第37卷 第17期
Vol.37 No.17

计 算 机 工 程
Computer Engineering

2011年9月
September 2011

·开发研究与设计技术· 文章编号：1000—3428(2011)17—0256—03 文献标识码：A 中图分类号：TP311

面向 ETL 的数据起源追踪系统

戴超凡，王 涛

(国防科学技术大学信息系统与管理学院信息系统工程重点实验室，长沙 410073)

摘 要： 提出一种面向提取-转换-加载(ETL)过程的数据起源追踪系统，讨论实现的关键技术，包括转换分类、元数据设计、转换序列构建、追踪流程设计以及不同转换的追踪方法。系统将追踪所需的元数据设计在包文件结构中，在逆向追踪时抽取元数据进行相关处理，构建各个层次的转换起源信息图，从而实现数据起源的追踪。

关键词： 数据起源；起源管理系统；提取-转换-加载；同步/异步转换

Data Provenance Tracing System for Extraction-Transform-Load

DAI Chao-fan, WANG Tao

(Science and Technology on Information Systems Engineering Laboratory, College of Information System and Management, National University of Defense Technology, Changsha 410073, China)

【Abstract】 This paper proposes a lineage tracing frame for Extraction-Transform-Load(ETL) process, and discusses some key issues in this system, such as classifying the transformations, the designing of metadata, constructing transformation series, tracing process design and the tracing methods for every type of the transformation. The metadata for tracing is injected into the package file, these information are extracted when tracing query were proposed to support data tracing.

【Key words】 data provenance; provenance management system; Extraction-Transform-Load(ETL); synchronous/asynchronous transformation

DOI: 10.3969/j.issn.1000-3428.2011.17.086

1 概述

数据起源[1]追踪技术兴起于20世纪90年代，文献[2]较早对其进行了系统研究。在不同的应用领域，数据志有不同的提法，如Data Lineage[2]、Data Provenance[3]、Derivation[4]等，文献[5]称其为数据起源，有些商用数据库系统中称为数据沿袭。数据起源就是记录数据从产生到消亡的整个生命周期内所发生的变化和经过的处理的信息。

在数据集成过程中，提取-转换-加载(Extraction-Transformation-Loading, ETL)是一种常用手段，ETL过程的实质就是将符合特定规则的数据从异构数据源流向统一的目标数据[6]的过程。这一过程涉及的转换种类繁多、过程复杂，转换中产生的误差和错误往往会通过累积、叠加而逐级放大，因此，ETL过程中的数据质量至关重要，数据起源追踪技术以逆向审计的方式对关键数据或可疑数据进行求本溯源，为数据证据和问题定位提供有效的技术支持。

数据起源追踪的系统可追溯到地理信息系统(Geographic Information System, GIS)数据库开发的LIP系统[7]。现在数据起源追踪系统在国际上应用非常广泛，主要集中在自然科学和计算机科学等领域，追踪的方法主要为注释和逆置。

起源信息的描述与追踪方法密切相关，主要包括虚拟数据语言(VDL)、XML/RDF、关系数据表等方式描述，信息的存储则是与描述方式对应的。对于起源信息的展示，一般都通过结构化查询语言(Structured Query Language, SQL)，或专用查询语言查询以可视化，或文本信息方式展示某项数据的起源。目前这一研究领域国际上较为成熟的系统有Chimera、CMCS、ESSW、VDS、DBNotes、Trio、Panda、myGrid、Taverna、PASOA、Kepler、Karma和Vistrail等。

文献[1]比较系统地研究了数据仓库系统中数据起源追踪的理论和方法；文献[5]总结了当前计算数据起源的主要方法和应用；文献[8]研究了科学工作流服务框架中的数据跟踪问题；文献[8-9]研究了数据起源追踪标注模式与描述模型，引入了7W模型；文献[10]研究了基于DNA双螺旋结构的数据起源追踪模型；文献[11]结合特定的应用背景、基于关系数据库设计了一种通用的数据追踪系统。本文以文献[1]的理论和方法为基础，研究面向ETL过程的数据起源追踪系统。

2 总体框架设计

面向ETL过程的数据起源追踪系统框架如图1所示，该系统主要由ETL模块和追踪模块构成。

(1)ETL模块实现数据的抽取转换加载，生成转换包文件，同时存储追踪起源所需的元数据。ETL模块支持ODBC数据源、XML文件、Excel文件和平面数据文件。数据的抽取依赖于连接管理器的管理和抽取规则的定义。

(2)追踪模块将追踪目标分为关系级、元组级、属性级和数据项级4个层次。根据追踪的层次，首先解析转换包，构建转换图并将其序列化，追踪过程则是将序列化的转换过程结合逆函数库和转换规则还原为转换图，并以可视化的方式根据需要展现为不同层次的起源图。

作者简介： 戴超凡(1973－)，男，副教授、博士，主研方向：智能辅助决策；王 涛，硕士研究生

收稿日期： 2011-02-28 **E-mail：** peter_king171@126.com

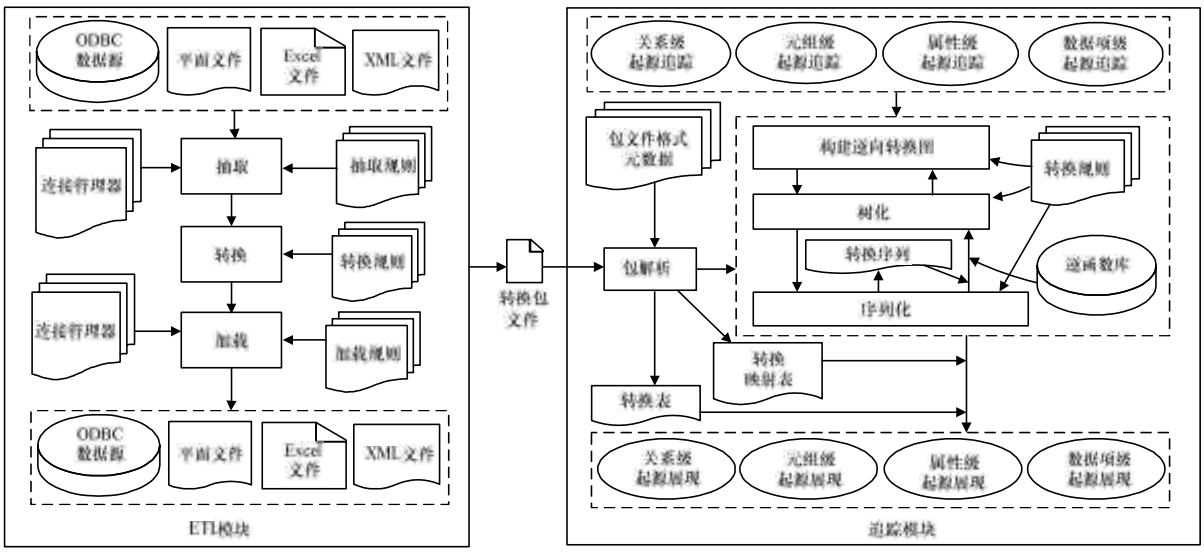

图 1　数据起源追踪系统总体框架

3　关键技术

数据的转换过程以及该数据转换前的数据状态是本系统追踪的目标，ETL 是本系统的基础功能，以 SQL Server 2005 中数据集成服务 SSIS 为内核，在此基础上添加追踪所需的元数据。本系统的重难点主要包括 ETL 过程涉及的转换、不同类型转换的追踪流程及算法、元数据模型设计、转换序列的构建以及追踪流程设计等。

3.1　ETL 过程的数据转换分类

根据微软 SQL Server 2005 中数据集成服务 SSIS 对转换的定义，ETL 过程涉及的转换主要包括行转换等 20 多种[12]。

根据数据起源追踪理论，数据转换可分为拆分转换、过滤转换等类型。而这些转换又是由单个或一系列的属性映射组成的，这些映射又可分为水平拆分、垂直拆分等。组成转换的映射的可逆性直接决定了转换的可逆性及数据起源追踪的精度。这些映射及转换的相关性质以及可逆性判断在文献[1]中有详细论述。根据 ETL 过程技术实现的实际，这些转换又可以划分为两大类[13]：同步转换和异步转换。上述 3 种分类方式有着不同的特点，实现追踪需要对每一种面向用户的转换首先区分其是同步转换还是异步转换，然后按追踪理论的分类方式将面向用户的转换逐一进行分析讨论。

3.2　元数据设计

元数据模型如图 2 所示。

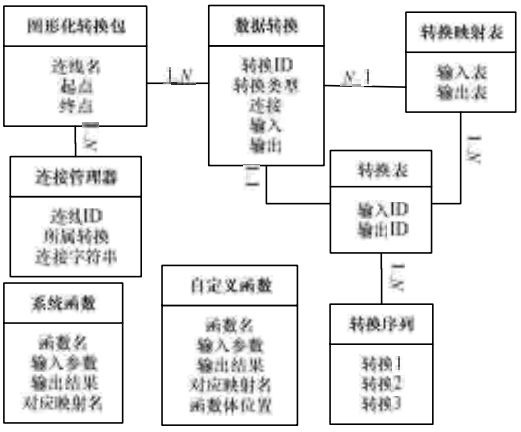

图 2　元数据模型

在起源追踪的过程中，需要存储的元数据可分为 3 类：

(1)转换过程的元数据，包括图形化转换包的元数据、转换的元数据以及连接管理器。转换包的元数据记录的是所有转换的源与目标及其执行关系，针对每一个转换需要记录其具体的转换属性等信息，针对数据源和数据目标需要建立连接管理器的相关元数据。

(2)追踪过程的元数据，包括转换表、转换映射表和转换序列三部分，每一个转换都有对应的转换表，映射表则是输入转换表与输出转换表之间的映射关系，转换序列是通过构建转换图经树化、序列化得到的结果。

(3)转换的逆函数元数据，包括系统函数和非系统函数两大类，系统函数不需要额外存储函数实体，非系统函数则需要存储。

3.3　转换序列的构建

构建转换图的过程就是可视化展示控制流的过程，也是构建 ETL 的逆过程。对于任何目标数据经历的转换，其逆向过程一定是一个有向无环图，最简单的转换过程表现为一棵只有一个叶节点的树。对于复杂的图状类型，将有多输入的节点拆开，有几个输入就视为几个节点，如图 3 中的 E 点将被视为 2 个独立的节点，从而将其转换为树状类型。对于树状类型，使用深度优先算法逐一遍历每一个叶节点，每个叶节点即对应一个转换序列。

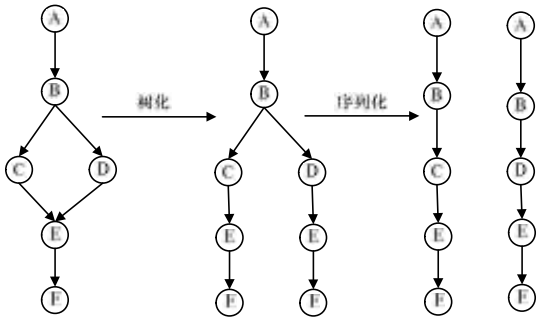

图 3　转换图树化、序列化示意图

这些转换序列将以关系表的形式存储到元数据库，其中对于转换树序列化的存储需要考虑存储结构的设计，由于各序列的深度可能不一致，因此该数据表结构可以采取 2 种方式确定：

(1)动态确定，即在加入记录时确定，每个节点做为元组的一个数据项，按节点顺序存储；

(2)静态确定，在存储之前，首先需要扫描追踪目标的整

个逆转换树，找出树的深度，将该表的属性项数设置为该深度值。

这 2 种方式得出的最终数据表是相同的，但动态方式可以在获取转换序列的同时对该序列进行处理，而静态方式则需要在扫描完整逆转换树之后才能够构建转换序列，动态方式的缺点在于需要经常修改表结构，而静态方式一旦确定了表结构之后，就不需要再进行修改。

4 数据起源追踪

4.1 追踪流程

数据起源的追踪过程与追踪精度密切相关，追踪精度可分为 4 个层次：关系级起源追踪，元组级起源追踪，属性级起源追踪和数据项级起源追踪。

追踪功能的实现首先由包解析模块将要追踪的转换包进行解析，从中抽取转换的相关信息，建立该 ETL 过程的元数据，形成控制流的转换框架。

对于指定的关系级起源追踪，首先确定关系所属的目标转换，从元数据库中构建该转换的逆转换树，属性级起源追踪，即列级起源追踪，需要更多的元数据支持，在构建的关系级转换序列基础上，列级查询需要结合元数据库中各个转换对应的转换表。

元组级起源追踪，即行级起源追踪，可以将元组划分为经过不同转换的属性组，以属性级追踪方式追踪每一组的数据起源，在将追踪结果合并，一一展现给追踪用户。数据项级起源追踪实现的关键是各转换的逆转换函数的设计，一旦转换在逆函数库中有对应的逆转换函数，则其追踪过程只需要在列级查询基础上根据逆函数逆向计算转换前的数据，将其作为上一转换逆函数的输入参与下一逆转换的计算，直到找出数据的起源。数据转换各个层次的追踪流程如图 4 所示。

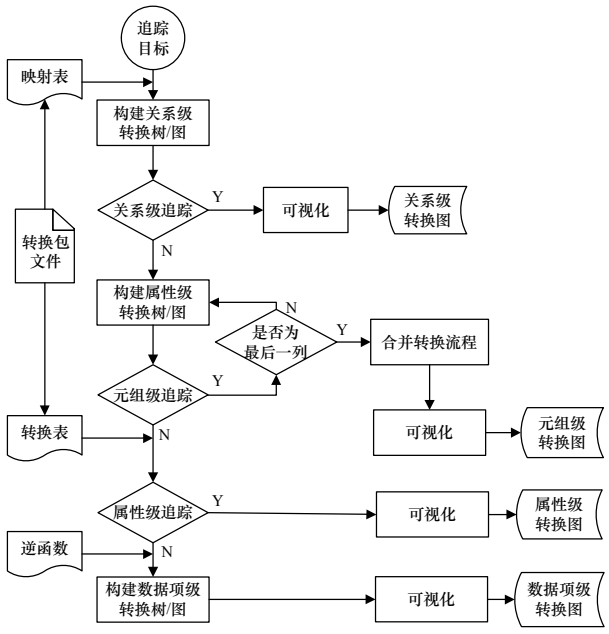

图 4 转换追踪流程

4.2 同步转换的追踪方法

同步转换是指从上游组件接收数据、只读或者修改其指定列的值，然后传入下游组件，其也可能定义从上游组件派生的数据列，但它不添加数据行到数据流中[12]。这类转换比较简单，例如字符映射转换中的更改大小写操作，当在组件中传递行时，组件通过修改指定列中的单个字符来更改字符串中字符的大小写。同步转换还可以细分为流转换和基于行的转换。

同步转换在处理过程中的特点是数据只针对特定处理列产生新列，不会影响其它列。设原转换为 C，对应逆转换为 RC，设原转换的输入为 p，得到的结果为 q，则原转换过程可表示为：C(p)=q，对应的逆过程可表示为：RC(q)=p。实现追踪的关键在与找到合适的 RC。此处 RC 可能是明确的数学函数，也可能是一系列的处理过程。

同步转换所包含的转换中有一些转换并未对数据做实质改变，这些转换的输出数据和转换前的输入数据是相同的。如复制列转换、条件性拆分等转换。复制列转换是将输入列的副本添加到转换输出的转换；条件性拆分转换是将数据行路由到不同输出的转换，此转换只是将输入数据行按照某种条件分流。根据文献[1]中对映射可逆性的判断，这些映射是可逆的，因此其构成的转换也是可逆的，并且其逆转换与原转换相同，可表述为：$C(p)=p \Leftrightarrow CR(q)=q$。

同步转换中其他对数据进行了实质性处理的转换，包括字符映射、数据类型转换等转换。下面以字符映射转换为例说明其追踪过程。字符映射表转换是将字符串函数应用于字符数据的转换。这是一系列的转换集合，包括很多转换，如大小写转换、全半角转换、简繁转换等。显然这类转换的逆转换也是一系列的转换集合，由于这类转换大都成对出现且互逆，因此其逆过程可以很方便地找到对应的逆函数。这些转换也只是由单一的映射构成，而且这些映射都能够找到相应的逆映射，因此由这些映射构成的转换可逆。以大小写转换为例，其映射关系可描述为为 U(p)=P，则对应的逆操作应为将小写转换为大写，即该映射对应的逆映射应为小写转换 L(P)=p，即 $U(p)=P \Leftrightarrow L(P)=p$。

理论上，建立该转换的逆转换非常简单，但实际处理并不简单，甚至无法逆转。因为实际过程中转换之前的数据可能是大小写混杂的字符串，转换操作只是将原来的大写转换成了小写，如果不知道原来的数据大小写分布规律就很难逆向追溯原来的数据状态。

4.3 异步转换的起源追踪方法

异步转换用于 2 种情况[14]：该转换只有在获取了所有的输入数据之后才能进行，或者该转换并不针对每一个具体输入行产生对应的输出行。例如聚合转换，其只有在所有输入数据行全部读取之后才能进行聚合计算。与同步转换相比，异步转换必须定义输出列向下游转换提供数据行，而同步转换则不需要自定义输出数据，只需要设置好数据映射关系就可以实现转换。异步转换也可以细分为半异步和全异步两种方式[13]，半异步是指该转换可以在输入的同时进行实时转换处理，部分转换结果可以释放到下游转换，但最终的结果必须要在所有输入结束之后才能得出，如合并转换，其处理过程中很多行进入该转换并不能立刻进行处理，而是要等待匹配的输入数据合并之后才能向下输出，这类转换包括合并、词查找等转换；全异步转换是指该转换的处理必须等到所有输入全部完成才能进行，在处理完成之前不得释放任何结果进入下游转换。包括聚合、排序等转换。例如要计算某列数据的和或者对某行进行排序，都必须在所有输入全部完成之后才可能进行。

异步转换一般是不可逆的，但在特定条件下的某些异步转换也是可逆的，对同一种转换，其可逆性在不同条件下也是不同的。如对于 SUM 聚合，设聚合过程为 $f=\mathrm{SUM}(v_i)$, $i=1, 2,\cdots,n$，$n \in \mathbf{N}$。 (下转第 261 页)