构。

[0021] 一般而言,每个架构将包括具有一列或多列数据的数据表。架构不仅由表及其存储的数据定义,还由表之间的关系定义。例如,可以通过链接存储在第一表和第二表中的每个表中的数据的外键(foreign key)来定义第一表和第二表之间的关系。在一些实施例中,两个表可以共享多个关系(例如,可以具有定义表之间的关系的多个外键)。表之间的不同类型的关系将在本文中进一步公开。

[0022] 数据架构的设计通常可以基于设计者而变化。例如,具有给定关系集合的给定数据集可以成功地由具有不同设计的多个数据架构表示。一些设计可能需要表联接(table join)来检索特定的数据集,而其它设计则不需要。此外,许多遗留系统实现不容易被当前

已知数据库实现的架构设计。在考虑如何从源架构中提取数据并将数据加载到目标架构时,精细粒度细节(诸如表、列和键的命名约定等)在规则/映射构建中起着重要的作用。由于架构设计的高度主观性,将源架构102中存储的数据提取、加载和/或变换到目标架构104的规则/映射可能是特定于这些架构的。

[0023] 在实施例中,为了将机器学习部件110配置为生成准确的规则预测,对源架构102和目标架构104执行特征提取106。例如,特征提取106可以用于向机器学习部件110提供使得能够进行ETL规则/映射生成的精细粒度细节。进而,学习部件110可以以这种精细粒度进行分析,以预测源架构102和目标架构104的特定于架构的规则/映射。为每个架构提取出的特征可以包括表名称、列名称、表类型、表之间的外键、自引用表(self-referential)以及许多其它特征。这些提取出的特征可以表达架构中的通用、常用结构或模式、可以增强架构信息,并且可以简化机器学习部件110的工作。

[0024] 实施例还将由特征提取106提取出的特征提供给规则解释器114。例如,规则解释器114可以利用提取出的特征来解释预测的规则以生成输出ETL映射116。可供特征提取106提取的特征将在本文中进一步公开。

[0025] 实施例向机器学习部件110提供示例ETL映射108,以用于从源架构102提取数据并将数据加载到目标架构104中。示例映射108可以是源架构102和目标架构104的数据(例如,表)之间的特定映射。由于ETL映射108是特定于特定数据迁移任务的(例如,从源架构102迁移到目标架构104),因此学习部件110可以预测针对正在被实现的特定迁移任务的ETL规则。

[0026] 实施例将生成的输出ETL规则112传递给规则解释器114,规则解释器114可以解释规则以生成输出ETL映射116。例如,机器学习部件110可以生成一个或多个条件语句(例如,如果X和Y,则Z)形式的规则,并且规则解释器114可以通过解释这些规则来生成源架构102和目标架构104之间的实际映射。实施例将源架构102、目标架构104以及从特征提取106提取出的特征提供给规则解释器114。例如,可以基于源架构102和目标架构104的特定结构以及基于从这些架构提取出的特定特征,将输出ETL规则112解释为输出ETL映射116。

[0027] 在一些实施例中,当为源架构102和目标架构104之间的特定映射提供ETL映射108时,减轻了由架构设计的主观性引入的机器学习复杂度。例如,由于提供给机器学习部件110的特定训练数据代表源架构102和目标架构104之间的设计差异,因此由机器学习部件110预测的输出ETL规则112将基于这些特定设计差异,而不是基于使用通用架构关系训练数据时可能发生的通用设计差异。规则解释器112然后可以使用这些特定规则,并且使用架构102、目标架构104以及来自特征提取106的提取出的特征来生成也特定于源架构102和目标架构104的输出ETL映射。实施例实现的功能是一种自动化技术,用于生成特定源架构和特定目标架构之间的ETL映射,这在以往需要大量人力。

[0028] 在一些实施例中,源架构102、目标架构104、特征提取106、示例ETL映射108、机器学习部件110、输出ETL规则112、规则解释器114以及输出ETL映射116可以由单个系统或计算设备实现、跨各种计算设备分布,并且在一些实施方式中跨各种位置、这些的组合、可以以任何其它合适的方式来配置。在示例实施方式中,系统100的元件可以是软件、硬件或这些的组合。

[0029] 图2是根据实施例的计算机服务器/系统210的框图。如图2所示,系统210可以包括

被配置为在系统210的各个部件(诸如处理器222和存储器214)之间传送信息的总线设备212和/或(一个或多个)其它通信机制。另外,通信设备220可以通过对要通过网络(未示出)从处理器222发送到另一个设备的数据进行编码并且对通过网络从另一个系统接收到的数据进行解码来为处理器222实现处理器222和其它设备之间的连接。

[0030] 例如,通信设备220可以包括被配置为提供无线网络通信的网络接口卡。可以使用各种无线通信技术,包括红外、无线电、**Bluetooth®**、Wi-Fi和/或蜂窝通信。替代地,通信设备220可以被配置为提供(一个或多个)有线网络连接,诸如以太网连接。

[0031] 处理器222可以包括一个或多个通用或专用处理器,以执行系统210的计算和控制功能。处理器222可以包括单个集成电路,诸如微处理设备,或者可以包括多个集成电路设备和/或电路板,它们协同工作以完成处理器222的功能。另外,处理器222可以执行存储在存储器214内的计算机程序,诸如操作系统215、学习部件216和附加应用模块218。

[0032] 系统210可以包括用于存储信息和指令以供处理器222执行的存储器214。存储器214可以包含用于检索、呈现、修改和存储数据的各种部件。例如,存储器214可以存储在由处理器222执行时提供功能的软件模块。这些模块可以包括为系统210提供操作系统功能的操作系统215。这些模块可以包括操作系统215、学习部件216以及附加应用模块218。操作系统215为系统210提供操作系统功能。学习部件216可以提供用于预测从源架构提取数据并将其加载到目标架构中的ETL规则/映射的系统功能,或者还可以提供本公开的任何其它功能。在一些情况下,学习部件216可以被实现为内存中(in-memory)配置。

[0033] 非暂态存储器214可以包括可以由处理器222访问的各种计算机可读介质。例如,存储器214可以包括随机存取存储器(“RAM”)、动态RAM(“DRAM”)、静态RAM(“SRAM”)、只读存储器(“ROM”)、闪存、高速缓存存储器和/或任何其它类型的非暂态计算机可读介质。

[0034] 处理器222经由总线212进一步耦合到显示器224,诸如液晶显示器(“LCD”)。键盘226和光标控制设备228,诸如计算机鼠标,进一步耦合到通信设备212,以使用户能够与系统210接口。

[0035] 在一些实施例中,系统210可以是更大系统的一部分。因此,系统210可以包括一个或多个附加应用模块218以包括附加功能。附加应用模块218可以包括数据仓库的各种部件,该数据仓库包括例如操作系统和数据仓库目标、**Oracle®**数据集成器(**Oracle®** Data Integrator,“ODI”)、可以包括在基于**Oracle®**Golden Gate的源系统中更改的数据以及更改数据捕获(Change Data Capture,“CDC”)机制和基于Kafka流的处理的Apache Kafka,以及其它合适的部件。数据库217耦合到总线212,以为学习部件216和附加应用模块218提供集中式存储,并存储例如无线设备活动,并且在一些实施例中,存储用户简档、交易历史等。数据库217可以将数据存储在逻辑上相关的记录或文件的集成集合中。数据库217可以是操作数据库、分析数据库、数据仓库、分布式数据库、最终用户数据库、外部数据库、导航数据库、内存中数据库、面向文档的数据库、实时数据库、关系数据库、面向对象的数据库、Hadoop分布式文件系统(“HFDS”)或本领域已知的任何其它数据库。

[0036] 虽然示出为单个系统,但是系统210的功能可以被实现为分布式系统。例如,存储器214和处理器222可以分布在共同表示系统210的多个不同的计算机上。在一个实施例中,系统210可以是设备(例如,智能电话、平板电脑、计算机等)的一部分。

[0037] 在实施例中,系统210可以与设备分开,并且可以远程地为设备提供所描述的功

能。此外,可以不包括系统210的一个或多个部件。例如,对于作为用户或消费者设备的功能,系统210可以是智能电话或其它无线设备,其包括处理器、存储器和显示器,不包括图2所示的一个或多个其它部件,并且包括未在图2中示出的其它部件。

[0038] 图3图示了根据示例实施例的样本星型数据架构(star data schema)。数据架构300包括事实表(fact table)302、维度_1(dimension_1)表304、维度_2(dimension_2)表306和维度_3(dimension_3)表308。一般而言,在星型架构中,事实表保持关于域(domain)的事实,而维度表保持这些事实的属性。作为结果,事实表302与维度_1表304、维度_2表306和维度_3表308具有各种外键关系。换句话说,事实表中的行中的一些行是外键(“FK”)。外键可以是到维度表的链接。维度表可以是存储与一个或多个事实表引用的事件相关联的上下文(context)的表。

[0039] 星型数据架构类似于雪花型数据架构(snowflake data schema),但有一些区别。例如,雪花型数据架构包含被规范化为多个相关表的维度,而星型架构具有去规范化的维度,其中每个维度由单个表表示。这些架构中的每个架构都提供与数据冗余性、查询设计的简单性等相关的不同优点。例如,规范化的存储效率优点可以导致对查询规范化数据架构的效率的折中。

[0040] 返回参考图1,机器学习部件110依赖于特征提取106来提取描述源架构102、目标架构104和示例ETL映射108的可测量和可观察属性的特征。特征可以特定于域并且可以封装解决方案的典型方面当中的一些。一般而言,存在的特征越独立,越有可能学习。以下是可以从源架构102和目标架构104中提取的样本特征:

[0041] 表或列名称

[0042] 在将源映射到目标表/列时,名称可以作为决策的源。示例特征包括:

[0043] • 源和目标表/列名称相同

[0044] • 文件夹名称由带有前缀或后缀的源/目标表名称组成

[0045] • 映射名称由带有前缀和后缀的源或目标表名称组成

[0046] • 映射名称由在表名称之前、之后和之间带有固定串的源名称和目标名称组成

[0047] • 映射名称包含一个在名称相似的其它列中单调增加的数

[0048] • 源表和目标表相似,具有{1,2,3,4,大于4}的相似性距离

[0049] • 列名称以串“UDF”开始

[0050] • 表以“_L”或“_F”结尾

[0051] • 列名称是“EFFECTIVE_DT”

[0052] •

[0053] 提取出的特征当中的一些将被定义为框架的一部分,并且与特定表或前缀和后缀相关的一些特征可以基于源和目标架构来创建。

[0054] 列类型

[0055] 列类型可以用在各种ETL规则/映射决策中。根据各种实施例的示例列特征是:

[0056] • 源列和目标列类型相同

[0057] • 源是目标类型的子类型(例如,源是VARCHAR(20)并且目标类型是VARCHAR(30))

[0058] • 目标列类型为“日期(Date)”

[0059] • 列属于主键

[0060] • 列属于外键约束

[0061] • 列属于该表的外键

[0062] • 列具有“非空(Not Null)”约束

[0063] • 表的NUMBER类型的列是其它类型的{2,3,更多}倍

[0064] 表关系

[0065] 表关系可以用于确定联接或查找条件、排序和表标识。根据各种实施例的示例表关系特征是：

[0066] • 表是自引用的

[0067] • 表具有传入的外键约束

[0068] • 表具有传出的外键约束

[0069] • 表具有{1,2,更多}个传出的外键约束

[0070] • 表具有{1,2,更多}个传入的外键约束

[0071] • 表T1引用表T2

[0072] • 表T1通过传出的外键直接或间接连接到T2

[0073] • 表T1通过具有少于X个间接关系的传出外键直接或间接连接到T2。

[0074] 在一些情况下,根据底层架构,设计者的意图可能并不明显。对于某些ETL规则/映射决策,域专业知识可能是有用的。例如,通常不清楚在高度连接的数据模型中的哪些表要包含在数据集中。在一些情况下,对于表和列,附加元数据可能被添加到底层数据库架构中以启用期望的行为。例如,可以通过外键约束来连接10个表,但是只有它们当中的具有相同的“group1”属性值的一个子集可能被包含在源数据集中。在另一个示例中,可以为架构或表分配类型。例如,架构可以是具有事实、维度、层次结构或桥表(bridge table)的星型架构。下面说明可以在特征提取期间提取的某些元数据：

[0075] • 元数据属性X的值为Y

[0076] • 未定义元数据属性X

[0077] • 表T1的元数据属性X值与表T2的元数据属性X值相同

[0078] • 表T1的元数据属性X值比表T2的元数据属性X值{小,大}

[0079] • 元数据属性X与{源,目标}{表,列}Y具有相同的名称

[0080] 在一些实施例中,元数据可以将信息提供给机器学习部件110和/或规则解释器114,否则该信息将不容易从其它输入(例如,源架构102的其它结构、目标架构104的其它结构、其它提取出的特征以及示例ETL映射108)传送。元数据可以被表示为名称/值对(name-value pair),其取决于特定的特定于域的目的。在一些情况下,值可以是值的列表/数组。

[0081] ETL生成器的一些实施方式可以包括以下元数据中的一个或多个：

[0082] • “源表”可以在目标表中被使用来识别一个或多个源表。

[0083] 这可以用于解决歧义的情况,诸如由于不正确的命名约定而导致的不匹配。

[0084] • “模式名称”和“模式参与者”可以被用于源表和目标表,以阐明所使用的总体模式以及每个表(参与者)所起的作用。例如,这可以起到与ETL用例中常见的其它设计模式相似的作用。

[0085] • 源表和目标表中的“组”可以被用于定义结构的范围。例如,在高度连接的数据库架构中,要确定通过FK约束(其定义单元)而相关的表可能具有挑战性。在这些情况下,指

示分组是有帮助的。

[0086] • 目标架构表中的“过滤条件”可以被用于表达源表上的用以定义信息子集的条件。此类条件通常是由源系统中特定于域的详细信息驱动的。提供此类信息允许学习算法将该条件包括在生成的规则中。

[0087] 在一些实施例中,可以为目标ETL系统的一些标准化元数据元素实现默认行为。上述示例可以被包括在这样的通用元素中。在一些实施例中,元数据可以以特定于源架构的其它方式来使用。在一些实施例中,特征提取106从源架构102和目标架构104提取多个以上示例特征。

[0088] 在一些实施例中,输出ETL映射116与自动化架构映射的概念相关,或者与源架构102到目标架构104的映射相关。在本公开的实施例中描述的技术不仅实现架构映射,而且实现从源表提取数据以及将数据变换并加载到目标表中的过程,即,被称为ETL或ELT的处理。架构映射通常是预先发生的,以便成功启用ETL解决方案。在一些实施方式中,确定将哪个(些)源表列映射到哪个(些)目标表列就足够了。在简单的情况下,匹配的源表列和目标表列使用足够相似的名称并定义相关的数据类型。

[0089] 实施例提供了将源架构102映射到目标架构104的示例ETL映射108。示例ETL映射108可以以能够容易地被分析的任何形式来表示,并且是生成ETL可执行体(例如,被配置为执行以进行数据迁移的输出ETL映射116)的基础。例如,在各种实施例中使用的ETL可扩展标记语言(“XML”)规范格式可以类似于来自ETL工具Oracle数据集成器(“ODI”)的ETL XML规范。在一些实施例中,示例XML ETL规范可以包括多个部分:

[0090] • 映射的定义

[0091] o映射的名称

[0092] o位于ETL软件工具中的文件夹名称(例如,逻辑文件位置)

[0093] o源表以及如何过滤、查找和联接源。在各种实施例中,源可以存在于源架构和目标架构中。

[0094] o目标表

[0095] o使用参数而配置的提取(例如,结构化查询语言(SQL)或数据库链接)和插入策略(例如,递增、合并、追加)

[0096] o用于每个目标列的映射表达式,其可能包含常量、1-1列映射,或者用于一个或多个源列的表达式

[0097] • 变量

[0098] o变量名称

[0099] o变量类型

[0100] o变量定义

[0101] • 序列

[0102] o序列的名称

[0103] o序列的类型(例如,DB序列)

[0104] o序列定义

[0105] • 如何编排ETL映射和其它ETL原语的定义(也称为包)

[0106] o包名称

[0107] o位于ETL工具中的文件夹名称

[0108] o映射的顺序或并行执行

[0109] o变量的初始化

[0110] o过程的执行

[0111] 除了XML规范之外,源数据库模型和目标数据库模型还包含要考虑的特征:

[0112] • 源架构和目标架构的名称

[0113] • 表名称和列名称

[0114] • 列的类型

[0115] • 非空、主键和外键约束

[0116] • 唯一索引

[0117] 在一些实施例中,变量可以是ETL工具的制品(artifact),以维持可以在映射和ETL编排定义中引用的状态(数字或串)。在一些实施例中,序列可以是ETL定义的一部分,通常在数据库本身中定义。例如,序列可以被提供为数据库的数据定义语言(“DDL”)脚本,或者可以被单独地生成。

[0118] 在各种实施例中,从源架构到目标架构的数据的提取、变换和加载被结构化为子项目集合,其中任一个或一组源被提取、变换并加载到一个或一组目标表中。例如,ETL设计可以包括两个几乎独立的级别:作为工作单元的一部分确定将映射哪些表,以及设计源列和目标列之间的映射表达式。工作单元可以包含一个或多个映射以及ETL解决方案的各种其它详细信息,诸如过程、序列定义、变量定义等。下面提供用于这两个级别的示例表和列映射:

[0119] 表映射设计

[0120] 表级别ETL设计考虑哪些源表和目标表参与工作单元。存在一些常见的模式:

[0121] • 一对一模式:一个源表被映射到一个目标表。

[0122] • 多对一模式:多个源表被联接到一个数据集中,并且被映射到一个目标表。当使用星型架构将数据从操作系统加载到数据仓库中时,这是常见的场景。例如,表示维度层次结构的多个表被联接以产生去规范化的维度表。

[0123] • 一对多模式:一个源表可以将记录的全部或子集映射到多个目标表,这些目标表包含源列的全部或子集。这个方法可以用于将不相关的表属性分离到单独的表中。

[0124] • 类型-子类型模式:一个源表被映射到多个目标表,这些目标表将公共数据分离到类型表中,并且将特定详细信息分离到一个或多个子类型表中。这是上面讨论的一对多模式的特例。反过来是可能的,将具有类型-子类型关系的源表映射到一个目标表。

[0125] • 聚合表模式:聚合表通常是在目标架构内使用关于测量的聚合函数和“group by”子句定义来聚合级别从而从具有更精细的粒度数据的表导出的。

[0126] 列映射设计

[0127] 列映射具有与列的类型、列类型和名称以及可用源列相关的多个常见变型:

[0128] • 属于表的主键并且可以从源复制、使用DB序列创建或使用查找操作(例如,利用左外联接(left outer join))从父表复制的列。

[0129] • 外键列表示在目标架构中(例如,利用左外联接)对所引用的表执行查找,以基于自然键(natural key)检索外键列。

[0130] • 列可以简单地在源表和目标表之间进行1-1复制。

[0131] • 可以使用简单的表达式(诸如CAST)将列映射到目标列的类型。

[0132] • 某些系统列(例如,数据源ID、日期更新或插入数据)可以被一致命名,并将使用相同的表达式。

[0133] 在一些实施例中,一个或多个源表到一个目标表的映射被组织成工作单元。在简单的情况下,一个映射是驻留在文件夹或包中的工作单元。在一些实施方式中,文件夹可以是对相关映射、变量定义、序列定义或过程进行分组的概念。类似地,包可以定义这些相关ETL元素被执行的顺序。

[0134] 在一些实施例中,类型-子类型模式可以被实现为将源表映射为类型和一个或多个子类型表的多个映射。这些映射可以创建位于文件夹或包中的一个工作单元。另外,表结构可能要求首先加载包含类型信息的目标表,然后再加载其它所有表,或者可能并行加载。

[0135] 在向机器学习部件110提供提取出的特征和示例ETL映射108之后,可以预测新的ETL规则/映射,以从源架构102中提取数据并将该数据加载到目标架构104中。在一些实施例中,可以用不同的模块来定义解决方案框架,以确定解决方案的各个方面。例如,可以将这些模块中的一个或多个模块设计为灵活且可配置的,使得可以为特定的源/目标架构定义特定的解决方案。在一些示例中,决策可以被表达为规则,并且在一些示例中,决策可以被表达为概率或权重。

[0136] 传统的ETL生成器通常对这些规则进行硬编码。一些生成器允许ETL团队手动地编写或配置规则,以使ETL生成器框架适应特定情况。这种方法比传统的生成器具有明显的优势,但是它仍然需要大量的工时。作为对照,在各种实施例中,针对ETL处理的不同方面,可以使用所提供的示例ETL映射108来自动地(即,无需人工干预)学习ETL规则和配置。自动生成ETL规则的处理可以被分解成多个组成部分,并且每个组成部分可以涉及能够被学习的决策。实施例然后可以将输出ETL规则112提供给规则解释器114以得出输出ETL映射116,其可以用于实现架构之间的迁移。

[0137] 在一些实施例中,机器学习部件110可以包括多个模块,这些模块可以具有基于从源架构102、目标架构104以及示例ETL映射108提取出的特征而学习的方面。例如,可以使用监督学习算法来完成学习,该监督学习算法学习确定ETL解决方案的各方面的架构、表和列特征的组合,它们是从示例ETL映射108中观察到的。

[0138] 在许多方面,学习不是普遍或通用的。例如,ETL解决方案的(例如,提供示例ETL映射108的)设计者可以基于所提供的示例(并且在一些情况下,所提供的元数据)来定制学习,以使用一种或多种学习技术来适应特定问题。

[0139] 在一些实施例中,可以在学习ETL规则之前执行各种准备。例如,表可以捕获操作数据,诸如行的数据源、初始加载的日期和时间、最后更新的日期和时间、最后更新记录的人员的用户ID等。这些列通常是所有或大多数表拥有的系统列。在一些情况下,表的组可以定义作为共享设计的指示的公共列。示例包括类型-子类型模式的类型列、版本记录的生效日期和当前标志列、或指示层次结构的通用LVL<n>列。可以将架构中经常使用的列转换成附加特征(例如,“isCommonColumnType_x”),这些附加特征可以在学习期间被机器学习部件110用来预测附加ETL规则。

[0140] 在另一个示例中,作为附加特征而导出关于架构的结构的信息可以帮助学习。一

个经典差异在于高度规范化的操作架构和去规范化并以星型组织的报告架构(例如,雪花型架构与星型架构)。通常可以使用能够确定架构是否由星组成的算法来进行这种确定。在授予Michael Sassin的题为AUTOMATIC GENERATION OF LOGICAL DATABASE SCHEMAS FROM PHYSICAL DATABASE TABLES AND METADATA的美国专利申请公开号2016/0078064中提供了进行这种确定的示例算法。

[0141] 相关的算法可以将星型架构表分类为事实、维度、层次结构、桥表、查找表等。在大多数情况下,算法可以使用外键约束来进行这些确定。也可以学习命名约定、以及某些列或列中数据类型的出现、或NUMBER类型与不是数字类型的列之间的比率。

[0142] 在一些实施例中,为了执行这种类型的学习,可以提供具有预分类的表类型的较大的星型架构数据集。规范化的架构表也可以使用类似的技术进行分类。在一些实施方式中,目标可以是识别模型模式或设计原型。在其它实施例中,该信息可以作为元数据提供,或者可以以其它方式对系统的学习部件可用。

[0143] 在一些实施例中,ETL映射被生成为工作单元。例如,先前公开的是用于基于由架构映射算法确定的列映射来识别表的技术。在一些实施例中,由架构映射定义的工作单元基于由示例ETL映射108定义的工作单元。

[0144] 在一些实施例中,可以对ETL解决方案的工作单元进行分类。每个工作单元可以遵循先前公开的为特定上下文中的类似问题提供可重用解决方案的解决方案模式。针对使数据去规范化、映射类型-子类型模式等建立了多种解决方案模式。分类作为附加的派生特征可能是有帮助的,因为它可以作为更复杂规则的捷径(shortcut)。在一些实施例中,算法被用于预测工作单元的模式。模式名称可以作为元数据被提供给机器学习部件110。

[0145] 在一些实施例中,学习算法可以被实现为分类算法或规则学习算法。在一些情况下,在尝试学习复杂决策时,通过示例来学习可能会遇到挑战。实施例还可以包括监督的实施方式,其中基于所获得的学习提示用户提供输入(例如,选择)。在S.Bocionek和M.Sassin的Dialog-Based Learning(DBL)for Adaptive Interface Agents and Programming-by-Demonstration Systems(Technical Report CMU-CS-93-175,卡内基梅隆大学,1993年7月)中进一步讨论了示例方法。

[0146] 在学习工作单元时,通常需要联接条件以诸如从星型架构或任何其它合适的架构中提取数据集。例如,可以使用利用表之间的外键约束定义的列的条件来联接表。取决于外键列的NOT NULL约束,联接可以是内部联接或外部联接。在一些实施例中,可以基于参与表的特征来学习这种标准方法的变型(例如,参与联接的列是替代键、索引或仅具有固定名称等)。

[0147] 在学习工作单元时,也学习源架构102和目标架构104之间的映射表达式。映射表达式可以分为多个大类(broad categories),包括:

[0148] • 主键列的映射

[0149] • 外键列的映射

[0150] • 代码列的映射

[0151] • 系统列

[0152] • 其它列

[0153] 在一些实施例中,使用代理键或通过(经由外键)查找父表以检索现有的主键列,

来创建主键列。例如,可以分析目标映射表达式以做出此类确定。代理键值可以被定义为数据库序列,其可以在目标映射表达式中。如果使用了自然键,那么表达式可以在目标映射表达式中定义源表的主键列。

[0154] 在一些实施例中,源架构102中的外键列被用于在源架构102中的父表中查找值。使用查找操作以类似的方式将代码列变换为外键列。在这些示例中,存在不是学得的而是由ETL框架提供的一组完善的查找或创建模式。

[0155] 例如,相关表达式可以由示例ETL映射108定义。例如,在将维度表加载到星型架构中时可以使用公共模式。维度表可以将代理键定义为主键,并提供源自源系统的自然键作为查找键。在一些情况下,维度表可以充当使用自然键的查找(lookup),来确定维度行是否已经被加载,在这种情况下,新到达的数据被视为触发维度更新的更改记录。特定的详细信息(诸如,列以及加载或更新维度的动作和/或序列)可能会因实施方式而异。对于ETL规则的每个实例,可以识别源表和目标表中的参与列,并且可以学习与它们的选择相关联的规则。

[0156] 例如,假设表S被映射到目标表T,目标表T与父表T2具有外键关系。为了填充T中的外键列,源表S应该具有用于该关系的在T2中也可用的一个或多个自然键列。在一些实施例中,学习算法可以使用在示例ETL映射108中使用的列的(例如,由特征提取106提取的)提取出的特征来学习关于外列名称的约定。在大多数情况下,命名约定中使用某种级别的规则性(例如,名称使用父表的名称,并使用相同的前缀或后缀)。在其中源表或目标表未遵循可以由特征表达的约定的一些情况下,可以添加列级别元数据来对架构进行补充。

[0157] 在一些实施例中,编码值的查找遵循与外键查找的情况相似的规则。潜在的区别是,检索的是映射值而不是行的键列。在其它方面,代码列可以遵循与键列相似的处理。

[0158] 在一些实施例中,系统列也遵循规则,诸如复制具有常数名称的源值、插入函数(例如,系统时间)、常数(例如,CURRENT_FLG被设置为'Y'),或基于提供的身份(例如,最后更新的用户的用户ID)来查找表。这些规则可以基于示例ETL映射108而学习得到。

[0159] 在一些实施例中,可以存在不属于以上类别之一的其它列。在许多情况下,这些列可以从源列进行1-1映射(例如,数据被简单地传递到目标列)。常见的变型是针对NULL值检查源数据并且提供默认值,或者将源列截断或强制转换成不同的类型。这些规则也可以基于示例ETL映射108而学习得到。

[0160] 在一些实施方式中,一些ETL规则/映射可能很复杂,因此可能不利于基于示例的学习环境。在这些情况下,学习系统可以跳过复杂的规则/映射,并且用户可以手动地生成这些规则/映射以实现最终的ETL解决方案。换句话说,在源架构与目标架构之间包括复杂规则和/或映射的一些实施例可以依赖于自动化规则和映射生成与手动规则和映射生成的混合。

[0161] 考虑一种用例,该用例包括源架构、目标架构、示例ETL映射和学得的ETL规则。例如,附录包含用于生成源架构和目标架构(带有父架构和子架构)的SQL。源架构的实施方式可以是数据仓库暂存(staging)架构。目标架构的实施方式可以是具有更改记录的3NF企业数据仓库架构。

[0162] 实施例可以在源表被用于加载一个或多个目标表的情况下实现ETL规则和/或映射。记录版本控制可以被用于使用自然键和版本号来查找架构的外键。可以提供元数据元

素,包括过滤条件。公共映射表达式可以被用于源架构和目标架构的系统列。一些示例ETL映射包括简单的1-1列映射。

[0163] 在一些实施方式中,源架构和目标架构可以包括超过500个表。这样的ETL项目可能需要大量的人力、时间(例如,年)和资源来手动执行。但是,许多映射可以通过少量ETL模式来表征(例如,90%的映射可以通过2或3个ETL模式来表征)。在一些实施方式中,实施例可以学习大约100个低复杂度的规则。在所考虑的用例中,源架构和目标架构遵循已定义的命名约定。

[0164] 示例ETL系统的部件包括ETL工具、ETL规范、ETL转换器、ETL生成器、规则解释器(例如,类似于图1的规则解释器114)和学习部件(例如,类似于图1的学习部件110)。在给定的实施方式中,规则可以是具有相关联的表达式的决策和条件。在一些情况下,模板或模板的一部分可以被链接到规则。

[0165] ETL工具的实施例可以表达ETL部件和映射。例如,ETL工具可以将定义转换成针对源系统和目标系统运行的可执行体。Oracle®数据集成器(“ODI”)ETL工具可以是在一些实施方式中使用的示例。ETL规范的实施方式可以是指定ETL解决方案的人类可读的和机器可读的文本格式定义。在一些实施例中,ETL规范大部分独立于目标ETL系统。示例ETL规范包括由Oracle®数据仓库工厂ETL转换器(“DWF ETL”)使用的ETL规范。

[0166] 图4图示了根据实施例的示例ETL规范。规范400可以包括用于以下各项中的一项或多项的定义:文件夹和制品名称;对源表和目标表的引用;过滤器、联接、查找、联合和聚合;它们的目标列和表达式;关于如何从表中提取和向表中加载数据的附加指示;规范400以XML格式定义,但是可以实现任何其它合适的格式(例如,JavaScript对象表示法(“JSON”)、YAML非标记语言(“YAML”)、纯文本等)。

[0167] ETL转换器的实施例可以将ETL规范转换成由目标ETL工具使用的定义。DWF ETL实现了用于生成ODI定义的示例转换器。ETL生成器的实施例可以通常根据源和目标表定义来生成ETL规范。例如,生成的ETL规范可以基于导出的元数据、从源架构和/或目标架构提取出的特征,和/或显式元数据。在一些实施方式中,可以例如基于自动化生成、手动生成或这些的组合来提供ETL规范。

[0168] 学习部件的实施例可以在ETL框架(例如,ETL规范和其它ETL配置)和ETL生成器的上下文中定义特定决策。如参考图1的机器学习部件110所公开的,学习部件可以基于示例ETL映射生成ETL规则。在这种考虑的用例中,示例ETL映射可以被表达为ETL规范定义和/或架构元数据。学习可以特定于架构的(一个或多个)不同规则/映射/数据组。在一些实施例中,学习部件可以被配置为实现多个机器学习算法,每个机器学习算法被配置为针对不同的规则/映射/数据组执行特定的学习任务(并进行特定的机器预测)。

[0169] 实施例包括利用机器学习来生成源架构与目标架构之间的架构映射的一部分。例如,基于提供给学习部件的示例ETL映射,可以学习和预测映射的一部分。在一些实施方式中,该部分可以是来自源架构的大比例的数据(例如,表/列的90%)。一些实施方式包括利用手动映射来生成全面集合。在一些实施例中,可以提供足够的示例ETL映射,以允许学习部件生成映射的全面集合。

[0170] 由学习部件执行的ETL学习的各个方面包括ETL解决方案的细节。图5图示了根据实施例的ETL工具(例如,ODI)的文件夹结构。结构500包括文件夹502,其可以是在映射和其

它ETL过程中使用的ODI文件夹和/或子文件夹的名称。文件夹命名方案的实施例可以包括常数(例如,“LOAD_DI_HDWF...”,如文件夹502的元素所示)和/或可以包含源/目标表的名称。结构500还可以包括ETL元素504,其可以是包、映射和过程。这些ETL元素504也可以包括源/目标表的名称。通常可以基于与映射相关的源表或目标表对映射进行分组。

[0171] 图6图示了根据实施例的映射结构。例如,映射结构600包括在给定映射中使用的表602。这些表通常在ETL的“模式”中定义。表可以经由映射结构中的外键而被引用。过滤器604可以应用于表602,例如应用于源表。示例过滤器功能可以包括基于规则/条件的行过滤和列过滤。过滤器(例如,过滤器的规则/条件)可以通过元数据来提供,或者可以通过其它机制来提供。过滤功能可以是ETL处理流程中数据操纵和/或变换阶段的一部分。例如,过滤器604可以表示在相关映射中用于将源数据从源数据表映射到目标数据表的数据操纵。

[0172] 实施例包括执行联接以操纵、变换和/或过滤数据(例如,源数据)。可以使用外键约束来执行联接。查找606可以是对映射结构中的相关目标数据表的引用。可以类似地基于外键约束来定义查找606。映射结构600可以表示根据各种实施例的用于在源架构和目标架构之间进行映射的结构。

[0173] 图7图示了根据实施例的映射表达式。例如,结构700包括映射表达式702、704、706和708。映射表达式可以将(例如,来自数据表的)源数据映射到目标属性和数据表。在一些情况下,可以将源列映射到具有相似名称的目标列,如映射表达式704所示。在一些情况下,可以在映射表达式中使用诸如sysdata之类的系统函数,如映射表达式708中所示。

[0174] 在一些情况下,可以在映射表达式中使用变量,诸如捕获以“:”或“#”开头的串的变量,如映射表达式702中所示。在一些情况下,函数可以用于包装列,诸如CAST、NVL或某个其它自定义函数,如映射表达式706中所示。实施例还包括映射表达式内的聚合函数,诸如SUM、AVG、Group By等。

[0175] 参考所考虑的用例以及附录中公开的源架构和目标架构,所实现的学习部件可以基于源架构、目标架构、提取出的特征和示例映射(以下进一步公开)来学习数据中的多个趋势。表中的系统列可以在源表和目标表中系统地使用。例如,可以添加、更新或删除数据记录,可以创建和更新用户,标记功能可以保持一致,等等。

[0176] 学习部件的实施例可以学习在大量源表和/或目标表中出现的具有相同名称的列的表达式。在一些情况下,列映射可以为每个列名称定义一个或几个变型(例如,基于类型)。学习部件可以具体地确定针对这种变型的标准。

[0177] 实施例可以包括选择数据以训练学习部件。例如,在示例ETL映射中选择到同一目标列的映射可以改善一些实施方式的学习成果。可以基于与参与列直接相关的元数据来执行关于列映射的决策,在某些情况下,可以扩展元数据以包括用于列的表的元数据,例如以包括诸如名称或模式类型之类的信息。所考虑的用例的示例列元数据包括名称、与名称相关的特征(诸如前缀或后缀)、列数据类型、附加添加的元数据元素等。

[0178] 图8图示了根据示例实施例的样本属性向量。例如,属性向量800可以包括与所考虑的用例相关的源架构、目标架构和提取出的特征数据。列802描绘了源列属性,而列804描绘了列目标属性。特征806描绘了从源架构中提取出的特征,诸如相关列名称的前缀和后缀。映射表达式808描绘了要由例如学习部件学习的映射表达式。

[0179] 在实施例中,可以在用例中向学习部件提供多个示例映射表达式。示例映射表达

式“trg.update_date＝src.update”在用例数据(例如,附录中的源架构和目标架构)中出现5次。另外,示例映射表达式“trg.update_date＝CAST(src.update,DATE)”在用例数据中出现4次。源列update_date在第一示例映射集合中使用日期(Date)类型,并且在第二示例映射集合中使用时间戳(Timestamp)类型。相应地,在第二示例集合的映射表达式中使用了CAST函数。在所示的用例中,学习部件的目标可以是学习何时应用相似的映射表达式,并在适当的情况下选取这些映射表达式之一(例如,是否为CAST)。

[0180] 学习部件的实施例可以实现ID3决策树学习算法或任何其它合适的学习算法,以实现期望的学习/预测。例如,学习部件可以学习逻辑:

[0181] if exist(src.update_date)&type(src.update_date)＝‘Timestamp’&

[0182] exist(trg.upate_dt)

[0183] then trg.update＝CAST(src.upate_date,DATE)

[0184] else trg.update_dt＝src.upate_date end

[0185] 在示例用例中,学习部件的实施例可以学习/预测用于系统列trg.update_dt的映射。该学习/预测可以基于提供给该学习部件的相似示例映射(例如,以上针对“trg.update”和“srcupdate”的映射表达式)。另外,在用例中,实施例可以学习/预测逻辑以确定何时选择具有“CAST”函数的映射表达式以及何时选择其它映射表达式。学得的映射表达式的逻辑类似于所提供的示例映射表达式中的逻辑,即,当数据类型为时间戳(Timestamp)时使用CAST。类似地,其它用例可以提供其它示例映射表达式,这些示例映射表达式说明了源架构与目标架构之间的模式,其中学习部件可以学习何时以及如何将相似的映射表达式应用于数据的其它方面以预测ETL规则。

[0186] 考虑到用例,可以通过规则解释器来解释学得的ETL规则。例如,规则解释器可以解释上述学得的逻辑,并且基于源架构、目标架构和提取出的特征,根据规则(和所表示的逻辑)生成ETL映射。具体地,可以为trg.update_dt列生成ETL映射。然后,可以将生成的映射用在ETL解决方案中,并最终可以被用于实现从源架构到目标架构的数据迁移。

[0187] 虽然在实施例中考虑了传统的ETL,但是也可以实现流处理ETL。例如,Apache Kafka可以被包含在各种实施方式中,其中消息流可以由各种流处理框架使用,诸如Kafka Streams应用编程接口(“API”)、Apache Storm、Apache Spark、Apache Samza等。在一些实施例中,可以实现流处理而不是批处理,来实现所公开的各种解决方案。

[0188] 图9图示了根据示例实施例的用于通过示例来学习提取、变换和加载规则的示例功能。在一个实施例中,下面图9的功能由存储在存储器或其它计算机可读或有形介质中的软件实现,并由处理器执行。在其它实施例中,每个功能可以由硬件执行(例如,通过使用专用集成电路(“ASIC”)、可编程门阵列(“PGA”)、现场可编程门阵列(“FPGA”)等),或由硬件和软件的任何组合执行。

[0189] 在902处,可以从源架构和目标架构中提取多个特征,这些特征至少包括源架构和目标架构的多个表的列。在一些实施例中,特征还可以包括列的标识符、列的元数据、表的外键以及其它合适的特征。在一些示例中,源架构中的列的元数据可以指示该列与源架构的其它列/表之间的关系,或者可以指示其它合适的数据关系。

[0190] 在904处,示例ETL映射可以被提供给机器学习算法,其中示例ETL映射是用于从源架构的一个或多个表中提取数据并将提取出的数据加载到目标架构的一个或多个表中的

定义。示例ETL映射可以是映射表达式,这些映射表达式定义源表的一个或多个源列与目标表的目标列之间的关系。

[0191] 在一些实施例中,示例ETL映射可以包括实现列名后缀或列名前缀中的至少一者的映射表达式。在一些实施例中,示例ETL映射可以包括映射表达式,该映射表达式实现该映射表达式的应用于源列的一个或多个函数。

[0192] 在906处,使用机器学习算法并基于源架构、目标架构和提取出的特征,可以预测一个或多个ETL规则,该一个或多个ETL规则定义用于从源架构提取数据并将提取出的数据加载到目标架构中的逻辑。在一些实施例中,预测的ETL规则可以定义由示例ETL映射表示的逻辑。

[0193] 在实施例中,示例ETL映射表示源架构与目标架构的第一列集合之间的关系。预测的ETL规则可以定义用于源架构和目标架构的第二列集合之间的关系的逻辑,其中第二列集合与第一列集合不同。在实施例中,附加的ETL映射包括基于预测的ETL规则的用于目标架构的第二列集合的映射表达式。

[0194] 在908处,可以解释预测的ETL规则。例如,规则解释器可以解释ETL规则中定义的逻辑,以生成源架构和目标架构之间的映射表达式。解释可以基于源架构、目标架构和提取出的特征。在一些实施例中,实现了定义ETL解决方案的细节的ETL规范,并且解释可以基于该ETL规范。

[0195] 在910处,可以基于预测的ETL规则、源架构、目标架构和提取出的特征来生成附加的ETL映射,附加的ETL映射提供用于从源架构的一个或多个表提取数据并将提取出的数据加载到目标架构的一个或多个表中的附加定义。例如,规则解释器可以用于生成附加ETL映射。在一些实施例中,附加ETL映射可以是映射表达式,其定义源表的一个或多个源列与目标表的目标列之间的关系。

[0196] 在实施例中,示例ETL映射可以包括映射表达式,该映射表达式实现应用于源架构的列以加载目标架构的第一列的第一类型的函数。附加ETL映射可以包括映射表达式,该映射表达式实现应用于源架构的该列以加载目标架构的第二列的第一类型的函数。

[0197] 实施例通过示例来学习提取、变换和加载映射。源架构与目标架构之间的关系通常可能很复杂,因此使ETL的机器学习应用变得不简单。实施例利用针对向机器学习算法提供粒度级别的数据点的源数据架构和目标数据架构的特征提取,其使得能够改善学习。此外,将源数据架构和目标数据架构之间的示例映射提供给根据ETL框架定义的机器学习算法。作为结果,机器学习算法可以基于示例中目标架构和源架构之间所示的关系来外推趋势。这些外推的趋势可以在有利于机器预测的详细级别进行定义,因为它们是在特定于架构的粒度级别定义的。随后,机器学习算法可以预测新的ETL规则,这些规则可以用于完成数据从源到目标的迁移。然后可以利用规则解释器基于预测的新ETL规则生成ETL映射。

[0198] 贯穿本说明书描述的本公开的特征、结构或特性可以在一个或多个实施例中以任何合适的方式组合。例如,贯穿本说明书的“一个实施例”、“一些实施例”、“某个实施例”、“某些实施例”或其它类似语言的使用是指这样的事实,即结合该实施例描述的特定特征、结构或特性可以被包括在本公开的至少一个实施例中。因此,贯穿本说明书的短语“一个实施例”、“一些实施例”、“某个实施例”、“某些实施例”或其它类似语言的出现不一定全都指同一组实施例,并且在一个或多个实施例中,所描述的特征、结构或特性可以以任何合适的

方式组合。

[0199] 本领域普通技术人员将容易理解,可以以不同顺序的步骤和/或以与所公开的配置不同的配置的元件来实践如上所讨论的实施例。因此,虽然本公开考虑了概述的实施例,但是对于本领域技术人员而言清楚的是,某些修改、变型和替代构造将是清楚的,同时仍在本公开的精神和范围内。因此,为了确定本公开的范围,应参考所附权利要求。

[0200] 附录:

[0201] 源表:

```
CREATE TABLE "OHF711_HDI"."HDI_ENC"
        ( "ID" NUMBER(38,0) NOT NULL ENABLE,
          "INT_ID" VARCHAR2(400 BYTE) NOT NULL ENABLE,
          "DAT_SRC_NUM_ID" NUMBER(10,0) NOT NULL ENABLE,
          "SRC_EFF_FROM_DT" DATE,
          "SRC_EFF_TO_DT" DATE,
          "PT_INT_ID" VARCHAR2(400 BYTE),
          "PT_DAT_SRC_NUM_ID" NUMBER(10,0),
          "PT_SRC_CHANGED_ON_DT" DATE,
          "SUBTYP_CD" VARCHAR2(80 BYTE),
          "SUBTYP_CDG_SYS" VARCHAR2(80 BYTE),
          "SUBTYP_CDG_SYS_VER" VARCHAR2(80 BYTE),
          "ENC_STUS_CD" VARCHAR2(80 BYTE),
          "ENC_STUS_CDG_SYS" VARCHAR2(80 BYTE),
          "ENC_STUS_CDG_SYS_VER" VARCHAR2(80 BYTE),
```

[0202]

```
          "CD" VARCHAR2(80 BYTE),
          "CDG_SYS" VARCHAR2(80 BYTE),
          "CDG_SYS_VER" VARCHAR2(80 BYTE),
          "ADMSN_TYP_CD" VARCHAR2(80 BYTE),
          "ADMSN_TYP_CDG_SYS" VARCHAR2(80 BYTE),
          "ADMSN_TYP_CDG_SYS_VER" VARCHAR2(80 BYTE),
          "ADMSN_STUS_CD" VARCHAR2(80 BYTE),
          "ADMSN_STUS_CDG_SYS" VARCHAR2(80 BYTE),
          "ADMSN_STUS_CDG_SYS_VER" VARCHAR2(80 BYTE),
          "PT_DC_STUS_CD" VARCHAR2(80 BYTE),
          "PT_DC_STUS_CDG_SYS" VARCHAR2(80 BYTE),
          "PT_DC_STUS_CDG_SYS_VER" VARCHAR2(80 BYTE),
          "PT_MD_CD" VARCHAR2(80 BYTE),
          "PT_MD_CDG_SYS" VARCHAR2(80 BYTE),
          "PT_MD_CDG_SYS_VER" VARCHAR2(80 BYTE),
          "CNFDLTY_CD" VARCHAR2(80 BYTE),
```

"CNFDLTY_CDG_SYS" VARCHAR2(80 BYTE),
"CNFDLTY_CDG_SYS_VER" VARCHAR2(80 BYTE),
"DX_RELTD_GRP_CD" VARCHAR2(80 BYTE),
"DX_RELTD_GRP_CDG_SYS" VARCHAR2(80 BYTE),
"DX_RELTD_GRP_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SRC_OF_ADMSN_CD" VARCHAR2(80 BYTE),
"SRC_OF_ADMSN_CDG_SYS" VARCHAR2(80 BYTE),
"SRC_OF_ADMSN_CDG_SYS_VER" VARCHAR2(80 BYTE),
"MD_OF_ARRV_CD" VARCHAR2(80 BYTE),
"MD_OF_ARRV_CDG_SYS" VARCHAR2(80 BYTE),
"MD_OF_ARRV_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_NBR" VARCHAR2(80 BYTE),
"STRT_DT" DATE,
"END_DT" DATE,
"ESTD_LEN_OF_STY" NUMBER(10,0),
"ESTD_LEN_OF_STY_UOM_CD" VARCHAR2(80 BYTE),

[0203] "ESTD_LEN_OF_STY_UOM_CDG_SYS" VARCHAR2(80 BYTE),
"ESTD_LEN_OF_STY_UOM_CD_SY_VR" VARCHAR2(80 BYTE),
"ACTL_LEN_OF_STY" NUMBER(22,7),
"ACTL_LEN_OF_STY_UOM_CD" VARCHAR2(80 BYTE),
"ACTL_LEN_OF_STY_UOM_CDG_SYS" VARCHAR2(80 BYTE),
"ACTL_LEN_OF_STY_UOM_CD_SY_VR" VARCHAR2(80 BYTE),
"LV_OF_ABSNC" NUMBER(22,7),
"LV_OF_ABSNC_UOM_CD" VARCHAR2(80 BYTE),
"LV_OF_ABSNC_UOM_CDG_SYS" VARCHAR2(80 BYTE),
"LV_OF_ABSNC_UOM_CDG_SYS_VER" VARCHAR2(80 BYTE),
"READM_FLG" CHAR(1 BYTE),
"PREADMT_NBR" VARCHAR2(80 BYTE),
"NT" VARCHAR2(4000 BYTE),
"RCRRNG_VST_FLG" CHAR(1 BYTE),
"EBC_INT_ID" VARCHAR2(400 BYTE),
"EBC_DAT_SRC_NUM_ID" NUMBER(10,0),
"EBC_SRC_CHANGED_ON_DT" DATE,

"ENC_ARRV_DT" DATE,
"PT_ACCT_INT_ID" VARCHAR2(400 BYTE),
"PT_ACCT_DAT_SRC_NUM_ID" NUMBER(10,0),
"PT_ACCT_SRC_CHANGED_ON_DT" DATE,
"ENC_SCHD_DT" DATE,
"ENC_OCCURRED_DT" DATE,
"ENC_CANCELLED_DT" DATE,
"ENC_RSCHD_D_DT" DATE,
"CHECKSUM" VARCHAR2(2000 BYTE),
"CREATED_BY_USER_ID" VARCHAR2(400 BYTE),
"CREATED_BY_USER_DS_NUM_ID" NUMBER(10,0),
"CHANGED_BY_USER_ID" VARCHAR2(400 BYTE),
"CHANGED_BY_USER_DS_NUM_ID" NUMBER(10,0),
"SRC_CREATED_ON_DT" DATE,
"SRC_CHANGED_ON_DT" DATE NOT NULL ENABLE,
"INCORRECT_FLG" CHAR(1 BYTE),

[0204] "INSERT_DT" DATE,
"REQUEST_ID" NUMBER(10,0),
"TENANT_ID" VARCHAR2(80 BYTE),
"DELETE_FLG" CHAR(1 BYTE),
"SRC_FILE_ID" NUMBER(10,0),
"ATTRIBUTE1" VARCHAR2(200 BYTE),
"ATTRIBUTE2" VARCHAR2(200 BYTE),
"ATTRIBUTE3" VARCHAR2(200 BYTE),
"ATTRIBUTE4" VARCHAR2(200 BYTE),
"ATTRIBUTE5" VARCHAR2(200 BYTE),
"ATTRIBUTE6" VARCHAR2(200 BYTE),
"ATTRIBUTE7" VARCHAR2(200 BYTE),
"ATTRIBUTE8" VARCHAR2(200 BYTE),
"ATTRIBUTE9" VARCHAR2(200 BYTE),
"ATTRIBUTE10" VARCHAR2(200 BYTE),
"ATTRIBUTE11" VARCHAR2(200 BYTE),
"ATTRIBUTE12" VARCHAR2(200 BYTE),

"ATTRIBUTE13" VARCHAR2(200 BYTE),
"ATTRIBUTE14" VARCHAR2(200 BYTE),
"ATTRIBUTE15" VARCHAR2(200 BYTE),
"ATTRIBUTE16" NUMBER(22,7),
"ATTRIBUTE17" NUMBER(22,7),
"ATTRIBUTE18" NUMBER(22,7),
"ATTRIBUTE19" DATE,
"ATTRIBUTE20" DATE,
"RECRD_SET_ID" NUMBER(10,0),
"ATTRIBUTE1_CD" VARCHAR2(80 BYTE),
"ATTRIBUTE1_CDG_SYS" VARCHAR2(80 BYTE),
"ATTRIBUTE1_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ATTRIBUTE2_CD" VARCHAR2(80 BYTE),
"ATTRIBUTE2_CDG_SYS" VARCHAR2(80 BYTE),
"ATTRIBUTE2_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ATTRIBUTE3_CD" VARCHAR2(80 BYTE),

[0205]

"ATTRIBUTE3_CDG_SYS" VARCHAR2(80 BYTE),
"ATTRIBUTE3_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ATTRIBUTE4_CD" VARCHAR2(80 BYTE),
"ATTRIBUTE4_CDG_SYS" VARCHAR2(80 BYTE),
"ATTRIBUTE4_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ATTRIBUTE5_CD" VARCHAR2(80 BYTE),
"ATTRIBUTE5_CDG_SYS" VARCHAR2(80 BYTE),
"ATTRIBUTE5_CDG_SYS_VER" VARCHAR2(80 BYTE),
"CORRECTION_FLG" CHAR(1 BYTE),
"ENC_RSN_NT1" VARCHAR2(4000 BYTE),
"ENC_RSN_NT2" VARCHAR2(4000 BYTE),
"ENC_RSN_NT3" VARCHAR2(4000 BYTE),
"ENC_RSN_SUBTYP1_CD" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP1_CDG_SYS" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP1_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP2_CD" VARCHAR2(80 BYTE),
"ENC RSN SUBTYP2 CDG SYS" VARCHAR2(80 BYTE),

"ENC_RSN_SUBTYP2_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP3_CD" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP3_CDG_SYS" VARCHAR2(80 BYTE),
"ENC_RSN_SUBTYP3_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_RSN1_CD" VARCHAR2(80 BYTE),
"ENC_RSN1_CDG_SYS" VARCHAR2(80 BYTE),
"ENC_RSN1_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_RSN2_CD" VARCHAR2(80 BYTE),
"ENC_RSN2_CDG_SYS" VARCHAR2(80 BYTE),
"ENC_RSN2_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENC_RSN3_CD" VARCHAR2(80 BYTE),
"ENC_RSN3_CDG_SYS" VARCHAR2(80 BYTE),
"ENC_RSN3_CDG_SYS_VER" VARCHAR2(80 BYTE),
"ENTERPRISE_ID" NUMBER,
"SPECL_CRTSY1_CD" VARCHAR2(80 BYTE),
"SPECL_CRTSY1_CDG_SYS" VARCHAR2(80 BYTE),

[0206] "SPECL_CRTSY1_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECL_CRTSY2_CD" VARCHAR2(80 BYTE),
"SPECL_CRTSY2_CDG_SYS" VARCHAR2(80 BYTE),
"SPECL_CRTSY2_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECL_CRTSY3_CD" VARCHAR2(80 BYTE),
"SPECL_CRTSY3_CDG_SYS" VARCHAR2(80 BYTE),
"SPECL_CRTSY3_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECLARNGMNT1_CD" VARCHAR2(80 BYTE),
"SPECLARNGMNT1_CDG_SYS" VARCHAR2(80 BYTE),
"SPECLARNGMNT1_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECLARNGMNT2_CD" VARCHAR2(80 BYTE),
"SPECLARNGMNT2_CDG_SYS" VARCHAR2(80 BYTE),
"SPECLARNGMNT2_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECLARNGMNT3_CD" VARCHAR2(80 BYTE),
"SPECLARNGMNT3_CDG_SYS" VARCHAR2(80 BYTE),
"SPECLARNGMNT3_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECLARNGMNT4_CD" VARCHAR2(80 BYTE),

"SPECLARNGMNT4_CDG_SYS" VARCHAR2(80 BYTE),
"SPECLARNGMNT4_CDG_SYS_VER" VARCHAR2(80 BYTE),
"SPECLARNGMNT5_CD" VARCHAR2(80 BYTE),
"SPECLARNGMNT5_CDG_SYS" VARCHAR2(80 BYTE),
"SPECLARNGMNT5_CDG_SYS_VER" VARCHAR2(80 BYTE),
"REFAL_RSN_CD" VARCHAR2(80 BYTE),
"REFAL_RSN_CDG_SYS" VARCHAR2(80 BYTE),
"REFAL_RSN_CDG_SYS_VER" VARCHAR2(80 BYTE),
"PLND_DC_DT" DATE,

[0207] "HMN_SUBJ_INT_ID" VARCHAR2(400 BYTE),
"HMN_SUBJ_DAT_SRC_NUM_ID" NUMBER(10,0),
"HMN_SUBJ_SRC_CHANGED_ON_DT" DATE,
"SRC_SURGT_KY" VARCHAR2(400 BYTE),
"SRC_BIZ_KY" VARCHAR2(400 BYTE),
"MSTR_KY_ROOT_ID" VARCHAR2(400 BYTE),
"MSTR_KY_EXTN_TXT" VARCHAR2(400 BYTE),
"RSPNSBL_SVCPRV_INT_ID" VARCHAR2(400 BYTE),
"RSPNSBL_SVCPRV_DAT_SRC_NUM_ID" NUMBER(10,0),
"RSPNSBL_SVCPRV_SRC_CH_ON_DT" DATE)

[0208] 目标表-父:

[0209]

```
CREATE TABLE "OHF711_HDM"."HDM_ACC_ENT"
 ( "ACC_ENT_ID" NUMBER(38,0) NOT NULL ENABLE,
  "CHRTFLD_COMB_ID" NUMBER(38,0) NOT NULL ENABLE,
  "FSCL_PD_ID" NUMBER(38,0) NOT NULL ENABLE,
  "GOV_ORG_ID" NUMBER(38,0),
  "ACC_ENT_LN_SUBTYP_ID" NUMBER(38,0) NOT NULL ENABLE,
  "ACC_ENT_LN_TYP_ID" NUMBER(38,0),
  "ACC_ENT_STCD_ID" NUMBER(38,0),
  "CURR_CD_ID" NUMBER(38,0),
  "ACC_ENT_ADDED_DT" DATE,
  "ACCTG_DT" DATE,
  "ACC_ENT_AMT" NUMBER(22,7),
  "STAT_AMT" NUMBER(22,7),
  "STAT_AMT_UOMCD_ID" NUMBER(38,0),
  "FRGHT_AMT" NUMBER(22,7),
  "DSCNT_AMT" NUMBER(22,7),
  "MISC_AMT" NUMBER(22,7),
  "EXCS_RCVRY_AMT" NUMBER(22,7),
  "EXCS_NON_RCVRD_AMT" NUMBER(22,7),
  "ACC_ENT_LN_QTY" NUMBER(22,7),
  "ACC_ENT_LN_QTY_UOMCD_ID" NUMBER(38,0),
  "ORIG_ACC_ENT_LN_SUBTYP_ID" NUMBER(38,0),
  "ORIG_ACC_ENT_LN_TYP_ID" NUMBER(38,0),
  "ORIG_ACC_ENT_STCD_ID" NUMBER(38,0),
  "ORIG_CURR_CD_ID" NUMBER(38,0),
  "ORIG_STAT_AMT_UOMCD_ID" NUMBER(38,0),
  "ORIG_ACC_ENT_LN_QTY_UOMCD_ID" NUMBER(38,0),
  "CREATED_BY_ID" NUMBER(38,0),
  "CHANGED_BY_ID" NUMBER(38,0),
  "CREATED_ON_DT" DATE,
  "CHANGED_ON_DT" DATE,
  "INSERT_DT" DATE NOT NULL ENABLE,
```

"UPDATE_DT" DATE NOT NULL ENABLE,
"REQUEST_ID" NUMBER(10,0),
"SRC_EFF_FROM_DT" DATE,
"SRC_EFF_TO_DT" DATE,
"EFFECTIVE_FROM_DT" DATE NOT NULL ENABLE,
"EFFECTIVE_TO_DT" DATE NOT NULL ENABLE,
"CURRENT_FLG" CHAR(1 BYTE),
"INCORRECT_FLG" CHAR(1 BYTE),
"DATASOURCE_NUM_ID" NUMBER(10,0) NOT NULL ENABLE,
"INTEGRATION_ID" VARCHAR2(400 BYTE) NOT NULL ENABLE,
"TENANT_ID" VARCHAR2(80 BYTE),
"DELETE_FLG" CHAR(1 BYTE),
"UDA1" VARCHAR2(200 BYTE),
"UDA2" VARCHAR2(200 BYTE),
"UDA3" VARCHAR2(200 BYTE),
"UDA4" VARCHAR2(200 BYTE),

[0210]

"UDA5" VARCHAR2(200 BYTE),
"UDA6" VARCHAR2(200 BYTE),
"UDA7" VARCHAR2(200 BYTE),
"UDA8" VARCHAR2(200 BYTE),
"UDA9" VARCHAR2(200 BYTE),
"UDA10" VARCHAR2(200 BYTE),
"UDA11" VARCHAR2(200 BYTE),
"UDA12" VARCHAR2(200 BYTE),
"UDA13" VARCHAR2(200 BYTE),
"UDA14" VARCHAR2(200 BYTE),
"UDA15" VARCHAR2(200 BYTE),
"UDA16" NUMBER(22,7),
"UDA17" NUMBER(22,7),
"UDA18" NUMBER(22,7),
"UDA19" DATE,
"UDA20" DATE,
"ACC_ENT_VN" NUMBER(10,0),

"CHRTFLD_COMB_VN" NUMBER(10,0) NOT NULL ENABLE,
"FSCL_PD_VN" NUMBER(10,0) NOT NULL ENABLE,
"GOV_ORG_VN" NUMBER(10,0),
"ORGNTNG_SRC_APP_NM" VARCHAR2(80 BYTE),
"ORIG_STAT_AMT_UOMCD_VN" NUMBER(10,0),
"ACC_ENT_LN_TYP_VN" NUMBER(10,0),
"ORIG_ACC_ENT_LN_QTY_UOMCD_VN" NUMBER(10,0),
"ORIG_CURR_CD_VN" NUMBER(10,0),
"ACC_ENT_LN_SUBTYP_VN" NUMBER(10,0) NOT NULL ENABLE,
"ORIG_ACC_ENT_LN_TYP_VN" NUMBER(10,0),
"ACC_ENT_LN_QTY_UOMCD_VN" NUMBER(10,0),
"STAT_AMT_UOMCD_VN" NUMBER(10,0),
"CURR_CD_VN" NUMBER(10,0),
"ORIG_ACC_ENT_LN_SUBTYP_VN" NUMBER(10,0),
"ORIG_ACC_ENT_STCD_VN" NUMBER(10,0),
"ACC_ENT_STCD_VN" NUMBER(10,0),

[0211] "RECRD_SET_ID" NUMBER(10,0),
"ENTERPRISE_ID" NUMBER,
"CORRECTION_FLG" CHAR(1 BYTE),
"UDA1_CD_ID" NUMBER(38,0),
"UDA1_CD_VN" NUMBER(10,0),
"UDA2_CD_ID" NUMBER(38,0),
"UDA2_CD_VN" NUMBER(10,0),
"UDA3_CD_ID" NUMBER(38,0),
"UDA3_CD_VN" NUMBER(10,0),
"UDA4_CD_ID" NUMBER(38,0),
"UDA4_CD_VN" NUMBER(10,0),
"UDA5_CD_ID" NUMBER(38,0),
"UDA5_CD_VN" NUMBER(10,0),
"ORIG_UDA1_CD_ID" NUMBER(38,0),
"ORIG_UDA1_CD_VN" NUMBER(10,0),
"ORIG_UDA2_CD_ID" NUMBER(38,0),
"ORIG_UDA2_CD_VN" NUMBER(10,0),

\

"ORIG_UDA3_CD_ID" NUMBER(38,0),

"ORIG_UDA3_CD_VN" NUMBER(10,0),

"ORIG_UDA4_CD_ID" NUMBER(38,0),

"ORIG_UDA4_CD_VN" NUMBER(10,0),

"ORIG_UDA5_CD_ID" NUMBER(38,0),

[0212] "ORIG_UDA5_CD_VN" NUMBER(10,0),

"SRC_SURGT_KY" VARCHAR2(400 BYTE),

"SRC_BIZ_KY" VARCHAR2(400 BYTE),

"MSTR_KY_ROOT_ID" VARCHAR2(400 BYTE),

"MSTR_KY_EXTN_TXT" VARCHAR2(400 BYTE),

CONSTRAINT "HDM_ACC_ENT_PK" PRIMARY KEY ("ACC_ENT_ID", "ACC_ENT_VN"))

[0213] 目标表-子:

[0214]

```
CREATE TABLE "TARGET"."HDM_INVT_ACCTG_LN"
    ( "ACC_ENT_ID" NUMBER(38,0) NOT NULL ENABLE,
     "INVT_TRANS_ID" NUMBER(38,0) NOT NULL ENABLE,
     "CREATED_BY_ID" NUMBER(38,0),
     "CHANGED_BY_ID" NUMBER(38,0),
     "CREATED_ON_DT" DATE,
     "CHANGED_ON_DT" DATE,
     "INSERT_DT" DATE NOT NULL ENABLE,
     "UPDATE_DT" DATE NOT NULL ENABLE,
     "REQUEST_ID" NUMBER(10,0),
     "SRC_EFF_FROM_DT" DATE,
     "SRC_EFF_TO_DT" DATE,
     "EFFECTIVE_FROM_DT" DATE NOT NULL ENABLE,
     "EFFECTIVE_TO_DT" DATE NOT NULL ENABLE,
     "CURRENT_FLG" CHAR(1 BYTE),
     "INCORRECT_FLG" CHAR(1 BYTE),
     "DATASOURCE_NUM_ID" NUMBER(10,0) NOT NULL ENABLE,
     "INTEGRATION_ID" VARCHAR2(400 BYTE) NOT NULL ENABLE,
     "TENANT_ID" VARCHAR2(80 BYTE),
     "DELETE_FLG" CHAR(1 BYTE),
     "ACC_ENT_VN" NUMBER(10,0) NOT NULL ENABLE,
     "INVT_TRANS_VN" NUMBER(10,0) NOT NULL ENABLE,
     "RECRD_SET_ID" NUMBER(10,0),
     "ENTERPRISE_ID" NUMBER,
     "CORRECTION_FLG" CHAR(1 BYTE),
     "UDA1" VARCHAR2(200 BYTE),
     "UDA2" VARCHAR2(200 BYTE),
     "SRC_SURGT_KY" VARCHAR2(400 BYTE),
     "SRC_BIZ_KY" VARCHAR2(400 BYTE),
     "MSTR_KY_ROOT_ID" VARCHAR2(400 BYTE),
     "MSTR_KY_EXTN_TXT" VARCHAR2(400 BYTE),
      CONSTRAINT "HDM_INVT_ACCTG_LN_PK" PRIMARY KEY
```

("ACC_ENT_ID", "ACC_ENT_VN"))

[0215]

```
ALTER TABLE "TARGET"."HDM_INVT_ACCTG_LN" ADD CONSTRAINT
"HDM_INVT_ACCTG_LN_HDM_ACC_FK" FOREIGN KEY ("HDM_ACC_ENT_PK")
REFERENCES "TARGET"."HDM_ACC" ("ACC_ENT_ID",
"ACC_ENT_VN") ENABLE;
```

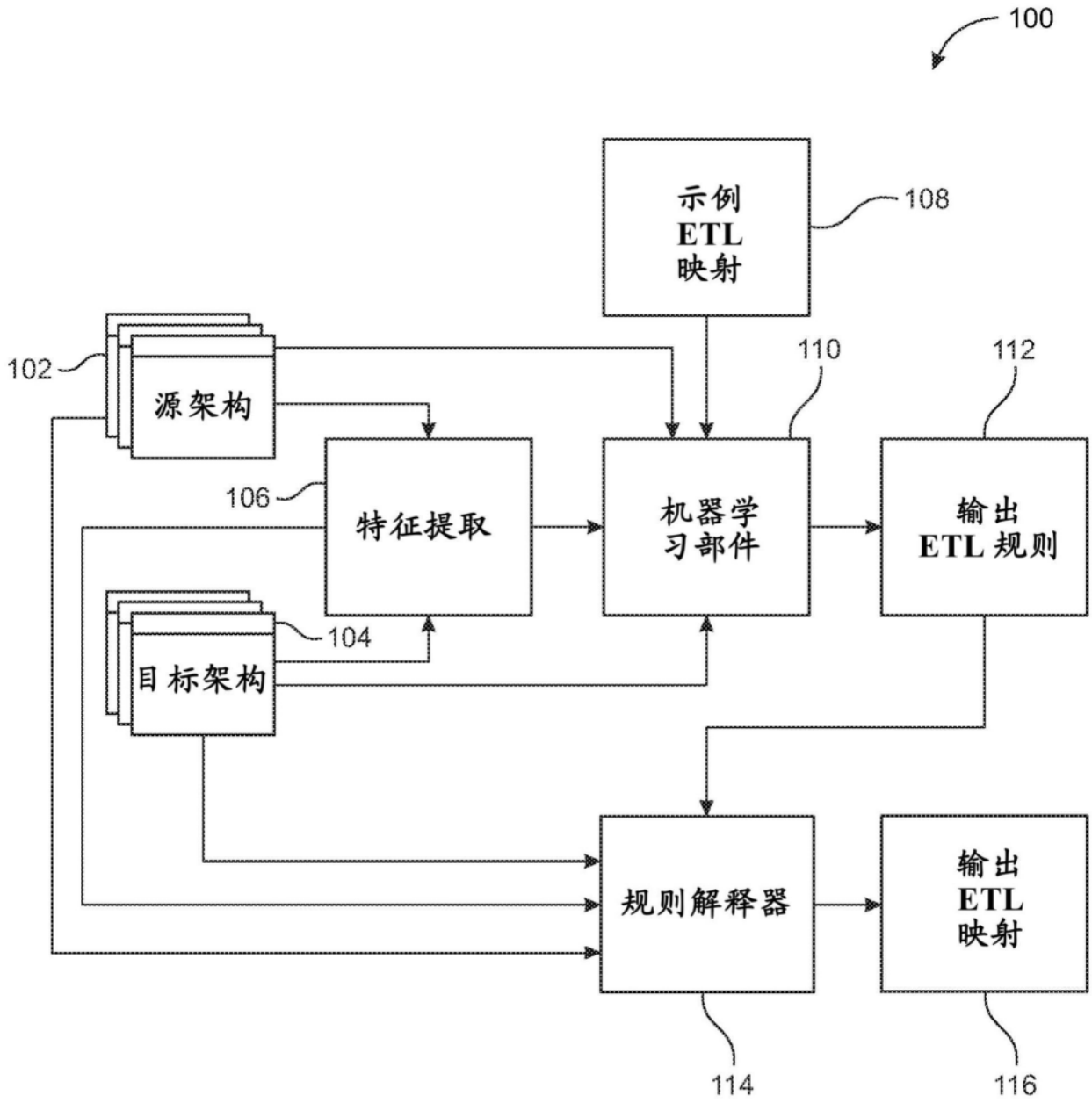

100
示例
ETL
映射
108
102
源架构
110
112
106
特征提取
机器学
习部件
输出
ETL 规则
104
目标架构
规则解释器
输出
ETL
映射
114
116

图1

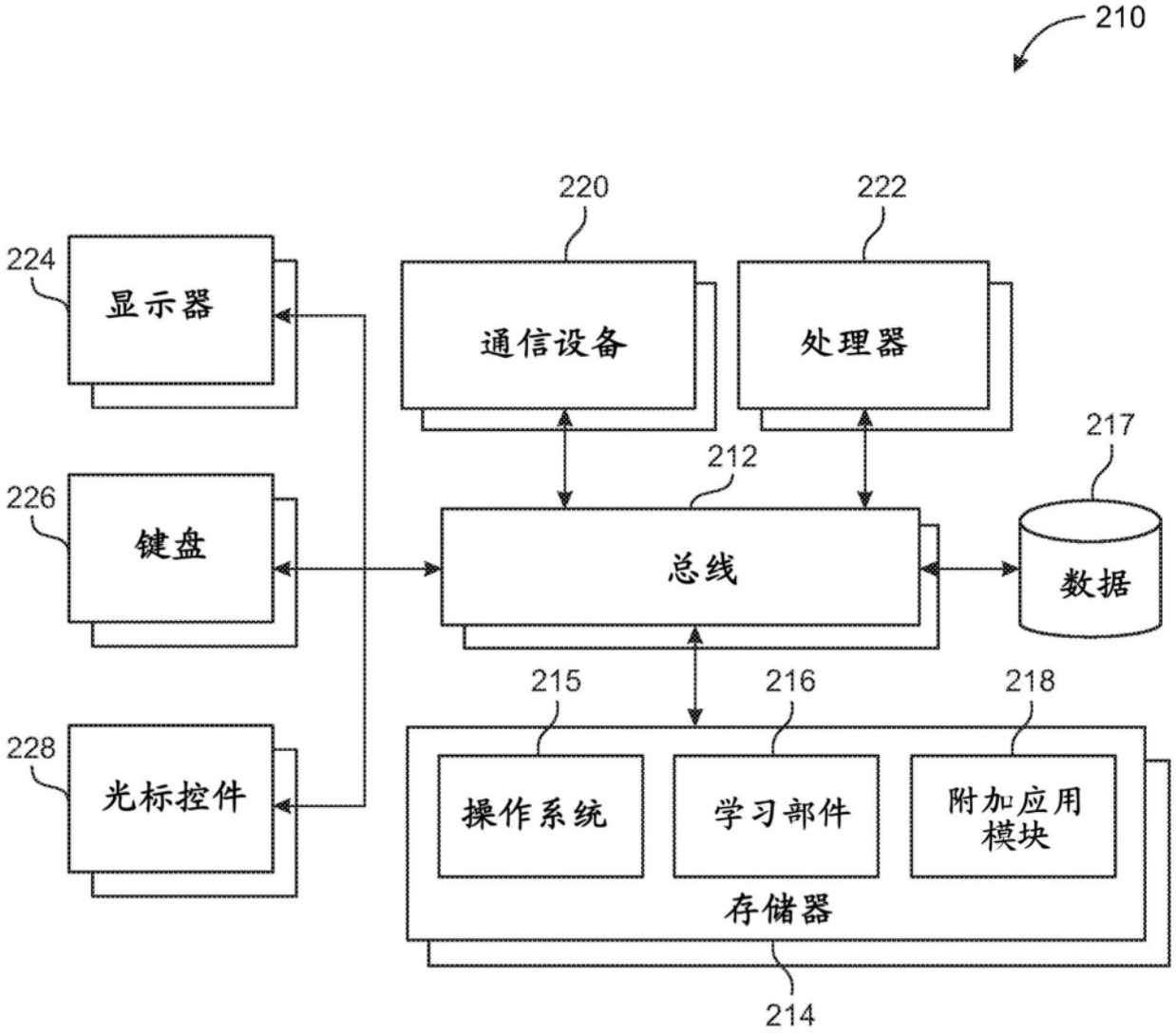

210
220
222
224
显示器
通信设备
处理器
217
212
226
键盘
总线
数据
215
216
218
228
光标控件
操作系统
学习部件
附加应用模块
存储器
214

图2

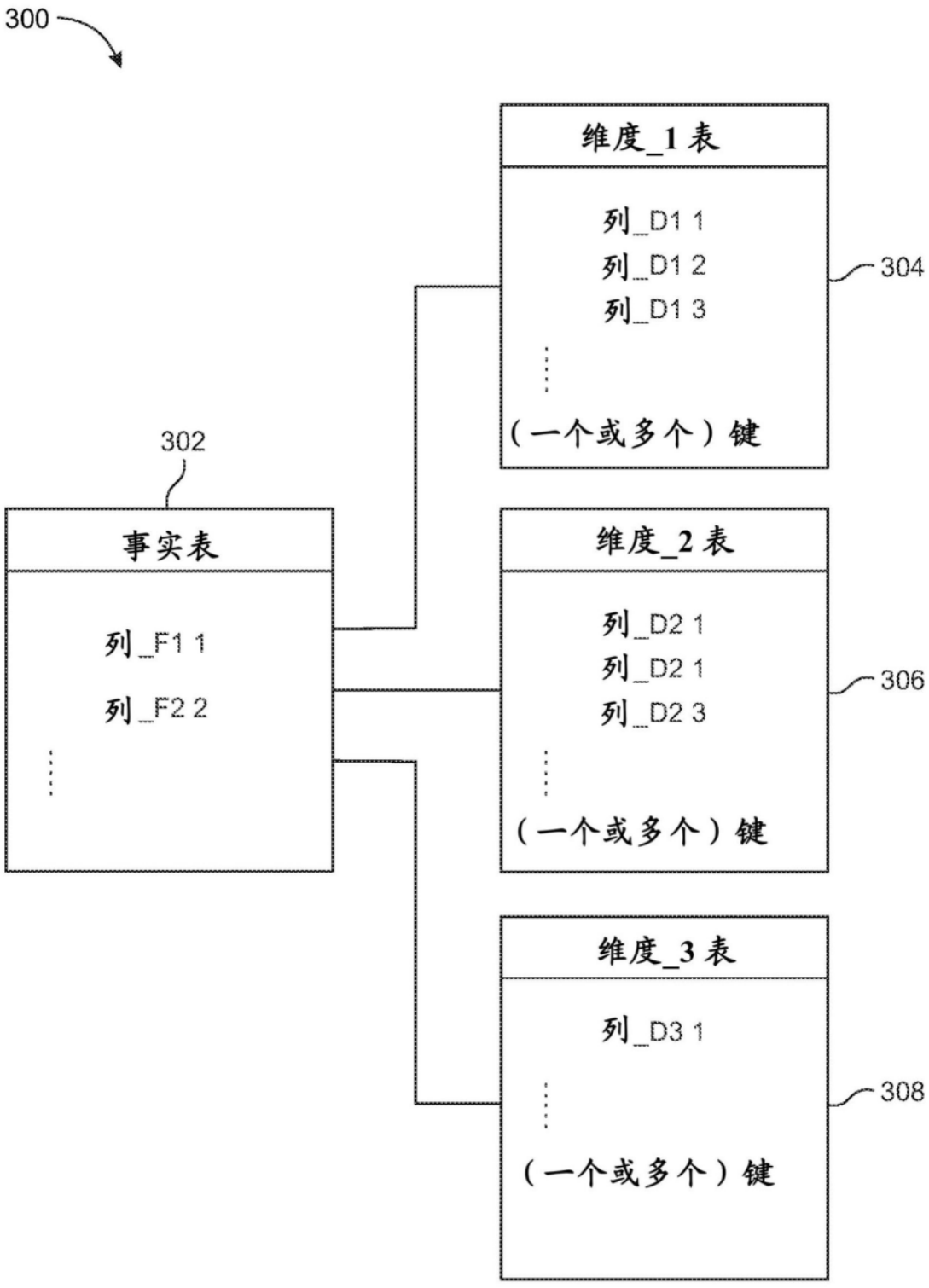

300
维度_1 表
列_D1 1
列_D1 2
列_D1 3
（一个或多个）键
304
302
事实表
列_F1 1
列_F2 2
维度_2 表
列_D2 1
列_D2 1
列_D2 3
（一个或多个）键
306
维度_3 表
列_D3 1
（一个或多个）键
308

图3


400
<Transformation>
<Name>HDI_ACC_ENT_HDM_ACC_ENT</Name>
<FolderName>DI_HDWF_INITIAL_LOAD/Load_DI_HDWF_INITIAL_LOAD_HDI_ACC_ENT</FolderName>
<PackageList>HDI_ACC_ENT_HDM_ACC_ENT<PackageList>
<Datasets>
<Datasets>
<Source>
<Name>HDI_ACC_ENT_</Name>
<Filter>1=1 AND ((HDI_ACC_ENT. INSERT_DT between to_date ('#OHADI. VAR_ACC_ENT_LAST_EXTRACT_DATETEST', 'MM/DD/YYYY HH24:MI:SS') and to_date ('#OHADI. VAR_ACC_ENT_EXTRACT_TO_DATETEST', 'MM/DD/YYYY HH24:MI:SS'))) </Filter>
<FilterExecutionLocation>SOURCE</FilterExecutionLocation>
<JoinType>INNER</JoinType>
<Lookups>
<Lookup>
<LookupDataStore>HDM_CHRTFLD_COMB</LookupDataStore>
<Alias>CHRTFLD_COMB_ID_CURRENT</Alias>
<Join>HDI_ACC_ENT. CHRTFLD_COMB_INT_ID=CHRTFLD_COMB_ID_CURRENT. INTEGRATION_ID AND HDI_ACC_ENT. CHRTFLD_COMB_DAT_SRC_NUM_ID=CHRTFLD_COMB_ID_CURRENT. DATASOURCE_NUM_ID AND CHRTFLD_COMB_ID CURRENT. CURRENT_FLG='Y'
</Join>
<Journalized>false</Journalized>
<SubSelect>false</SubSelect>
</Lookup>
<Lookup>
<LookupDataStore>HDM_FSCL_PD</LookupDataStore>
<Alias>FSCL_FD_ID_CURRENT</Alias>
<Join>HDI_ACC_ENT. FSCL_FD_INT_ID=FSCL_FD_ID_CURRENT. INTEFRATION_ID AND HDI_ACC_ENT. FSCL_FD_DAT_SRC_NUM_ID=FSCL_FD_ID_CURRENT. DATASOURCE_NUM_ID AND FSCL_FD_ID_CURRENT. CURRENT_FLG='Y'</Join>
<Journalized>false</Journalized>
<SubSelect>false</SubSelect>
</Lookup>
<Lookup>
<LookupDataStore>HDM_GOV_ORG</LookupDataStore>
<Alias>GOV_ORG_ID_CURRENT</Alias>
<Join>HDI_ACC_ENT.GOV ORG_INT_ID=GOV_ORG_ID_CURRENT.INTEGRATION_ID AND HDI_ACC_ENT. GOV_ORG_DAT_SRC_NUM_ID=GOV_ORG_ID_CURRENT. DATASOURCE_NUM_ID AND GOV_ORG_ID_CURRENT. CURRENT_FLG='Y'</Join>
<Journalized>false</Journalized>
<SubSelect>false</SubSelect>
</Lookup>

图4

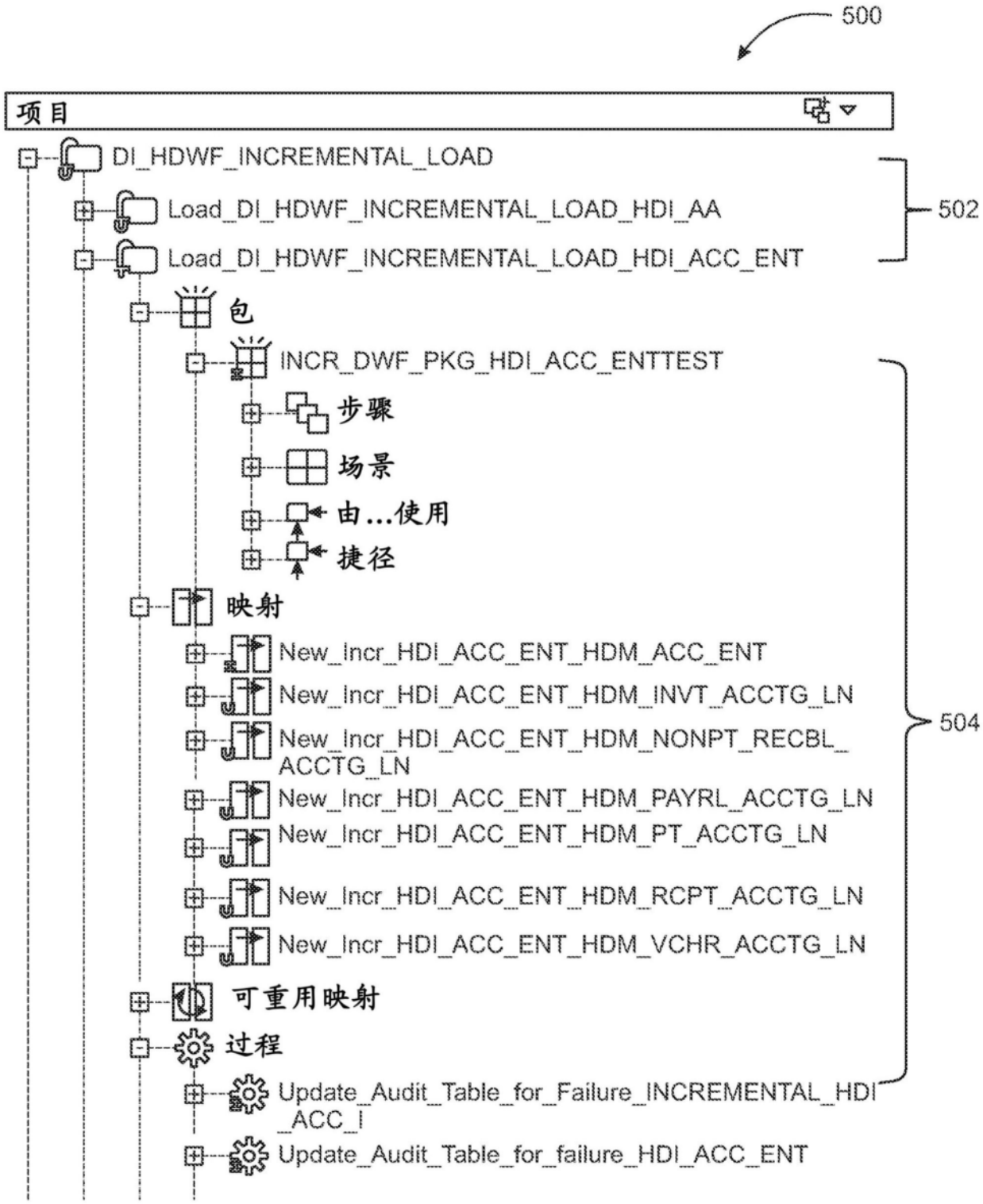

500
项目
DI_HDWF_INCREMENTAL_LOAD
Load_DI_HDWF_INCREMENTAL_LOAD_HDI_AA
Load_DI_HDWF_INCREMENTAL_LOAD_HDI_ACC_ENT
502
包
INCR_DWF_PKG_HDI_ACC_ENTTEST
步骤
场景
由...使用
捷径
映射
New_Incr_HDI_ACC_ENT_HDM_ACC_ENT
New_Incr_HDI_ACC_ENT_HDM_INVT_ACCTG_LN
New_Incr_HDI_ACC_ENT_HDM_NONPT_RECBL_ACCTG_LN
New_Incr_HDI_ACC_ENT_HDM_PAYRL_ACCTG_LN
New_Incr_HDI_ACC_ENT_HDM_PT_ACCTG_LN
New_Incr_HDI_ACC_ENT_HDM_RCPT_ACCTG_LN
New_Incr_HDI_ACC_ENT_HDM_VCHR_ACCTG_LN
可重用映射
过程
Update_Audit_Table_for_Failure_INCREMENTAL_HDI_ACC_I
Update_Audit_Table_for_failure_HDI_ACC_ENT
504

图5

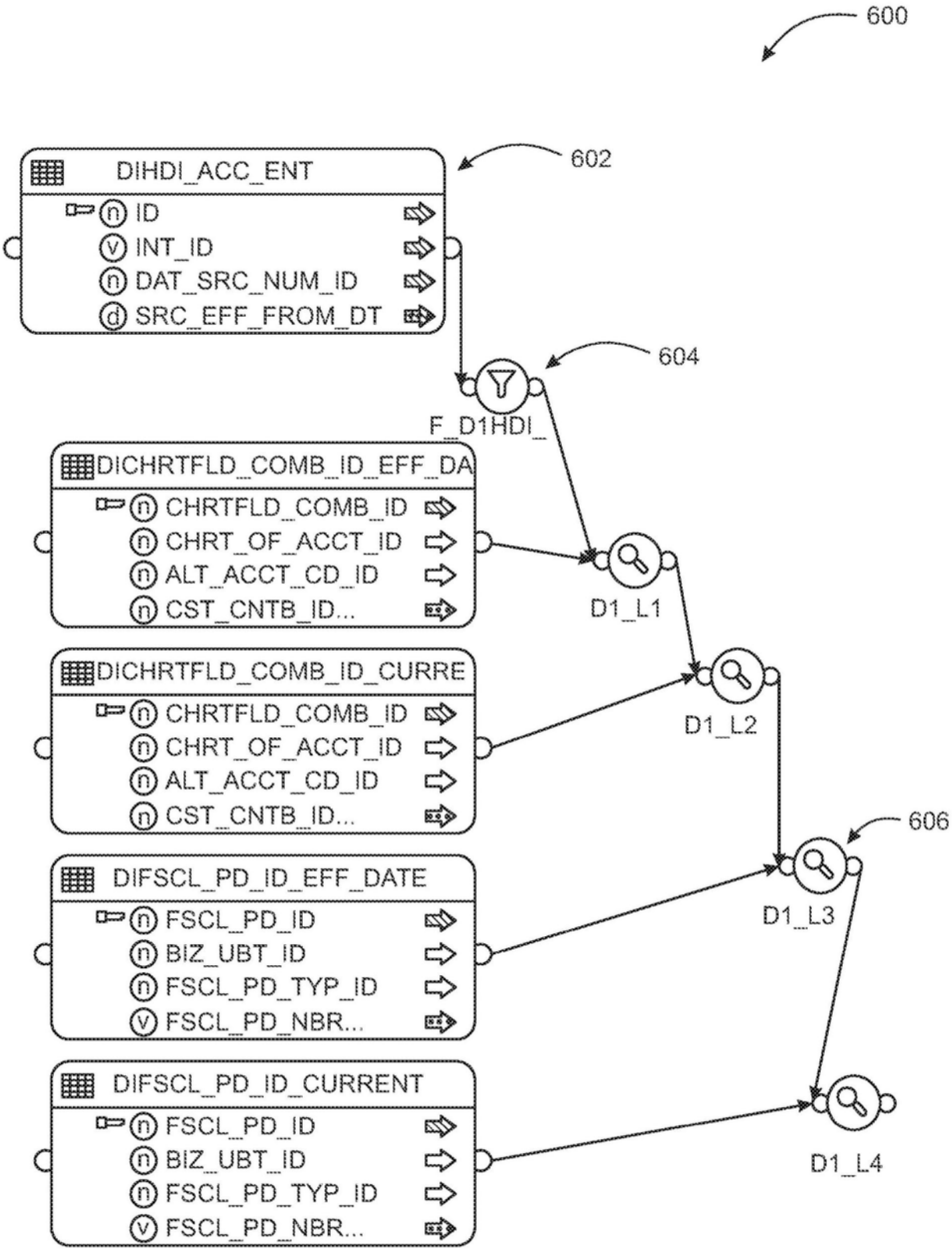

600
602
DIHDI_ACC_ENT
ID
INT_ID
DAT_SRC_NUM_ID
SRC_EFF_FROM_DT
604
F_D1HDI_
DICHRTFLD_COMB_ID_EFF_DA
CHRTFLD_COMB_ID
CHRT_OF_ACCT_ID
ALT_ACCT_CD_ID
CST_CNTB_ID...
D1_L1
DICHRTFLD_COMB_ID_CURRE
CHRTFLD_COMB_ID
CHRT_OF_ACCT_ID
ALT_ACCT_CD_ID
CST_CNTB_ID...
D1_L2
606
DIFSCL_PD_ID_EFF_DATE
FSCL_PD_ID
BIZ_UBT_ID
FSCL_PD_TYP_ID
FSCL_PD_NBR...
D1_L3
DIFSCL_PD_ID_CURRENT
FSCL_PD_ID
BIZ_UBT_ID
FSCL_PD_TYP_ID
FSCL_PD_NBR...
D1_L4

图6

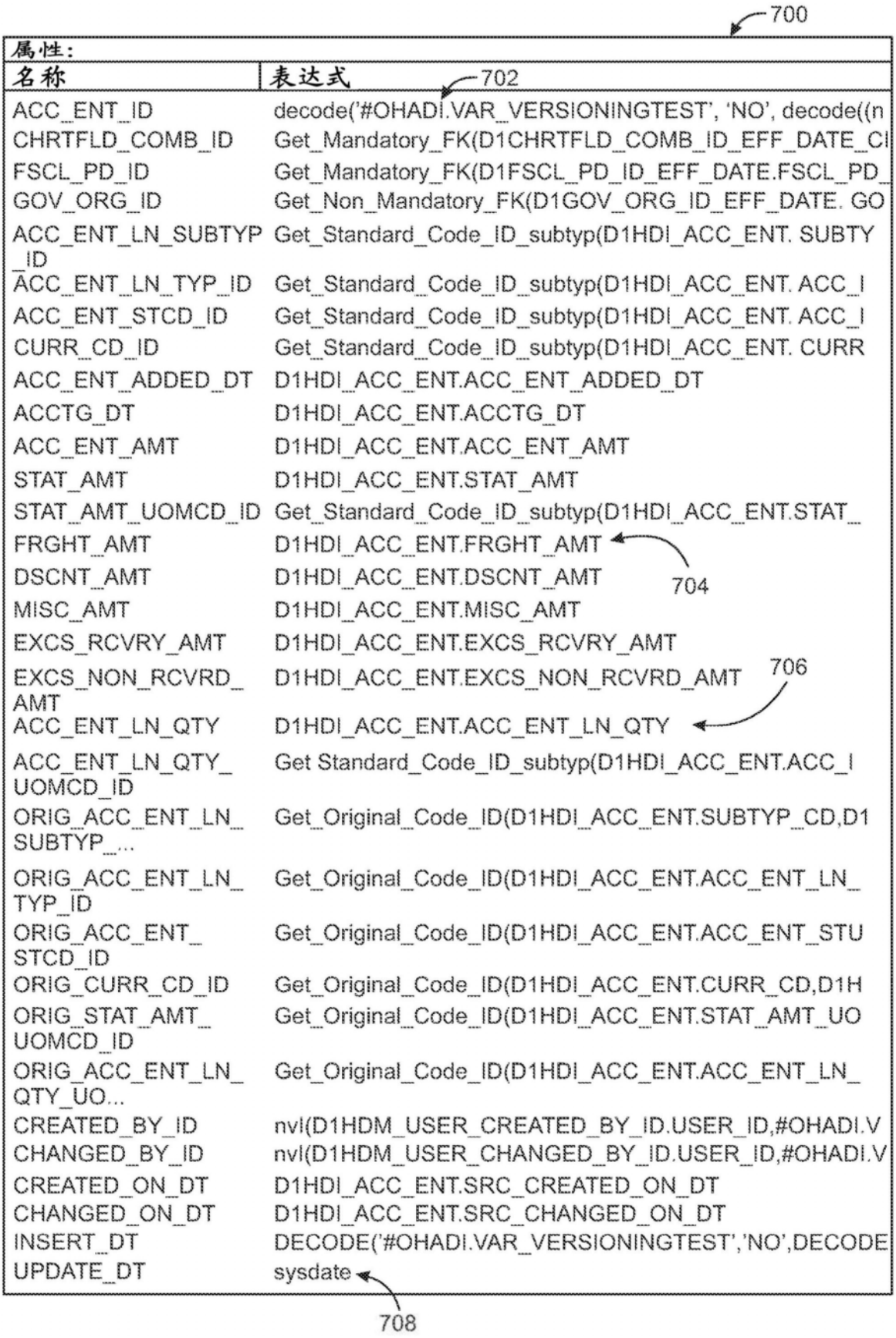

700
属性:
名称
表达式
702
ACC_ENT_ID
decode('#OHADI.VAR_VERSIONINGTEST', 'NO', decode((n
CHRTFLD_COMB_ID
Get_Mandatory_FK(D1CHRTFLD_COMB_ID_EFF_DATE_CI
FSCL_PD_ID
Get_Mandatory_FK(D1FSCL_PD_ID_EFF_DATE.FSCL_PD_
GOV_ORG_ID
Get_Non_Mandatory_FK(D1GOV_ORG_ID_EFF_DATE. GO
ACC_ENT_LN_SUBTYP_ID
Get_Standard_Code_ID_subtyp(D1HDI_ACC_ENT. SUBTY
ACC_ENT_LN_TYP_ID
Get_Standard_Code_ID_subtyp(D1HDI_ACC_ENT. ACC_I
ACC_ENT_STCD_ID
Get_Standard_Code_ID_subtyp(D1HDI_ACC_ENT. ACC_I
CURR_CD_ID
Get_Standard_Code_ID_subtyp(D1HDI_ACC_ENT. CURR
ACC_ENT_ADDED_DT
D1HDI_ACC_ENT.ACC_ENT_ADDED_DT
ACCTG_DT
D1HDI_ACC_ENT.ACCTG_DT
ACC_ENT_AMT
D1HDI_ACC_ENT.ACC_ENT_AMT
STAT_AMT
D1HDI_ACC_ENT.STAT_AMT
STAT_AMT_UOMCD_ID
Get_Standard_Code_ID_subtyp(D1HDI_ACC_ENT.STAT_
FRGHT_AMT
D1HDI_ACC_ENT.FRGHT_AMT
DSCNT_AMT
D1HDI_ACC_ENT.DSCNT_AMT
704
MISC_AMT
D1HDI_ACC_ENT.MISC_AMT
EXCS_RCVRY_AMT
D1HDI_ACC_ENT.EXCS_RCVRY_AMT
EXCS_NON_RCVRD_AMT
D1HDI_ACC_ENT.EXCS_NON_RCVRD_AMT
706
ACC_ENT_LN_QTY
D1HDI_ACC_ENT.ACC_ENT_LN_QTY
ACC_ENT_LN_QTY_UOMCD_ID
Get Standard_Code_ID_subtyp(D1HDI_ACC_ENT.ACC_I
ORIG_ACC_ENT_LN_SUBTYP_...
Get_Original_Code_ID(D1HDI_ACC_ENT.SUBTYP_CD,D1
ORIG_ACC_ENT_LN_TYP_ID
Get_Original_Code_ID(D1HDI_ACC_ENT.ACC_ENT_LN_
ORIG_ACC_ENT_STCD_ID
Get_Original_Code_ID(D1HDI_ACC_ENT.ACC_ENT_STU
ORIG_CURR_CD_ID
Get_Original_Code_ID(D1HDI_ACC_ENT.CURR_CD,D1H
ORIG_STAT_AMT_UOMCD_ID
Get_Original_Code_ID(D1HDI_ACC_ENT.STAT_AMT_UO
ORIG_ACC_ENT_LN_QTY_UO...
Get_Original_Code_ID(D1HDI_ACC_ENT.ACC_ENT_LN_
CREATED_BY_ID
nvl(D1HDM_USER_CREATED_BY_ID.USER_ID,#OHADI.V
CHANGED_BY_ID
nvl(D1HDM_USER_CHANGED_BY_ID.USER_ID,#OHADI.V
CREATED_ON_DT
D1HDI_ACC_ENT.SRC_CREATED_ON_DT
CHANGED_ON_DT
D1HDI_ACC_ENT.SRC_CHANGED_ON_DT
INSERT_DT
DECODE('#OHADI.VAR_VERSIONINGTEST','NO',DECODE
UPDATE_DT
sysdate
708

图7

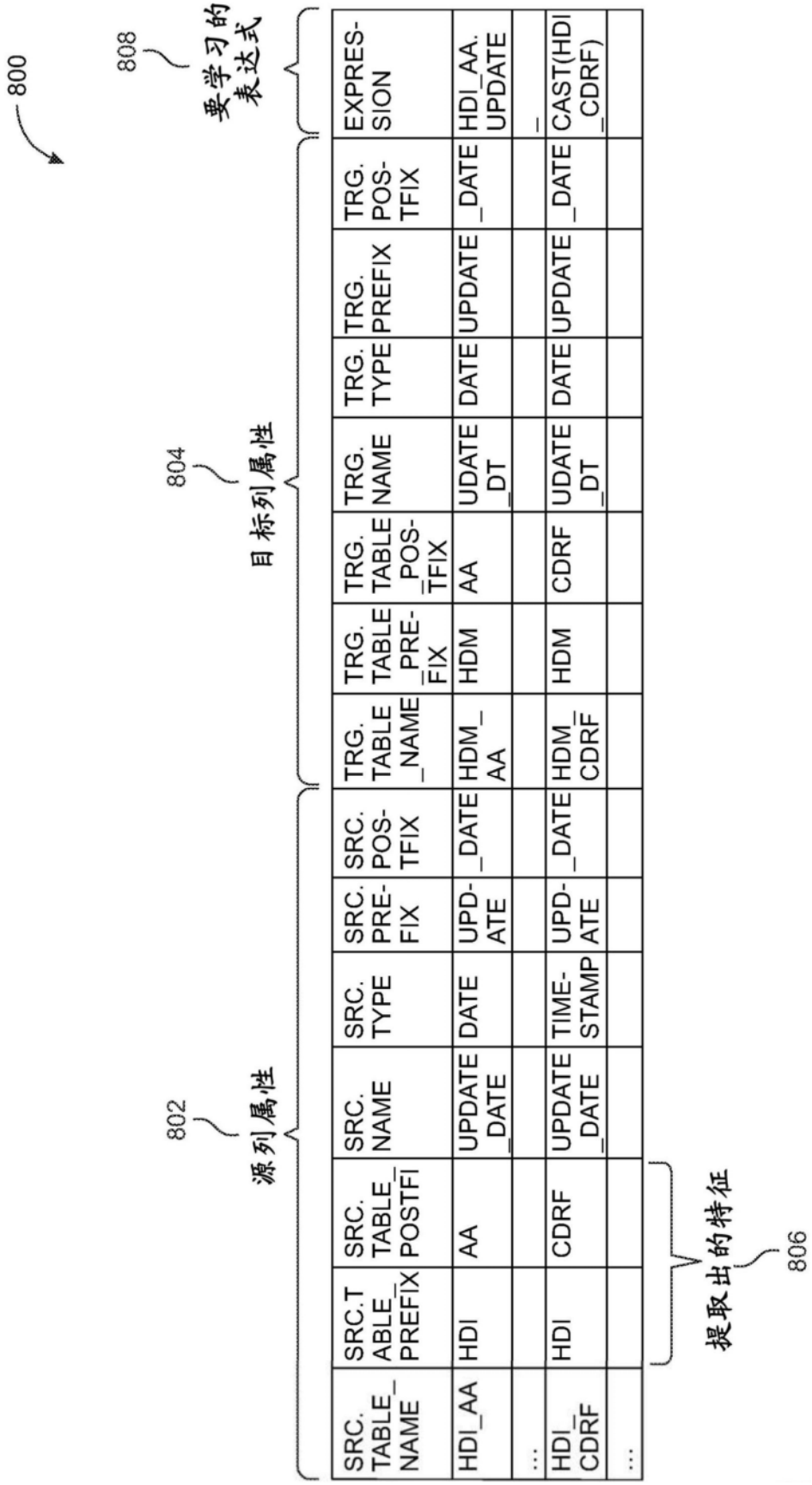

800
808
要学习的表达式
804
目标列属性
802
源列属性
806
提取出的特征
SRC. TABLE_ NAME
SRC.T ABLE_ PREFIX
SRC. TABLE_ POSTFI
SRC. NAME
SRC. TYPE
SRC. PRE- FIX
SRC. POS- TFIX
TRG. TABLE _NAME
TRG. TABLE _PRE- FIX
TRG. TABLE _POS- TFIX
TRG. NAME
TRG. TYPE
TRG. PREFIX
TRG. POS- TFIX
EXPRES- SION
HDI_AA
HDI
AA
UPDATE _DATE
DATE
UPD- ATE
_DATE
HDM_ AA
HDM
AA
UDATE _DT
DATE
UPDATE
_DATE
HDI_AA. UPDATE
...
HDI_ CDRF
HDI
CDRF
UPDATE _DATE
TIME- STAMP
UPD- ATE
_DATE
HDM_ CDRF
HDM
CDRF
UDATE _DT
DATE
UPDATE
_DATE
CAST(HDI _CDRF)
...

图8

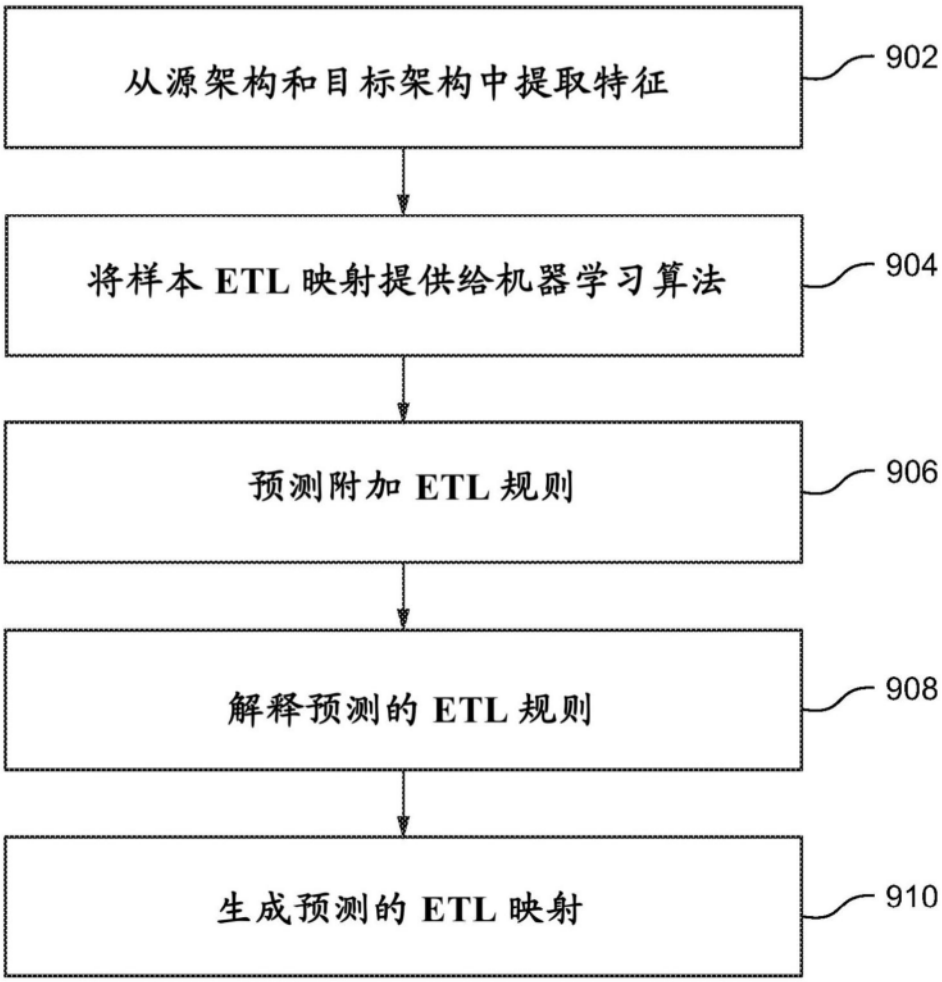

从源架构和目标架构中提取特征
902
将样本 ETL 映射提供给机器学习算法
904
预测附加 ETL 规则
906
解释预测的 ETL 规则
908
生成预测的 ETL 映射
910

图9