(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 101606154 B
(45)授权公告日 2016.09.28

(21)申请号 200880003975.2

(22)申请日 2008.02.05

(65)同一申请的已公布的文献号
申请公布号 CN 101606154 A

(43)申请公布日 2009.12.16

(30)优先权数据
11/671,414 2007.02.05 US

(85)PCT国际申请进入国家阶段日
2009.08.04

(86)PCT国际申请的申请数据
PCT/US2008/053095 2008.02.05

(87)PCT国际申请的公布数据
WO2008/098009 EN 2008.08.14

(73)专利权人 微软技术许可有限责任公司
地址 美国华盛顿州

(72)发明人 H·J·M·梅杰 A·K·西尔弗 P·A·维克 E·扎伯克利特斯基 A·V·青高兹

(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 顾嘉运 钱静芳

(51)Int.Cl.
G06F 17/30(2006.01)

(56)对比文件
US 2005/0203869 A1,2005.09.15,
CN 1881208 A,2006.12.20,
US 2006/0294081 A1,2006.12.28,
US 2006/0224627 A1,2006.10.05,

审查员 张雯

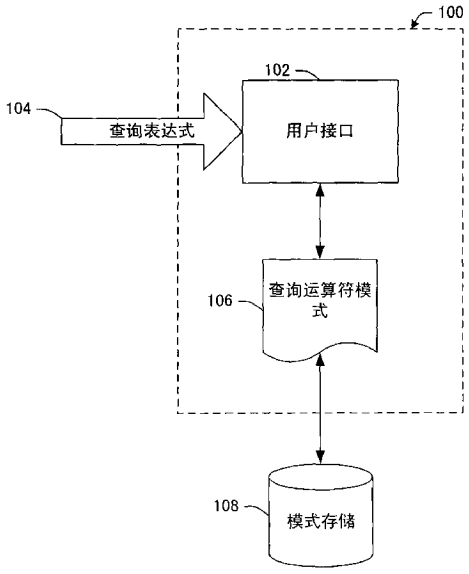
权利要求书2页 说明书32页 附图10页

(54)发明名称

允许元素类型的类型流的查询模式

(57)摘要

所要求保护的主题涉及运算符模式的形式化，其可方便表达式的第一子句中的运算符到表达式的下一子句中的运算符的元素类型的类型流。流到下一子句的类型以及其中现存的运算符可被组合以推断下一子句的元素类型。由此，可增量式地实现类型检查、自动完成和其它有利特征而无需在前对表达式的完全转换。



CN 101606154 B

1.一种用于方便元素类型的类型流的计算机实现的方法，包括：

接收查询子句，所述查询子句包括与查询运算符模式相关联的查询运算符、元素类型T和源类型，所述源类型是可查询类型，其中所述查询运算符映射到根据所述查询运算符模式定义的方法调用，所述方法调用表达所述查询运算符，并且其中所述元素类型是作为关于所述源类型的控制变量被引入的，所述控制变量的范围基于所述查询运算符来确定，并且范围中的控制变量能够被传递到下一查询子句；

利用所述源类型和所述查询运算符来确定元素类型；以及

采用前一查询子句的元素类型作为所述利用动作的源类型。

2.如权利要求1所述的方法，其特征在于，所述查询运算符是笛卡尔积形式，并且所述元素类型T是匿名类型，该匿名类型是关于所述源类型的范围中的全部控制变量的聚集。

3.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型S的可查询类型的投影或映射查询运算符，该函数包括元素类型T的自变量。

4.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型T的可查询类型的过滤或限制查询运算符，该函数包括元素类型T的自变量。

5.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型T的已排序类集的排序或定序查询运算符，该函数包括元素类型T的自变量。

6.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成返回具有元素类型T的可查询类型的重复移除查询运算符。

7.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一元素类型T为源类型的类集作为自变量，并返回具有元素类型T的可查询类型的集合查询运算符。

8.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收值作为自变量并返回具有元素类型T作为源类型的可查询类型的选择或过滤查询运算符。

9.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型T的可查询类型的条件选择或过滤查询运算符，该函数包括元素类型T作为自变量。

10.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回数值类型的聚集查询运算符，该函数包括元素类型T的自变量。

11.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型S的可查询类型的聚集查询运算符，该函数包括元素类型T的自变量。

12.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回可被隐式地转换成布尔值的类型的匹配查询运算符，该函数包括类型T的自变量。

13.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型V的可查询类型的联接查询运算符，该函数包括元素

类型T的自变量以及元素类型T'的自变量，其中V是散列值。

14.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括被配置成接收一函数作为自变量并返回具有元素类型V的可查询类型的嵌套联接查询运算符，该函数包括元素类型T的自变量以及具有元素类型U的类集的自变量。

15.如权利要求1所述的方法，其特征在于，所述查询运算符模式包括要接收一函数作为自变量并返回具有元素类型V的可查询类型的分组查询运算符，该函数包括类型K的自变量以及具有类型T的类集的自变量，其中类型K是关键字。

16.如权利要求1所述的方法，其特征在于，还包括基于所述源类型以及与至少一个下一查询子句相关联的查询运算符来实时地推断元素类型。

17.如权利要求1所述的方法，其特征在于，还包括使用所述查询运算符模式来增量式地提供与所述元素类型或源类型的控制变量相关联的上下文反馈。

18.一种用于方便元素类型的类型流的计算机实现的系统，包括：

用于接收查询子句的装置，所述查询子句包括与查询运算符模式相关联的查询运算符、元素类型T和源类型，所述源类型是可查询类型，其中所述查询运算符映射到根据所述查询运算符模式定义的方法调用，所述方法调用表达所述查询运算符，并且其中所述元素类型是作为关于所述源类型的控制变量被引入的，所述控制变量的范围基于所述查询运算符来确定，并且范围中的控制变量能够被传递到下一查询子句；

用于利用所述源类型和所述查询运算符来确定元素类型的装置；以及

用于采用前一查询子句的元素类型作为所述利用动作的源类型的装置。

允许元素类型的类型流的查询模式

[0001] 发明背景

[0002] 在计算机语言的领域中，传统上有许多不同的语言类别。例如，许多编程语言目的是为大多数最终用户编程目标提供一般的解决方案，但另一方面，查询语言却通常专用于基于对数据库查询的信息挖掘。多年来，已经进行了许多尝试来将查询能力添加到编程语言，但是存在很多困难。

[0003] 一个这样的困难是大多数现代编程语言是基于面向对象的模型的，该模型允许分层、抽象、模块性、封装和旨在降低编程任务的复杂度的若干其它范例。另一方面，现代数据库很大一部分是关系型数据库，因此查询语言往往是基于关系模型而非面向对象的模型。

[0004] 另一困难是用于编程语言的开发环境(例如，集成开发环境(IDE))已经进化成向开发者提供非常精密的协助手段，导致没有这种开发环境来编程变得很费劲。一个示例是诸如自动完成实用程序或机制等内联上下文信息。这些自动完成机制在其中类、变量名和其它构造事先定义的面向对象的领域能起很好的作用，但是通常不可用于查询语言，因为对于自动完成所必需的元素类型的类型检查在编译或转换了查询之前是不可用的。此外，查询在完成该查询之前不能被转换，这使得自动完成机制没有实际意义。再者，如果查询表达式是不良地形成的，则编译错误将是晦涩难解的且难以补救。

[0005] 发明概述

[0006] 以下提出了所要求保护的主题的简化概述以提供对所要求保护的主题的某些方面的基本理解。本概述并不是对所要求保护的主题的全面综述。它既不旨在标识所要求保护的主题的关键或重要的元素，也不描绘所要求保护的主题的范围。其唯一目的是以简化的形式来介绍所要求保护的主题的一些概念，作为稍后呈现的更为详细的描述的前序部分。

[0007] 此处所公开并要求保护的主题的一方面包括用于方便表达式中的运算符之间的元素类型的类型流的计算机实现的技术。运算符可以是，但不限于，查询运算符，并且由此，类型可以贯穿整个查询表达式从一个查询子句流到下一查询子句。

[0008] 根据所要求保护的主题的一方面，运算符可被映射到相关联的方法调用，并且方法调用可以根据运算符模式来定义。由此，可期望表达式中的任何给定运算符遵循形式化的运算符模式。相应地，算符以及整个表达式的项可以按照所期望的方法调用来表达。可以理解，方法可以是实例方法、静态方法、虚拟方法或扩展方法。

[0009] 据此，通过方便类型流并约束运算符以遵循运算符模式，此处所公开的体系结构可以对表达式的每一子句递增地推断元素类型。例如，元素类型可以通过将源类型与运算符相组合来确定，并且这可以对表达式的每一连续的子句实时地完成。由此，类型信息可以本地地解析而无需完全转换整个表达式。因此，可以用较不昂贵(在资源利用方面)的方式且更快速地执行元素类型的类型检查，并且元素类型的类型检查可以在表达式的构造期间而非仅在表达式被最终化之后才实现，这可以产生附加的益处。

[0010] 根据所要求保护的主题的一方面，一个这样的附加益处是可以对于表达式采用自动完成机制。例如，通过在进行中推断元素类型，可提供基于可用类型的上下文信息，这可

以帮助表达式构造。

[0011] 以下描述和附图详细阐明了所要求保护的主题的某些说明性方面。然而，这些方面仅指示了可采用所要求保护的主题的原理的各种方法中的几种，且所要求保护的主题旨在包括所有这些方面及其等效方面。当结合附图考虑以下所要求保护的主题的详细描述时，所要求保护的主题的其它优点和区别特征将变得显而易见。

[0012] 附图简述

[0013] 图1是可采用查询模式来方便元素类型的类型流的计算机实现的系统的框图。

[0014] 图2A示出了对于一示例性查询表达式的非限制性代表性第一查询子句和各种非限制性代表性下一查询子句。

[0015] 图2B更详细地示出了示例性查询表达式的查询子句以及元素类型的类型流。

[0016] 图3描绘了具有示例性输入和输出的查询运算符模式的各种非限制性示例。

[0017] 图4是实时地推断元素类型和/或基于所推断的元素类型递增地提供上下文信息的计算机实现的系统的框图。

[0018] 图5描绘了定义用于方便查询表达式内的元素类型的类型流的计算机实现的方法的过程的示例性流程图。

[0019] 图6是可方便可组成查询综合和/或可扩展查询表达式的计算机实现的系统的框图。

[0020] 图7示出了与从示例性查询表达式的示例性查询子句得到的控制变量的范围有关的示例性描述的框图。

[0021] 图8描绘了定义用于方便以组成方式构造查询综合的计算机实现的方法的过程的示例性流程图。

[0022] 图9示出了可用于执行所公开的体系结构的计算机的框图。

[0023] 图10示出示例性计算环境的示意性框图。

[0024] 详细描述

[0025] 现在参考附图来描述所要求保护的主题，所有附图中使用相同的附图标记来指代相同的要素。在以下描述中，为解释起见，阐明了众多具体细节以提供对所要求保护的主题的全面理解。然而，很明显，所要求保护的主题可以在没有这些具体细节的情况下实施。在其它情况下，以框图形式示出了公知的结构和设备以便于描述所要求保护的主题。

[0026] 如在本申请中所使用的，术语“组件”、“模块”、“系统”、“接口”等一般旨在表示计算机相关的实体，其可以是硬件、硬件和软件的组合、软件、或者执行中的软件。例如，组件可以是，但不限于是，在处理器上运行的进程、处理器、对象、可执行码、执行的线程、程序和/或计算机。作为说明，运行在控制器上的应用程序和控制器都可以是组件。一个或多个组件可以驻留在进程和/或执行的线程中，并且组件可以位于一个计算机内和/或分布在两个或更多的计算机之间。作为另一示例，接口可包括I/O组件以及相关联的处理器、应用程序、和/或API组件，并且可以像命令行那样简单，或者是更复杂的集成开发环境(IDE)。

[0027] 此外，所要求保护的主题可以使用产生控制计算机以实现所公开的主题的软件、固件、硬件或其任意组合的标准编程和/或工程技术而被实现为方法、装置或制品。在此使用的术语“制品”旨在涵盖可以从任何计算机可读设备、载体或介质访问的计算机程序。例如，计算机可读介质可以包括但不限于磁存储设备(例如，硬盘、软盘、磁带……)、光盘(例

如，紧致盘(CD)、数字多功能盘(DVD)……)、智能卡和闪存设备(例如，卡、棒、钥匙驱动器……)。另外应该明白，可以采用载波来承载计算机可读电子数据，例如那些用于发送和接收电子邮件或用于访问如因特网或局域网(LAN)等网络的数据。当然，本领域的技术人员将会认识到，在不背离所要求保护的主题的范围或精神的前提下可以对这一配置进行许多修改。

[0028] 此外，在此使用词语“示例性”意指用作示例、实例或说明。在此被描述为“示例性”的任何方面或设计并不一定要被解释为相比其它方面或设计更优选或有利。相反，使用词语示例性旨在以具体的方式呈现各个概念。如本申请中所使用的，术语“或”意指包括性“或”而非互斥性“或”。即，除非另有指定或从上下文可以清楚，否则“X使用A或B”意指任何自然的包括性排列。即，如果X使用A；X使用B；或X使用A和B两者，则在任何以上情况下，都满足“X使用A或B”。另外，本申请中和所附权利要求书中所使用的冠词“一”和“一个”一般应被解释为是指“一个或多个”，除非另有指定或从上下文可以清楚指的是单数形式。

[0029] 如在此所使用的，术语“推断”或“推论”通常是指从经由事件和/或数据捕捉的一组观察结果中推断或推理系统、环境和/或用户的状态的过程。例如，推断可用于标识特定的上下文或动作，或可生成状态的概率分布。推断可以是概率性的，即，基于对数据和事件的考虑计算所关注状态的概率分布。推断也可以指用于从一组事件和/或数据组成更高级事件的技术。这类推断导致从一组观察到的事件和/或存储的事件数据中构造新的事件或动作，而无论事件是否在相邻时间上相关，也无论事件和数据是来自一个还是若干个事件和数据源。

[0030] 现在参考附图，最初参考图1，描绘了能够采用查询模式来方便元素类型的类型流的计算机实现的系统100。一般而言，系统100可包括可操作地耦合到公知的基于计算机的硬件(例如，控制器)、软件(例如，应用程序)、以及此处所描述的其它组件的用户接口102。用户接口102可接收查询表达式104，该表达式可包括第一查询子句和一个或多个下一查询子句。可以理解，第一查询子句可以包括可查询源类型(例如，类集、流等)以及可由第一查询子句作为例如关于源类型的控制变量来引入的元素类型。通常，所有查询子句包括查询运算符，然而，如所见的，查询表达式104的下一查询子句不必是完整的，而是可以按部分(例如，增量式地)接收，诸如在子句由用户输入的时候。查询表达式104和相关的子分量在下文中结合图2A和2B来更详细地描述。

[0031] 系统100还可包括用于任何有效查询运算符(例如，与第一查询子句和后续的下一查询子句相关联的查询运算符)的查询运算符模式106。因此，在例如可操作地耦合到系统100的模式存储108中可以存在任何数量的查询运算符模式106。特别地，来自查询表达式104的查询子句(例如，各自具有特定的查询运算符)可以符合相关联的查询运算符模式106，该模式可以定义例如可访问实例方法。通过符合查询运算符模式106，可以实质上确保在一个查询子句和下一查询子句之间存在已知关系，并且因此存在从一个查询运算符到下一查询运算符的关系。查询运算符模式106结合图3来更详细描述。

[0032] 应该理解，通过对查询表达式104施加约束以使查询表达式104的运算符符合查询运算符模式106，系统100可以确保从查询表达式104的一个查询子句/运算符到下一查询子句/运算符的元素类型的类型流。此外，通过确保此处所描述的元素类型流，可以在进行中实现某些有利的类型推断和/或上下文判定，而无需对完整查询表达式104的常规转换/编

译，如参考图4更详细讨论的。此外，上述推断和/或判定可以在查询表达式104完成之前，并甚至在查询表达式104的形式错误时提供。作为进一步的解释而非限制，符合查询运算符模式106的示例查询表达式104以及与该架构化相关联的某些优点在下文中更详细描述。

[0033] 仍参考图1，但也转向图2A，示出了示例性查询表达式104。一般而言，查询表达式104可以是向特定类集应用一系列查询运算符的表达式。查询运算符是可跨值的类集一次应用于该整个类集的运算符(例如，FROM、WHERE、SELECT.....)。查询表达式104可包括第一查询子句202，以及任何数量的下一查询子句204_1、204_2等，其每一个都包含查询运算符中的一个。

[0034] 尽管可存在许多其它查询运算符，并且这被认为是在所要求保护的主题的精神和范围之内，但是在下文中提供了多个具体示例以及对每一示例的简要描述。

[0035] • FROM(从)运算符可以引入一个或多个控制变量，并且或者指定要查询的类集，或者为控制变量计算值。

[0036] • RETURN(返回)和SELECT(选择)运算符可以指定输出类集的形状。在某些情况下，SELECT运算符可以引入新的控制变量。

[0037] • WHERE(其中)和DISTINCT(不同)运算符可以限制类集的值。

[0038] • ORDERBY(按……排序)运算符可以对类集施加排序。

[0039] • SKIP(跳过)、SKIPWHILE(跳过同时)、TAKE(获取)和TAKEWHILE(获取同时)运算符可以基于次序或条件返回类集的子集。

[0040] • UNION(并)、UNIONALL(并全部)、EXCEPT(除外)和INTERSECT(交)运算符可以接收两个类集并产生单个类集。

[0041] • GROUP(组)运算符可以聚集类集并可返回按照关键字来分组的类集。

[0042] • GROUPBY(按……分组)运算符可以基于一个或多个关键字来对类集分组。在某些情况下，GROUPBY运算符可以引入新的控制变量。

[0043] • AVG(平均)、SUM(求和)、COUNT(计数)、MIN(最小)和MAX(最大)运算符可以聚集类集并产生值。

[0044] • ANY(任何)和ALL(全部)运算符可以聚集类集并基于条件返回布尔(BOOLEAN)值。

[0045] • JOIN(联接)运算符可以接收两个类集，并可以基于从元素中导出的匹配关键字来产生单个类集。

[0046] • GROUPJOIN(组联接)运算符可以基于从元素中提取的匹配关键字来对两个类集执行分组联接。

[0047] 查询表达式104中的查询运算符的转换可以按照运算符出现在查询表达式104中的次序从左到右来进行。通常，查询表达式104的第一个查询子句202包括FROM运算符，因为FROM运算符引入要查询的类集。另外，查询表达式104中的最后一个查询子句一般包括指定从查询所得的类集的最终形状的RETURN和SELECT运算符。然而，可以理解，最后的SELECT或RETURN运算符可被省略。例如，如果查询表达式104不是以RETURN或SELECT运算符结束，则可以假定隐含的SELECT运算符，这可将范围中的所有控制变量提升为所返回的匿名类型的属性。此外，应当理解，查询表达式104可以在应用了SELECT运算符之后继续，在该情况下，后续运算符可访问的控制变量仅限于范围中的那些控制变量。

[0048] 根据所要求保护的主题的一方面，查询表达式104可如下架构化：

[0049] 查询表达式::＝查询运算符列表

[0050] 查询运算符列表::＝

[0051] From运算符|

[0052] 查询运算符列表 查询运算符

[0053] 查询运算符::＝

[0054] From运算符|

[0055] Select运算符|

[0056] Return运算符|

[0057] Distinct运算符|

[0058] Where运算符|

[0059] OrderBy运算符|

[0060] Partition运算符|

[0061] Set运算符|

[0062] GroupBy运算符|

[0063] Join运算符|

[0064] GroupJoin运算符

[0065] 为了提供用于所要求保护的主题的附加上下文以及相关编程实现和/或语言的高级概览，考虑示例性查询表达式104的整体：

[0066] Dim names＝_

[0067] From C In Customers_

[0068] Where C.State＝“WA”_

[0069] Select C.Name

[0070] 该查询表达式104可以取Customer(顾客)对象的类集，并返回具有华盛顿(WA)中的每一顾客的单个Name(姓名)属性的行的类集。

[0071] 很容易明白，查询表达式104句法可以相当接近于标准关系型结构查询语言(SQL)句法，其意图是熟悉SQL的任何人将能够以极少的进一步指令来使用查询表达式104。然而，该句法不必受SQL约束，并且此外，查询表达式104不必是SQL到另一编程语言的转换。由于SQL是围绕纯关系型模型来设计的，因此其某些习惯用语在包含分层概念的类型系统中并不能起很好的作用。另外，SQL通常采用的某些句法和语义元素会冲突或者不能与现有的编程语言句法或语义很好地集成。由此，尽管查询表达式104对于熟悉SQL的用户而言可以是直观的，但是也可存在某些区别。

[0072] 例如，在SQL以及诸如过程语言/结构化查询语言(PLSQL)等其它语言中，查询和查询的结果通过对整个查询求值来绑定。由此，不需要能够解释所应用的特定查询运算符的元素类型。此外，这一判定在输入了整个无错误的查询并且通过相对昂贵的转换过程绑定之前是不能做出的。相反，所要求保护的主题能够将查询表达式104输入和查询表达式104的结果按照特定源类型上的个别查询运算来绑定。这些和其它优点可以参考图2B来更深入地示出。

[0073] 图2B更详细地示出了示例性查询表达式104的查询子句。在该示例中，第一查询子

句202包括第一查询运算符(例如,FROM运算符206),其可用语引入可查询源类型(例如,Customers 208),并且可声明用于该源类型的元素类型的控制变量210(例如,C)。如图所示,第一查询子句202的结果流212到下一查询子句204_1(此处统称为下一查询子句204),使得在相邻查询子句之间可存在已知关系。特别地,元素类型可流212到下一查询运算符(例如,WHERE运算符214)以用作下一查询子句204_1的源类型。同样,下一查询子句204_1的输出流212到下一查询子句204_2,其中它与下一查询运算符(例如,SELECT运算符216)相关联,该运算符也可产生流212到后续的下一查询子句204的元素类型或描述查询表达式104的输出的最终形状。

[0074] 特别地,用于下一查询子句204的每一个的查询运算可以基于一组方法(例如,查询运算符模式106),其中每一方法与一特定查询运算符相关联,并且被应用于类集或序列(例如,源类型)。这些查询运算符可以相关于类集的元素类型(或行)来定义。然而,给定具有元素类型T并应用了查询运算符的特定源类型,流212到下一查询运算符204的结果可以是具有元素类型S的类集。因此,流动212的元素类型可以根据与给定查询运算符相关联的查询运算符模式106来变换。例如,与诸如SELECT和RETURN等投影查询运算符相关联的查询运算符模式106一般返回与从流212接收到的元素类型不同的元素类型。

[0075] 在本示例中,元素类型T(其可以是例如与Customer类型208相关联的所有控制变量210的聚集)流212到下一查询子句204_1的WHERE运算符214。如将结合图3描述的,与WHERE运算符214相关联的查询运算符模式106可以过滤该类集的值,但是在其它方面通常不改变元素类型。此处,不在华盛顿州的顾客从类集中过滤掉,但是流212到下一查询运算符,即SELECT运算符216的结果仍是具有元素类型Customer 208的类集。

[0076] 另一方面,如果假定Name属性是String(串)类型的,则SELECT运算符216的结果是具有元素类型String的类集。对查询结果同时支持SELECT和RETURN运算符的组合可提供对运算符的输入和输出的可预测语义。另外,可以促进更大的灵活性以在与关系型数据交互时精确地模仿期望的SQL语义。此外,在产生非表结果时可以促进更大的简单性,以及在迭代地形成查询表达式时可以促进对编译器错误和所需改变的改进的定位。

[0077] 另外,对于可查询可扩展标记语言(XML)数据的类集的某些查询语言(例如,XQuery,其是包括某些编程语言特征的查询语言);所要求保护的主题也可结合XML文字来使用。例如,可以采用以下示例性查询表达式104来返回包括在华盛顿的所有顾客的姓名的XML元素类集,其类似于以上所讨论的示例性查询表达式104的结果。

[0078] Dim names＝_

[0079] From c In Customers_

[0080] Where c.State＝“WA”_

[0081] Return<Name><%＝c.Name%></Name>

[0082] 还可以理解,诸如,例如FROM、SELECT和GROUPBY等某些查询运算符可以引入称为控制变量(例如,控制变量C 210)的一种特殊的局部变量。默认地,控制变量的范围可以定为从引入的运算符到可以隐藏该控制变量的运算符,并且可以表示查询所求值的类集中的个别行的属性或列。例如,在以下查询中:

[0083] Dim WACusts＝_

[0084] From C As Customer In Customers_

[0085] Where C.State＝“WA”

[0086] FROM运算符引入了类型为Customers的控制变量C。后面的WHERE查询运算符然后参考控制变量C在过滤表达式C.State＝"WA"中表示每一个别顾客。

[0087] 诸如DISTINCT等某些查询运算符不必使用或改变控制变量。诸如SELECT等其它查询运算符可以在范围中隐藏当前控制变量并且可引入新的控制变量。例如，在以下查询中：

[0088] Dim YourUncles＝_

[0089] From C In Customers_

[0090] Select LastName＝C.Name_

[0091] Where LastName.StartsWith(“Bo”)

[0092] WHERE查询运算符只能访问由SELECT运算符引入的LastName(姓)控制变量。如果WHERE运算符试图引用C，则可能会导致编译时错误。

[0093] 现在参考图3(同时仍参考图1)，提供了查询运算符模式的众多非限制示例。查询运算符模式302-328是查询运算符模式106的具体图示，并且此处由参考标号302-328来个别地引用，或者统称为查询运算符模式106。还可以理解，若干查询运算符模式106可适用于多个查询运算符，即使多个查询运算符中涉及查询运算符模式302-328中的单个模式的每一查询运算符可具有将其与多个查询运算符中的其它运算符相区分的特性。此外，在某些情况下，特定查询运算符可以结合查询运算符模式302-328中的一个以上模式来使用(例如，重载函数)。

[0094] 一般而言，查询运算符模式106可以是类型C必须实现以便可被查询的方法签名。可以回想，查询表达式104可以向特定类集应用一系列查询运算符。每一查询运算符的控制变量可以是用于应用该特定查询运算符的范围中的全部变量或全部变量的子集，而可从一个查询运算符流到下一查询运算符的元素类型T可以是作为范围中的全部控制变量的聚集的匿名类型。因此，查询运算符模式106可以特别地帮助确保类型C是可查询类型。对于任何给定查询表达式104，如果以下条件的至少一个为真，则类型C是可查询类型：

[0095] (1)类型C包括返回可查询类型的具有签名AsQuerable()的可访问实例方法。

[0096] (2)类型C包括返回IEnumerable(Of T)的具有签名AsEnumerable()的可访问实例方法。

[0097] (3)类型C实现符合查询运算符模式302-328(下文进一步详述)的一个或多个查询运算符，并且对于在类型C上定义的所有查询运算符，类型T必须匹配并且不可以是开放类型参数。

[0098] 参考查询运算符模式302-328，查询运算符模式302可包括FROM运算符。在查询表达式104中，FROM运算符通常在第一查询子句中使用。由此，FROM模式302可以引入查询的源以及要使用的控制变量，并且可输出具有元素类型T的可查询类型。应当理解，在对查询表达式求值之前，如果类型C不满足设计模式(例如，不符合查询运算符模式106)，则可对查询表达式104调用AsQueryable或AsEnumerable方法，并且该函数的返回值可被存储在临时位置中。

[0099] 如上所述，FROM运算符可以引入可被查询的类集以及可表示该类集中的个别成员的控制变量。查询表达式：

[0100] From b As Book In Books...

[0101] 可被认为等价于

[0102] For Each b As Book In Books

[0103] Next b

[0104] 类集表达式运算数一般必须被分类为值并且必须是可查询类型。FROM运算符所声明的控制变量通常必须遵循用于相对于命名和定范围来声明局部变量的常规规则(如以上句法转换所暗示的)。由此,控制变量一般不能隐藏封闭方法中的局部变量或参数的名称。

[0105] 应当理解,FROM运算符还可引入其值可由表达式而非类集来确定的控制变量。这一特征可用于例如计算将在稍后的查询运算符中使用多次的值(例如,计算该值一次而非在随后每次使用它时计算)。例如:

[0106] Dim TaxedBookPrices＝_

[0107] From b in Books_

[0108] From Tax＝b.Price*0.088_

[0109] Where Tax＞3.50_

[0110] Select b.Price,Tax,Total＝b.Price+Tax

[0111] 可被认为等价于:

[0112] For Each b In Books

[0113] Dim Tax＝b.Price*0.088

[0114] Next b

[0115] 声明表达式控制变量的FROM运算符的句法可以与FOR LOOP中的控制变量声明相同,例外是该控制变量一般通过显示初始化程序来初始化。不要求表达式控制变量引用另一控制变量,因为这样做可能是不确定值。表达式控制变量通常不能是在查询表达式104中声明的第一控制变量。

[0116] 出于简短和/或方便的目的,FROM运算符的运算数可以省略AS子句,在这一情况下,控制变量的类型可以从该变量范围所在的类集或表达式来推断。如果控制变量的类型不能从查询运算符方法中导出,则将导致编译时错误。FROM运算符可以在架构上如下定义:

[0117] From运算符::＝

[0118] From From声明列表

[0119] From声明列表::＝

[0120] From声明|

[0121] From声明列表,From声明

[0122] From声明::＝

[0123] 变量标识符[As类型名]In表达式|

[0124] 变量声明函数

[0125] 另外,尽管FROM运算符一般作为第一个查询子句202的查询运算符出现,但是查询表达式104可以包括一个以上FROM运算符。因此,当FROM运算符是下一查询子句204的查询运算符时,则可导致叉积(例如,简单、隐式联接)。在这一情况下,FROM运算符可以将新控制变量联接(参见下文的JOIN运算符模式324)到现有的控制变量集。结果可以是联接的类集中的所有元素的叉积。因此,例如,表达式

[0126] From b In Books_

[0127] From p In Publishers_

[0128] 可被认为等价于嵌套的For Each循环

[0129] For Each b In Books

[0130] For Each p In Publishers

[0131] Next p

[0132] Next b

[0133] 前一查询运算符中引入的控制变量可以在范围内，并且可在包括在下一查询运算符204中的FROM运算符内使用。按照SQL，上述特征可被认为是支持“相关子查询”。例如，在以下查询表达式中，第二个FROM运算符涉及第一个控制变量的值：

[0134] From c As Customer In Customers_

[0135] From o As Order In c.Orders_

[0136] Select c.Name,o

[0137] 另外，在单个查询子句中可以出现多个FROM运算符，在这一情况下，运算数之间的逗号可以完全等价于下一查询子句204中的另一FROM运算符。由此，示例：

[0138] From b In Books,p In Publishers_

[0139] Select b,p

[0140] 可以等价于

[0141] From b In Books_

[0142] From p In Publishers_

[0143] Select b,p

[0144] 查询运算符模式304可以包括一查询运算符，其是名为SELECT或SELECTMANY的可访问实例方法，该方法可以接收一函数作为自变量。函数可以包括类型为T的自变量，并产生类型S。该方法可以返回具有元素类型S的可查询类型。

[0145] SELECT运算符可以描述查询表达式104的结果。SELECT运算符可以取声明列表，并构造所得类集的元素类型。例如，如果查询表达式104为：

[0146] Dim CustAndOrderNames＝_

[0147] From c In Customers,o In c.Orders_

[0148] Select c.Name,o.Product

[0149] 则所得类型可以是IEnumerable(Of{Name As String,Product As String})，因为Name的类型是String，且Product(产品)的类型是String。

[0150] 后面的这一查询(例如，查询表达式104)可以等价于显示地返回匿名类型：

[0151] Dim CustAndOrders＝_

[0152] From c In Customers,o In c.Orders_

[0153] Return New With{c.Name,o.Product}

[0154] 如果SELECT运算符只有一个声明，则结果可以是具有一个属性的匿名类型。由此，结果类型可以是IEnumerable(Of{Name As String})，因为Name的类型是String。例如，

[0155] Dim CustNames＝_

[0156] From c In Customers

[0157] Select c.Name

[0158] SELECT运算符中使用的声明内的表达式在转换成底层匿名类型之前被绑定。更具体而言，SELECT运算符中的声明内的表达式可以开始成员访问，其中点运算符不必绑定到查询中的控制变量。相反，点运算符可改为绑定到封闭的WITH块，如果存在这样的块的话。

[0159] 应当理解，除了SELECT运算符之外，查询表达式104还可包括其它投影运算符。例如，查询表达式可以包括RETURN运算符，对于该运算符，可以有相关联的查询运算符模式106。类似于SELECT运算符，RETURN运算符可以描述查询表达式104的结果。RETURN运算符可以取产生所得类集的元素的表达式。例如，查询：

[0160] Dim ReducedBookPrices＝_

[0161] From b in Books_

[0162] Return b.Price*.8

[0163] 可产生被降低到其原始成本的80％的价格的类集。

[0164] 以RETURN运算符结束的查询表达式104的结果可以是其元素类型是所返回的表达式的类型的类集。例如，在以下查询表达式中，结果类型是IEnumerable(OF String)，因为c.Name的类型是String：

[0165] Dim CustNames＝_

[0166] From c In Customers_

[0167] Return c.Name

[0168] 如果查询表达式104在没有RETURN或SELECT运算符的情况下结束，则所得的类集的元素类型可以是具有范围中的所有控制变量的属性的匿名类型：

[0169] ′结果类型是IEnumerable(Of{Name As String,Product As String})

[0170] Dim CustNames＝_

[0171] From c In Customers,O In C.Orders

[0172] 即使SELECT运算符可以描述查询表达式104的结果，但任何查询表达式104可在SELECT运算符之后继续。在该情况下，由SELECT语句引入的控制变量是范围中的控制变量，但是所有先前的控制变量通常在范围外：

[0173] CustNames＝_

[0174] From c In Customers,O In C.Orders

[0175] Select Name＝C.Name,Price＝O.Price

[0176] Where Price＞500

[0177] Return Name

[0178] 在SELECT运算符之后继续的查询表达式104可以等价于嵌套查询：

[0179] Dim CustNames＝_

[0180] From X In(From c In Customers,O In C.Orders_

[0181] Return New With{C.Name,O.Price}),_

[0182] Name＝X.Name,Price＝X.Price

[0183] Where Price＞500

[0184] Return Name

[0185] RETURN和SELECT运算符分别可基于以下示例来定义：

[0186] Return运算符::＝

[0187] Return表达式

[0188] Select运算符::=Select Select声明列表

[0189] Select声明列表::=

[0190] Select声明|

[0191] Select声明列表,Select声明

[0192] Select声明::=

[0193] 变量声明函数

[0194] 查询运算符模式306可以包括一查询运算符,其是名为WHERE的可访问实例方法,该方法取一函数作为自变量。类型T可以是该函数的自变量,并且结果可以是可被隐式地转换成布尔类型的类型。例如,结果类型可以具有与可在IF、WHILE或DO语句中使用的表达式的类型相同的规则。因此,类型可以具有到布尔类型的隐式转换,或者需要已经定义了IsTrue和IsFalse运算符。WHERE运算符可以返回具有元素类型T的可查询类型。

[0195] WHERE运算符可以将类集中的值限于满足给定条件的那些值。WHERE运算符可以接收对每一组控制变量值求值的布尔表达式。如果表达式的值为真,则该值可出现在输出类集中,否则该值可被跳过。查询表达式:

[0196] From b In Books,p In Publishers_

[0197] Where b.PublisherID=p.PublisherID_

[0198] 可被认为等价于嵌套循环

[0199] For Each b In Books

[0200] For Each p In Publishers

[0201] If b.PublisherID=p.PublisherID Then

[0202] End If

[0203] Next p

[0204] Next b

[0205] WHERE运算符可被定义为:

[0206] Where运算符::=Where布尔表达式

[0207] 查询运算符模式308可以包括一查询运算符,其是名为ORDERBY或ORDERBYDESCENDING(按降序排序)的可访问实例方法,该方法可取一函数作为自变量,该函数进而可取类型T的自变量,并产生类型S。与模式308相关联的方法可以返回具有元素类型T的已排序类集。

[0208] ORDERBY运算符可以基于一次序对出现在控制变量中的值排序。ORDERBY运算符可以取指定应使用来对控制变量排序的值的表达式。例如,以下查询返回按照价格(Price)排序的书名(book.Title):

[0209] From book In Books_

[0210] Order By book.Price

[0211] Select book.Title

[0212] 排序可以是升序的,在这一情况下较小的值可以在较大的值之前出现;或者排序可以是降序的,在这一情况下较大的值在较小的值之前出现。排序的默认值是升序。例如,以下查询返回按照价格排序的书名,且最贵的书在最前面(Descending):

[0213] From book In Books_

[0214] Order By book.Price Descending

[0215] Select book.Title

[0216] ORDERBY运算符还可指定用于排序的多个表达式，在这一情况下类集可以用嵌套方式来排序。例如，以下查询表达式按照州(State)，然后按照每一州内的市(City)，在按照每一市内的邮政编码(ZIP)，来对作者(Author)排序：

[0217] From author In Authors_

[0218] Order By author.State,author.City,author.ZIP_

[0219] Select author.Name,author.State,author.City,author.ZIP

[0220] ORDERBY运算符可以基于以下示例来架构化：

[0221] OrderBy运算符::=Order By Order表达式列表

[0222] Order表达式列表::=

[0223] Order表达式|

[0224] Order表达式列表,Order表达式

[0225] Order表达式::=

[0226] 表达式[排序]

[0227] 排序::=Ascending|Descending

[0228] 另外，查询运算符模式310可以包括一查询运算符，其是名为DISTINCT的为可访问实例方法。通常，该方法返回具有与源类型C相同的元素类型的可查询类型。DISTINCT运算符可以将类集中的值仅限于具有不同值的那些(例如，不返回重复值)。例如，查询：

[0229] From c In Customers,o In c.Orders_

[0230] Select c.Name,o.Price_

[0231] Distinct

[0232] 将仅对每一对不同的顾客名和定单(Order)价格(Price)返回一行，即使该顾客具有价格相同的多个定单。DISTINCT运算符可被架构化为：

[0233] Distinct运算符::=Distinct

[0234] 现在参考查询运算符模式312，模式312可以包括一查询运算符，其是名为CONCAT、UNION、INTERSECT或EXCEPT的可访问实例方法。该方法可以接收具有与源C相同的元素类型T的类集作为自变量，并返回具有元素类型T的可查询类型。

[0235] 下一查询运算符模式，即模式314可以包括一查询运算符，其是名为TAKE或SKIP的可访问实例方法，该方法可以接收值作为自变量，并且可以返回具有与源类型C相同的元素类型的可查询类型。

[0236] TAKE运算符可以产生来自类集的给定数量的元素。当随WHILE修饰符一起使用时(参见例如查询运算符模式316)，TAKE运算符可以在条件保持为真时产生一元素序列。

[0237] SKIP运算符可以忽略来自类集的给定数量的元素，然后返回该类集的其余元素。当结合WHILE修饰符一起使用时，SKIP运算符在条件保持为真时跳过元素，然后返回类集的剩余元素。TAKE和SKIP的一个示例架构如下提供。

[0238] Partition运算符::=

[0239] Take表达式|

[0240] Skip表达式

[0241] Take表达式::=

[0242] Take[While]表达式|

[0243] Skip表达式::=

[0244] Skip[While]表达式

[0245] 查询运算符模式316可以包括一查询运算符，其是名为TAKEWHILE或SKIPWHILE的可访问实例方法。该方法可取一函数作为自变量，其中该函数可接收类型T的自变量，并产生布尔值。该方法可以返回具有元素类型T的可查询类型。

[0246] 查询运算符模式318可以包括一查询运算符，其是名为SUM、MIN、MAZ、COUNT或AVERAGE的可访问实例方法。该方法可取一函数作为自变量，其中该函数进而接收类型T的自变量，并产生数值类型。该方法可返回数值类型。

[0247] 查询运算符模式320可以包括一查询运算符，其是名为MIN或MAX的可访问实例方法，该方法可取一函数作为自变量，该函数进而可取类型T的自变量，并产生类型S。该方法可以返回具有元素类型S的可查询类型。

[0248] 查询运算符模式322可以包括一查询运算符，其是名为ANY或ALL的可访问实例方法，该方法可以接收一函数作为自变量。函数可以包括类型为T的自变量，并产生可被隐式地转换成布尔值的类型。该方法可返回可被隐式地转换成布尔值的类型。

[0249] 可以理解，与模式318-322相关联的查询运算符可以结合AGGREGATE(聚集)运算符来使用。在聚集查询子句内，标准AGGREGATE运算符集可被应用于范围中的分组的控制变量。AGGREGATE运算符可括但不限于：ANY、ALL、COUNT、LONGCOUNT(长计数)、SUM、MIN、MAX、AVERAGE(平均)和/或GROUP。

[0250] ANY聚集运算符可以查明在组中是否有满足给定条件的元素。ALL聚集运算符可以确定是否组中的全部元素都满足给定条件。例如，以下的示例查询表达式104可以检查是否有低于18岁的任何顾客(cust.Age＜18)：

[0251] From cust In Customers_

[0252] Group By cust.State_

[0253] Aggregate YoungerThan18＝Any(cust.Age＜18)

[0254] 而以下示例查询表达式104可以返回给定州中(cus.State)具有至少5个定单的所有顾客(cust.Orders.Count()＞5)：

[0255] From cust In Customers_

[0256] Group By cust.State_

[0257] Aggregate AtLeast5＝All(cust.Orders.Count()＞5)

[0258] COUNT和LONGCOUNT聚集运算符可以取可任选布尔表达式，并对该组中满足给定条件的元素的数量计数。

[0259] From cust In Customers_

[0260] Group By cust.State_

[0261] Aggregate Seniors＝Count(cust.Age＞50)

[0262] SUM聚集运算符可以基于特定选择器表达式来计算组中的元素之和。例如，以下示例性查询表达式104可以计算按照类别(Category)分组的所有定单的总值：

[0263] From order In Orders_

[0264] Group By order.Category

[0265] Aggregate Total＝Sum(order.Price)

[0266] MIN和MAX聚集运算符可以基于某一选择器表达式来计算组的元素的最小值(或最大值)。例如,以下示例查询表达式104可以计算每一州的最年轻(Youngest)和最年长(Oldest)的顾客:

[0267] From cust In Customers_

[0268] Group By cust.State

[0269] Aggregate Oldest＝Max(cust.Age),Youngest＝Min(cust.Age)

[0270] AVERAGE聚集运算符通常基于特定选择器表达式来计算组中的元素的平均值。例如,以下示例查询表达式104可以计算按照类别分组的所有产品(Products)的平均价格:

[0271] From prod In Products_

[0272] Group By prod.Category

[0273] Aggregate AveragePrice＝Average(prod.Price)

[0274] 特殊的GROUP聚集运算符可以基于可任选选择器表达式将组的所有元素累积成显式类集。例如,以下示例性查询表达式104可以将给定州中的所有顾客姓名收集到单个类集中。

[0275] From cust In Customers_

[0276] Group By cust.State

[0277] Aggregate Names＝Group(cust.Name)

[0278] 查询运算符模式324可以包括一查询运算符,其是名为JOIN的可访问实例方法。该方法可以取以下各项作为自变量:具有类型T′的元素的可查询类型S;用作内部选择器的函数,其可取类型T的自变量,并产生表示关键字的类型K;用作外部选择器的函数,其可取类型T′作为自变量,并产生表示关键字的类型K;和/或用作联接条件的函数,其可取类型分别为T和T′的两个自变量,并产生类型V的散列值。该方法可以返回具有元素类型V的可查询类型。

[0279] 因此,JOIN运算符可以取两个类集,并可以基于从元素中导出的匹配关键字来产生单个类集。在指定要联接的条件时可以应用某些限制。例如,可以要求对于条件表达式的运算数可被显示地转换为布尔值。JOIN架构的一个示例如下。

[0280] Join运算符::＝

[0281] Join循环控制变量[As类型名]In表达式Where Join条件表达式

[0282] Join条件表达式::＝

[0283] Relational运算符表达式|

[0284] Like运算符表达式

[0285] 查询运算符模式326可以包括一查询运算符,其是名为GROUPJOIN的可访问实例方法,该方法可以接收以下各项作为自变量:具有类型T′的元素的可查询类型S;用作外部关键字选择器的函数,其取类型T的自变量,并产生表示关键字的类型K;用作内部关键字选择器的函数,其取类型U的自变量,并产生表示关键字的类型K;和/或可产生结果的函数,其中该函数可以取类型T的自变量,以及具有元素类型U的类集的自变量,并产生所选类型V。该

方法可以返回具有元素类型V的可查询类型。

[0286] GROUPJOIN运算符可以基于从元素中提取的匹配关键字来创建两个类集的分组联接。运算符可以产生分层结果(例如,与匹配的内部元素的类集配对的外部元素),并且不需要关系型数据库项中的直接等价物。以下示例查询表达式104可以执行顾客与其定单的分组联接,从而产生具有顾客姓名以及该顾客的定单总额(order.Total)的匿名类型的类集:

[0287] From cust In db.Customers_

[0288] Group Join order In db.Orders On cust.ID=order.CustID_

[0289] Aggregate TotalOrders=Sum(order.Total)_

[0290] Select cust.Name,TotalOrders

[0291] 可以有对固定集合操作以计算单个值的任何数量的附加运算符。在查询的上下文中,可以认为这些运算符聚集所计算的值。为对集合进行聚集,该集合一般必须首先被引入。在查询内,可以有各种方式来指定用于聚集计算的类集。在引入了一个组后,该组可被聚集以计算单个值。因此,除了分组运算符GROUPJOIN(以及以下结合查询运算符模式328讨论的GROUPBY)之外,可存在其它分组运算符。一个这样的示例是OVER运算符。

[0292] OVER运算符可以指定一个组应被累积以便进行聚集综合。OVER运算符可用作独立表达式,或者其可用于指定应对聚集计算采用预形成的类集。例如,以下示例查询表达式104可以聚集2006年1月1日以前所有定单总额之和:

[0293] Over order In Orders_

[0294] Where order.Date<=#01/01/2006#_

[0295] Aggregate Sum(order.Total)

[0296] OVER运算符的每次应用通常必须以相应的AGGREGATE运算符关闭来基于类集内容计算单个值。查询的结果可以是对应于聚集计算的单个值,在这一情况下其是表示2006年以前的所有定单的总金额的整数。

[0297] 在OVER和AGGREGATE运算符之间可以使用可能任何查询运算符。在AGGREGATE运算符内,仅可对前一OVER运算符指定的控制变量使用AGGREGATE运算符(例如,与模式318-322相关联的运算符,如上所述)。

[0298] 当已经形成一个集合时,如通常是分层对象图的情形,则可使用OVER运算符来指定应对聚集计算使用一个集合。例如,以下示例性查询表达式104可以聚集2006年以前华盛顿的所有顾客发出的所有定单的总金额:

[0299] From cust In db.Customers_

[0300] Where cust.State="WA"_

[0301] Over order In Orders_

[0302] Where order.Date<=#01/01/2006#_

[0303] Aggregate Sum(order.Total)

[0304] 该查询的结果可以是其元素类型是具有以下两个属性的匿名类型的可查询类型:1)称为cust的顾客属性,和称为Sum的整数属性。

[0305] 查询运算符模式328可以包括一查询运算符,其是名为GROUPBY的可访问实例方法,该方法可取以下各项作为自变量:用作关键字选择器的函数,其接收类型T的自变量,并产生表示关键字的类型K;和/或产生结果的函数,其取类型K的自变量以及作为具有元素类

型T的类集的自变量，并产生所选类型V。该方法可以返回产生具有元素类型V的可查询类型。

[0306] GROUPBY运算符可以基于一个或多个共同的关键字表达式来对类集的元素分组(例如，逻辑上)。对于初始类集的这些分区的每一个，后续的AGGREGATE语句可以指定这些组的每一个如何被聚集成单个值或行。

[0307] 例如，以下示例性查询表达式104可以按照州来对所有顾客分组，然后计算每一组的计数(Count)和平均年龄(AverageAge)：

[0308] From cust In Customers_

[0309] Group By cust.State

[0310] Aggregate Total＝Count(),AverageAge＝Avg(cust.Age)

[0311] 与SELECT运算符一样，GROUPBY运算符可以将在关键字选择器中声明的变量作为控制变量带入范围中，并且可隐藏先前在范围中的所有控制变量。相反，与SELECT运算符的某些方面不同，这些隐藏的控制变量可以由在后续的AGGREGATE运算符内使用的AGGREGATE运算符再次带入范围中。此外，与OVER运算符的某些实现不同，在GROUPBY和AGGREGATE运算符之间不能使用任何其它查询运算符。

[0312] From cust In Customers_

[0313] Group CustomersByState By cust.State

[0314] 这一构造可以等价于通过显式聚集来构造组，但是在GROUPBY运算符子句之后立即终止查询时是有用的。由此，以上表达式可以等价于以下表达式：

[0315] From cust In Customers_

[0316] Group By cust.State_

[0317] Aggregate CustomersByState＝Group(cust)

[0318] 不必要求GROUPBY运算符的实际实现实际构建每一个别的组的表示。相反，该实现可使用将分组与后续的聚集成单个运算相组合的底层实现。

[0319] 可以理解，控制变量的类型和名称可以从查询表达式104的目标以及所应用的查询运算符中推断(将在下文中结合图4来更详细讨论)。由此，元素类型(其是范围中的控制变量的聚集)可以用类似的方式来推断。对于以上的每一条规则，元素类型的所推断的类型可以是T，并且当可查询类型C是在例如FOR EACH...NEXT语句中使用的类集时，元素类型通常必须匹配元素类型T。

[0320] 还应当理解，已排序类集可被定义为实现以下运算符的子集的类集：名为THENBY或THENBYDESCENDING的、取一函数作为自变量的可访问实例方法，该函数取类型T的自变量，并产生类型S。该方法可以返回具有元素类型T的已排序类集。

[0321] 由于查询运算符模式106可以方便将查询句法绑定到实现特定查询运算的方法，因此不需要由所采用的底层语言来规定次序保留。相反，次序保留可以由运算符本身的实现来确定。这在要重载的实现方面可以类似于用户定义的运算符，例如，用于用户定义的数值类型的加法运算符不能执行类似加法的任何运算。然而，为保留所要求保护的主题内在的可预测性，实现不匹配用户期望的运算符可能不是推荐的过程。

[0322] 为帮助更完整地理解所要求保护的主题，现在可进一步讨论其它特征、方面和/或实现。例如，以下的某些节详述了每一查询运算符映射到对标准查询运算符方法的调用的

示例性方式。其它节描述了如上引入的元素类型信息流的方式，并且将在稍后的节中使用(例如，结合图4和以下相关联的描述)。

[0323] 查询运算符转换

[0324] 可以理解，各种查询运算符的每一个可以直接映射到可根据标准查询运算符模式(例如，查询运算符模式106)来定义的方法调用。由此，查询运算符(或作为整体的查询表达式)的含义可以按照要调用的查询运算符方法来表达。方法可以是所查询的对象的实例方法，或对象外部的扩展方法。例如，查询：

[0325] Dim names＝_

[0326] From c In Customers_

[0327] Where c.State＝"WA"_

[0328] Return c.Name

[0329] 可以等价于：

[0330] Dim names＝Customers.Where(c＝＞c.State).Select(c＝＞c.Name)

[0331] 合成控制变量

[0332] 尽管控制变量可由所要求保护的主题为此处所描述的各种目的而采用，但是可以理解，控制变量一般不是对最终用户直接可访问的。诸如WHERE或SELECT等某些查询运算符取可以引用查询中在范围中的控制变量的表达式。这些表达式被表达为取控制变量并返回表达式结果的局部函数。例如，查询：

[0333] Dim WACusts＝_

[0334] From cust In Customers_

[0335] Where cust.State＝"WA"

[0336] 可以等价于：

[0337] Dim WACusts＝Customers.Where(c＝＞c.State＝“WA”)_

[0338] .Select(c＝＞New With{c.Name})

[0339] 其中控制变量cust用作用于拉姆达表达式(lambda expression)的参数。类似地，在多个控制变量在范围中的情况下，控制变量通常必须被一起分组成单个匿名类型，以使它们可被传入局部函数。例如，查询：

[0340] Dim WACustsAndOrders＝_

[0341] From cust In Customers_

[0342] From order In cust.Orders_

[0343] Where cust.State＝"WA"AndAlso order.Price＞10.50

[0344] 可以等价于：

[0345] Function Filter1(it As{c As Customer,o As Order})As Boolean

[0346] Return it.c.State＝"WA"AndAlso it.o.Price＞10.50

[0347] End Function

[0348] Function Join1(c As Customer)As_

[0349] IEnumerable(Of{c As Customer,o As Order})

[0350] Function Select1(o As Order)As{c As Customer,o As Order}

[0351] Return New{c,o}

[0352] End Function

[0353] Return c.Orders.Select(AddressOf Select1)

[0354] End Function

[0355] Dim WACustsAndOrders＝_

[0356] Customers.SelectMany(AddressOf Join1).Where(AddressOf

[0357] Filter1)

[0358] 其中控制变量c和o被一起组合成称为IT的合成控制变量，其类型是匿名类型{c As Customer,o As Order}。

[0359] 还可以理解，也可以提供对于拉姆达表达式的支持，并且其被认为是在所要求保护的主题的精神和范围之内。还要注意，合成控制变量通常是不嵌套的。作为一个示例：

[0360] From b In Books_

[0361] From ba In BookAuthors_

[0362] From a In Authors_

[0363] 可以等价于：

[0364] Function Join1(b As Book)As IEnumerable(Of{b As Book,ba As

[0365] BookAuthor})

[0366] Function Select1(ba As BookAuthor)As{b As Book,ba As

[0367] BookAuthor}

[0368] Return New{b,ba}

[0369] End Function

[0370] Return BookAuthors.Select(AddressOf Select1)

[0371] End Function

[0372] Function Join2(it As{b As Book,ba As BookAuthor})As_

[0373] IEnumerable(Of{b As Book,ba As BookAuthor,a As Author})

[0374] Function Select2(a As Author)As_

[0375] {b As Book,ba As BookAuthor,a As Author}

[0376] Return New{b,ba,a}

[0377] End Function

[0378] Return Authors.Select(AddressOf Join2)

[0379] End Function

[0380] Books.SelectMany(AddressOf Join1).SelectMany(AddresOf Join2)

[0381] FROM运算符转换

[0382] 标准FROM运算符一般仅引入查询源以及要使用的控制变量。本质上，它不转换成任何特定查询运算符调用。在JOIN的情况下，FROM运算符在与常规控制变量联接时可被转换成对SELECTMANY查询运算符方法的调用，而在与表达式控制变量联接时可被转换成对SELECT查询运算符方法的调用。因此，查询表达式：

[0383] From b In Books,p In Publishers_

[0384] Select b.Title,p.Name

[0385] 可以等价于：

```
[0386] Function Join1(b As Book)As IEnumerable(Of{b As Book,p As
[0387] Publisher})
[0388]  Function Select2(p As Publisher)As{b As Book,p As Publisher}
[0389]     Return New With{b,p}
[0390]  End Function
[0391]  Return Publishers.Select(AddressOf Select2)
[0392] End Function
[0393] Function Select1(it As{b As Book,p As Publisher})As_
[0394]     {Title As String,Name As String}
[0395]  Return New With{it.b.Title,it.p.Name}
[0396] End Function
[0397] Books.SelectMany(AddressOf Join1).Select(AddressOf Select1)
```

[0398] 并且查询表达式：

```
[0399] From b In Books,Tax＝b.Price*0.088_
[0400] Select b.Title,Tax
```

[0401] 可以等价于：

```
[0402] Function Select1(b As Book)As{b As Book,Tax As Double}
[0403]  Return New With{b,Tax＝b.Price*0.088}
[0404] End Function
[0405] Function Select2(it As{b As Book,Tax As Double})As{Title As
[0406] String,Tax As Double}
[0407]  Return New With{it.b.Title,it.Tax}
[0408] End Function
[0409] Books.Select(AddressOf Select1).Select(AddressOf Select2)
```

[0410] 每一查询运算符可被绑定到粘附到特定模式的源类型上的底层方法。该模式规定了运算符的结果的元素类型，并且在某些情况下对可传入底层方法的表达式施加了约束。

[0411] 显式类型化的控制变量

[0412] 如果FROM运算符使用As子句为控制变量指定了目标类型Object(对象)，并且T的类型不能从源类型中推断，则元素类型可以使用Case(Of Object)运算符来转换。查询表达式：

```
[0413] Dim nums＝_
[0414]  From addrNum As Object In publisher.Address_
[0415]  Where addrNum＜5
```

[0416] 等价于：

```
[0417] Function Filter1(addrNum As Integer)As Boolean
[0418]  Return addrNum＜5
[0419] End Function
[0420] Dim names＝publisher.Address.Cast(Of Object)().Where(AddressOf
[0421] Filter1)
```

[0422] 如果FROM运算符使用As子句为控制变量指定了目标类型T(不是Object),并且T的类型不能从源类型中推断或者如果目标类型不匹配源的元素类型,则元素类型可以使用Cast(强制转换)查询运算符转换成Object,并且可利用后续的SELECT运算符来获得目标类型。例如,查询表达式:

[0423] Dim nums＝_

[0424] From addrNum As Integer In publisher.Address_

[0425] Where addrNum＜5

[0426] 等价于:

[0427] Function Filter1(addrNum As Integer)As Boolean

[0428] Return addrNum＜5

[0429] End Function

[0430] Function Select1(addrNum As Integer)As String

[0431] Return addrNum

[0432] End Function

[0433] Dim names＝publisher.Address.Cast(Of Object)().Select(AddressOf

[0434] Select1).Where(AddressOf Filter1)

[0435] 如果FROM运算符使用As子句为控制变量指定了目标类型T,并且T的类型匹配源的元素类型,则不需要后续应用Cast或SELECT运算符。

[0436] WHERE运算符转换

[0437] WHERE运算符可被转换成对Where查询运算符方法的调用。例如,查询表达式:

[0438] From book In Books_

[0439] Where book.PublisherID＝10_

[0440] Return book.Title

[0441] 可以等价于:

[0442] Books.Where(b＝＞b.PublisherID＝10).Select(b＝＞b.Title)

[0443] ORDERBY运算符转换

[0444] ORDERBY运算符对于第一次排序可被转换成对ORDERBY或ORDERBYDESCENDING查询运算符方法的调用(取决于排序的类型),并且然后对于以后的排序被转换成对THENBY和THENBYDESCENDING查询运算符方法的调用。例如,查询表达式:

[0445] From a In Authors_

[0446] Order By a.State,a.City Descending_

[0447] Return a.Name

[0448] 可以等价于:

[0449] Authors.OrderBy(a＝＞a.State).ThenByDescending(a＝＞a.City)._

[0450] Select(a＝＞a.Name)

[0451] RETURN和SELECT运算符转换

[0452] RETURN运算符可被转换成对SELECT查询运算符方法的调用。例如:

[0453] From e In Employees_

[0454] Return e.Name

[0455] 可以等价于：

[0456] Employees.Select(e＝＞e.Name)

[0457] SELECT运算符可被转换成对SELECT查询运算符方法的调用以及匿名类型的构造。例如：

[0458] From e In Employees_

[0459] Select e.Name,e.Salary

[0460] 可以等价于：

[0461] Employees.Select(e＝＞New With{e.Name,e.Salary})

[0462] DISTINCT运算符转换

[0463] DISTINCT运算符可被转换成对DISTINCT查询运算符方法的调用。例如：

[0464] From e In Employees_

[0465] Distinct

[0466] 可以等价于：

[0467] Employees.Distinct()

[0468] TAKE或SKIP[WHILE]运算符转换

[0469] TAKE或SKIP运算符可以调用相应的TAKE或SKIP查询运算符方法。

[0470] 例如：

[0471] From e In Employees_

[0472] Order By e.Salary Descending_

[0473] Return e.Name

[0474] Take 5

[0475] 可以等价于：

[0476] Employees.OrderByDescending(e＝＞e.Salary).Select(e＝＞e.Name).Take(5)

[0477] 并且查询：

[0478] From e In Employees_

[0479] Order By e.Salary Descending_

[0480] Return e.Name

[0481] Skip 5

[0482] 可以等价于：

[0483] Employees.OrderByDescending(e＝＞e.Salary).Select(e＝＞e.Name).Skip(5)

[0484] 当结合WHILE修饰符使用时，可以创建条件表达式来应用于底层的TAKEWHILE或SKIPWHILE查询运算符。例如，查询：

[0485] From e In Employees_

[0486] Order By e.Salary Descending_

[0487] Take While e.Salary＞50000_

[0488] Return e.Name

[0489] 可以等价于：

[0490] Employees.OrderByDescending(e＝＞e.Salary)._

[0491] .TakeWhile(e＝＞e.Salary＞50000).Select(e＝＞e.Name)

[0492] GROUPBY和AGGREGATE运算符转换

[0493] 后跟聚集SELECT或RETURN运算符的GROUPBY运算符可被转换成对后跟构造的组上的聚集的GROUPBY查询运算符方法的调用。例如，查询：

[0494] From C In Customers_

[0495] Group By C.State_

[0496] Aggregate Youngest＝Min(c.Age),Oldest＝Max(c.Age)

[0497] Select State,Youngest,Oldest

[0498] 可以等价于：

[0499] Customers

[0500] .GroupBy(c＝＞New With{.State＝c.State},_

[0501] (g,k)＝＞New With{.State＝k,_

[0502] .Youngest＝g.Min(c＝＞c.Age),

[0503] .Oldest＝g.Max(c＝＞c.Age)})

[0504] 示例对象

[0505] 此处包括的所有示例使用两个单独的对象集中的一个。Company(公司)对象可以表示关于一公司的信息，并且可以分层地连接：

[0506] Class Company

[0507] Dim Name As String

[0508] Dim Employees As List(Of Employees)

[0509] Dim Customers As List(Of Customer)

[0510] End Class

[0511] Class Customer

[0512] Dim Name As String

[0513] Dim Orders As List(Of Order)

[0514] End Class

[0515] Class Order

[0516] Dim Product As String

[0517] Dim Price As Integer

[0518] End Class

[0519] Class Employee

[0520] Dim Name As String

[0521] Dim Birthday As Date

[0522] Dim Position As String

[0523] Dim Salary As Decimal

[0524] End Class

[0525] Dim Employees As List(Of Employee)

[0526] Dim Customers As List(Of Customer)

[0527] Book(书)对象可以表示关于例如书店中库存的书的信息，并且可通过关系连接：

[0528] Class Book

[0529] Dim BookID As Long

[0530] Dim Title As String

[0531] Dim PublisherID As Long

[0532] Dim Price As Double

[0533] End Class

[0534] Class Author

[0535] Dim AuthorID As Long

[0536] Dim Name As String

[0537] Dim Address As String

[0538] Dim City As String

[0539] Dim State As String

[0540] Dim ZIP As String

[0541] End Class

[0542] Class BookAuthor

[0543] Dim BookID As Long

[0544] Dim AuthorID As Long

[0545] End Class

[0546] Class Publisher

[0547] Dim PublisherID As Long

[0548] Dim Name As String

[0549] Dim Address As String

[0550] Dim State As String

[0551] End Class

[0552] Dim Books As List(Of Book)

[0553] Dim Authors As List(Of Author)

[0554] Dim BookAuthors As List(Of BookAuthor)

[0555] Dim Publishers As List(Of Publisher)

[0556] 在记住了以上内容之后，现在转向图4，描绘了可以实时地推断元素类型和/或基于所推断的元素类型增量式地提供上下文信息的计算机实现的系统400。一般而言，系统400可以包括接收查询表达式104的用户接口102，如基本在上文结合图1所描述的。另外，系统400可以包括可操作地耦合(未示出)到用户接口102和/或上下文组件402的模式存储108。上下文组件402可以检查查询表达式104并实时地做出关于元素类型的推断。例如，如上所述，由于用于每一查询子句的查询运算符符合一特定模式(例如，图1的查询运算符模式106)，并且此外，由于元素类型可从一个查询子句流到下一查询子句，因此这两条信息可用于为任何给定下一查询子句确定元素类型。

[0557] 当查询运算符被描述为方法调用并且查询运算作为纯机制变换来应用时，一般要求类型解析在重载解析期间进行。以此方式，假定查询运算符可将类型从一个运算符流到

另一个，则运算符一般必须被完全绑定以便能够将类型提供给要应用的下一运算符。结果，如果重载解析由于句法错误或某一其它问题而失败，则没有可用于向应用的下一运算符提供类型信息的常规机制。没有这一类型流机制的合成查询运算符(以下从图6开始更详细讨论)导致多余的错误消息、不同的表达式语义、缺少用于自动完成的上下文信息、以及总体上更慢的编译器吞吐量。

[0558] 如所描述的，所要求保护的主题可以采用查询运算符的模式的形式化，这可允许元素类型不依赖于完全方法绑定，而是依赖于所采用的特定运算符的模式。作为进一步说明，考虑由用户接口102接收的当前示例性查询表达式104(例如，在类型化查询表达式104时)：

[0559] Dim q＝From cust In Customers_

[0560] Where cust.Name＝“XYZ”& errorUnresolvedVarName_

[0561] Select cust.

[0562] 第一个查询子句包括FROM运算符，其可引入可查询源类型“Customers”作为匿名元素类型“cust”，这可以是源类型的元素。如所讨论的，该元素类型可以流到下一查询子句，其中WHERE运算符可以过滤该类集，但是在其它方面不更改元素类型。最后，该元素类型流到下一查询子句，其中可找到SELECT运算符以及SELECT运算符方法期望接收的函数的一部分，但是下一查询子句的其余部分尚未完成。

[0563] 流入SELECT子句的元素类型现在可用于该特定查询子句的源类型。上下文组件402因此可以基于源类型和查询运算符来实时地推断该查询子句的元素类型(例如，cust)。因此，推断可以完全是局部的，并且不需要对查询表达式104的完全转换。相反，上下文组件402可以用类似于后台编译器的方式来执行推断。

[0564] 另外，上下文组件402的推断可以用多种方式来促进更大的效率。例如，推断仅需在执行转换时被执行一次。由此，如果在查询表达式104中有两个WHERE运算符，则对第二个WHERE运算符的转换不必是必需的。另外，推断可以用于完全和通常更昂贵的编译，以缩短一般必须执行的类型发现的巡回。由此，事先推断元素类型可以用各种方式来使用以加快编译时间。

[0565] 此外，一旦上下文组件402推断了元素类型(此处，是“Select cust.”查询子句中的“cust”)，则上下文组件402可以增量式地提供上下文信息404。例如，当点的类型是“cust.”时，则上下文信息404可以用方便且熟悉的弹出窗口的形式来动态地提供，以允许如图所示或另一形式的自动完成。可以理解，这一特征常规上对查询表达式不可用，因为查询表达式(常规上)在可确定元素类型之前必须首先被转换。并且，由于提供完整且无错误的查询表达式是常规转换/编译的先决条件，因此迄今为止阻碍了任何类似的上下文和/或自动完成机制。

[0566] 尽管上下文信息404对于查询表达式有很大的效用，并且大部分在于IDE中提供上下文反馈的上下文中示出，但是没有一种情况是必需的。例如，此处所描述的可适用于不仅仅是查询表达式的查询运算符。特别地，类型从一个运算符流到下一运算符并在两者之间有关系的概念可应用于其中类型流有用的任何运算符或任何API。一个这样的示例存在于将API调用链接在一起(例如，包括在命令行中的流水线，其中目的是类型从一个命令流到下一个)。另一示例可以是后绑定，其中没有类型，但是可采用某种形式的流诸如用于后绑

定的优化。

[0567] 还可以理解，上下文组件402所执行的推断可以完全基于所接收到的已知信息来预定(例如，关于已知且定义的查询运算符的源类型)，或者根据其它方面，推断可以是概率性的。例如，当用于不同类型的运算符时，上下文组件402可用于检查可用数据的整体或其子集，并且能够从经由事件和/或数据捕捉的一组观察结果中推出或推断系统、环境和/或用户的状态。例如，推断可用于标识特定的上下文或动作，或可生成状态的概率分布。推断可以是概率性的，即，基于数据和事件的考虑计算感兴趣的状态的概率分布。推断也可以指用于从一组事件和/或数据组成更高级事件的技术。

[0568] 这类推断可导致从一组观察到的事件和/或储存的事件数据中构造新的事件或动作，而无论事件是否在相邻时间上相关，也无论事件和数据是来自一个还是若干个事件和数据源。可采用各种分类(显式和/或隐式训练的)方案和/或系统(例如，支持向量机、神经网络、专家系统、贝叶斯信任网络、模糊逻辑、数据融合引擎.....)来执行关于所要求保护的主题的自动化和/或推断的动作。

[0569] 分类器可以是将输入属性矢量x＝(x1，x2，x3，x4，xn)映射到该输入属于一个类的置信度的函数，即f(x)＝confidence(class)。这一分类可采用基于概率和/或基于统计的分析(例如，分解成分析效用和成本)来预测或推断用户期望自动执行的动作。支持向量机(SVM)是可采用的分类器的一个示例。SVM通过找出可能输入空间中的超曲面来操作，其中，超曲面试图将触发准则从非触发事件中分离出来。直观上，这使得分类对于接近但不等同于训练数据的测试数据正确。可采用其它定向和非定向模型分类方法，包括，例如，朴素贝叶斯、贝叶斯网络、决策树、神经网络、模糊逻辑模型以及提供不同独立性模式的概率分类模型。此处所使用的分类也包括用于开发优先级模型的统计回归。

[0570] 图5示出了根据所要求保护的主题的方法。尽管出于简化解释的目的，该方法和其它方法被示出和描述为一系列动作，但应该理解和明白，所要求保护的主题不受动作的顺序限制，因为某些动作能够以与在此所示出和描述的不同的顺序发生和/或与其它动作同时发生。例如，本领域技术人员将会明白并理解，方法可被替换地表示为一系列相互关联的状态或事件，诸如以状态图的形式。而且，并非所有示出的动作都是实现根据所要求保护的主题的方法所必需的。另外还应该理解，下文以及本说明书全文中所公开的方法能够被存储在制品上，以便于把此类方法传送和转移到计算机。在此使用的术语“制品”意指包含可以从任何计算机可读设备、载体或介质访问的计算机程序。

[0571] 现在参考图5，示出了用于方便元素类型的类型流的计算机实现的方法。概括地，在参考标号502处，可接收包括查询运算符的查询子句的一部分。查询运算符可被映射到相关联的方法调用，该方法调用可接收自变量(例如，函数、值、类集.....)并可返回类型化的结果(例如，可查询类型、数值类型、已排序类集.....)。该方法调用可以根据查询运算符模式来定义。

[0572] 在参考标号504处，可利用源类型和查询运算符来确定元素类型。例如，由于查询运算符模式可被形式化，因此查询运算符的模式可结合源类型使用来推断元素类型。在参考标号506处，可采用前一查询的元素类型作为参考标号504处所讨论的利用动作的源类型。据此，元素类型可以按递归的方式从一个查询子句流到下一查询子句。

[0573] 在记住了上述内容之后，可以理解和明白，某些特征和/或方面可用于以其它方式

来扩展编程语言的查询能力。作为一个示例，此处所开发的方面可有效地利用来提供基于计算机和/或基于语言的构造，如与常规上与API调用相关联的合成能力的综合。这些和其它相关概念的描述可参考图6找到。

[0574] 现在转向图6，提供了方便可组成查询综合和/或可扩展查询表达式的计算机实现的系统600的框图。一般而言，系统600可包括可接收初始化数据604的变换组件602。如图所示，初始化数据604可以是示例性查询表达式606中的第一个查询子句，其中该第一个查询子句通常限于是FROM子句(例如，第一个查询子句的查询运算符是FROM运算符)。初始化数据604可以包括类集(例如，如图所示的Customers)或表达式(例如，如上文结合关于图3详述的FROM运算符描述的Tax＝b.Price*0.088)。在任一情况下，初始化数据还可包括控制变量，取决于应用，该控制变量与类集(例如，C是控制变量)或表达式(例如，Tax是控制变量)相关联。

[0575] 变换组件602还可接收由查询表达式606表征的序列中的一组查询子句608。可以理解，该组查询子句608可以是空集，在这一情况下，查询表达式606由单个查询子句构成(例如，用作初始化数据604的FROM子句)，如将在下文中更详细描述的。如可以对初始数据604所完成的那样，变换组件602还可对组608中的每一查询子句解析(例如，填充或变换)范围中的控制变量。例如，变换组件602可以进而通过向可用类型应用每一查询子句的查询运算符，来以查询表达式608所规定的次序处理组608中的每一查询子句。

[0576] 可以理解，根据所要求保护的主题，该组查询子句608不需要由诸如，例如XQuery语言的各种版本所定义的FLWOR/FLWR或SQL的实现中所期望的Select-From-Where等普遍存在的语义模板来限制。相反，组608可以在本质上是合成的，使得个别查询子句可被妥当地提供、以模块化方式接合在一起、并且即使在运算符的次序是任意的情况下也产生直观结果。

[0577] 据此，系统600还可包括可管理查询表达式606所引入的所有控制变量的范围的综合组件610。控制变量的范围可以基于查询子句的运算符来确定，并且范围中的控制变量可被传递到下一查询子句。例如，变换组件602可以确立可被认为是流水线的事物，使得某些信息可通过流水线从一个查询子句流到下一查询子句。由此，变换组件可以接收关于查询子句的类集，基于与该查询子句相关联的运算符来变换(例如，过滤、投影.....)该类集，并将所变换的类集输出到流水线以使其可用于下一查询子句。

[0578] 类似地，范围中的控制变量也可被传递到下一查询子句，并且精确而言什么控制变量在范围中可以由综合组件610基于所接收到的查询子句的类型来定义。通常，下一查询子句只能访问在范围中的变量。该特征以及其它特征可通过除了图6之外再参考图7来进一步描述。

[0579] 图7是涉及如可由综合组件610所确定的示例查询表达式606的控制变量的范围的示例性图示。根据所要求保护的主题的一方面，综合组件610可以基于查询子句/运算符的类型来声明新的控制变量(并且可将该新的控制变量带入范围)。例如，某些类型的查询子句可以除了已经在范围中的控制变量之外还引入新的控制变量，即，FROM、LET、SELECT或GROUPBY查询子句可产生这一结果。由FROM子句引入的控制变量通常被累积，直到下一SELECT或GROUPBY子句，在这之后它们可由综合组件610带出范围。这一情况由FROM子句702、704(映射到初始化数据604以及包括在查询子句组608中的初始查询子句)示出，其控

制变量停留在范围中，直到应用了SELECT子句710。

[0580] 子句702引入Customer类集和控制变量C。因此，该信息可以通过流水线提供给下一查询子句。因此，可以使{C As Customer}对子句704可用。子句704也是FROM子句，这由于所要求保护的主题的合成本质是可能的。子句704采用了控制变量C来引入新的类集C.Orders(例如，所有顾客的所有定单)以及新的控制变量O，综合组件610可将该新的控制变量带入范围。由此，下一查询子句706能够访问范围中的两个控制变量C和O。

[0581] 基本如以上所描述的，尽管某些查询子句类型可影响范围中的变量，但是其它查询子句类型(例如，WHERE、ORDERBY、DISTINCT等)却不必。由此，尽管查询子句706和708可影响类集，但范围中的变量(例如C和O)可以相同，并且因此可分别由查询子句708和710在其各自由变换组件602接收时访问。

[0582] 根据所要求保护的主题的一方面，综合组件610可以基于所接收到的查询子句的类型将现有控制变量带出范围。SELECT查询子句710提供了该特征的表征图示。查询子句710被展示给控制变量C和O，但是SELECT子句的特征之一可以是隐藏范围中的当前控制变量并在范围中引入新的控制变量(例如，Name As String和Price As Int)。可以向查询子句712提供这些新的控制变量Name和Price，但是查询子句712一般不能访问综合组件610关于SELECT子句710带出范围的先前的控制变量(例如，C和O)。为完整性起见，应注意几个特征。首先，查询子句710示出了逗号分割的复合运算数的又一示例。

[0583] 要注意的第二个特征是WHERE子句712在SELECT子句710之后。常规的查询语言在应用了具有诸如SELECT、RETURN等投影运算符或类似的运算符的查询子句之后终止查询。如果这一常规查询语言的用户希望在投影之后继续查询运算，则必须实现可引用前一查询的结果的新查询。然而，由于例如所要求保护的主题的合成本质，查询表达式606可以包括多个SELECT子句，并且查询表达式606可以在出现SELECT子句之后继续(例如，由子句712示出)。类似地，尽管在前一节中提到，但应指出，查询表达式606也包括两个FROM子句(例如，子句702和704)，这是常规查询语言所无法提供的另一方面。

[0584] 尽管未由解释示例描绘，但查询表达式606还可包括其它查询子句类型，并且特别地，这些其它查询子句类型中的许多可影响控制变量的范围。例如，在接收到RETURN子句之后，综合组件610可以将所有变量带出范围。作为另一示例，基于查询子句的类型，综合组件610可以声明新的控制变量并对其定范围(例如，带入范围)，同时为特定一组后续查询子句确定现存的控制变量(例如，对于前一查询子句在范围中的控制变量)的范围。GROUPBY、AGGREGATE等可方便这一行为，由此现存控制变量的范围可被定在跨每一GROUP运算符的AGGREGATE查询子句的范围上。

[0585] 根据所要求保护的主题的另一方面，综合组件610可以基于所接收到的查询子句的类型为范围中的所有控制变量的元组生成别名。例如，INTO查询子句可以方便例如由综合组件610来创建别名，即使该别名本身不需要是控制变量。

[0586] 仍参考图6和7，可强调所要求保护的主题的各种附加方面。尽管有与有效查询表达式606的可扩展性有关的多个优点(例如，查询表达式606可用实际上任何数量的实际上任何类型的查询子句来扩展)，但另一优点可以是查询表达式606也可以在任何点终止。由此，尽管查询表达式606有可能包括无限数量的查询子句，但整个查询表达式606也可由仅单个在引入了单个控制变量之后终止的查询子句组成。因此，完整且有效的查询表达式606

的最简单的示例之一是仅包括初始化数据604的查询表达式606：

[0587] From C In Customers

[0588] 而不要求查询表达式606包括显式的SELECT、RETURN或其它终止查询运算符。

[0589] 作为解释，变换组件602可被配置成向接收到的每一查询子句追加隐式SELECT或RETURN。由此，只要包括在流水线中的所得类集处于期望的形状，就不需要显式语句来选择或返回这些中间结果。相反，如果查询表达式606在给定子句，例如702-708中的任一个之后结束，则流水线中的类集可以输出结果，并且可以基于范围中的控制变量的数目来推断输出的格式。

[0590] 例如，如果查询表达式606在子句704-708中的任一个之后终止(其中在每一情况下范围中有两个控制变量C和O)，则可以推断所需输出应作为名-值对的类集存在。因此，输出可以是对每一控制变量有一字段的元组。另一方面，在范围中仅有一个控制变量的情况下，诸如在子句702之后，则查询表达式606的实现器通常将不期望得回具有字段c的元组。相反，实现器可能期望顾客的类集。因此，在范围中仅有一个控制变量的情况下，输出可以仅仅是类集的底层值。

[0591] 根据所要求保护的主题的一方面，由FROM子句引入的控制变量可以与表达式以及类集相关联。作为一个代表性示例，考虑以下如上所述的查询片段：

[0592] From b in Books_

[0593] From Tax＝b.Price*0.088_

[0594] 第一个FROM子句为类集Books引入控制变量b，该控制变量被带入范围并可由下一查询子句来访问。下一FROM子句引入与表达式(而非类集)b.Price*0.088相关联的控制变量Tax。对于两个FROM子句中的后者，效果可以是为表达式的结果提供了可访问名称。该名称可由控制变量tax来引用，该控制变量可保留在范围中，直到例如后续的SELECT子句。该特征可被认为类似于查询中的过程语句，这可允许表达式的值被计算一次，即使该值随后可被多次采用，也不需要重新计算该值。

[0595] 根据所要求保护的主题的另一方面，可以在单个查询表达式606的上下文中采用集合运算。尽管常规查询语言通常仅提供单个类集作为输入，但是变换组件602可以接收两个类集作为输入。例如，变换组件602可被配置成接收多个类集，并可输出具有与根据查询子句的类型(例如，UNION子句、INTERSECT子句.....)合并的多个类集相关联的控制变量的单个类集。合并的控制变量可被带入范围、通过相关联的集合运算符传递、并对下一查询子句可访问，因此查询表达式606可继续。控制变量一般必须是相同的类型，并且一般必须具有相同的名称，但是当然，它们可以与两个不同的类集相关联。

[0596] 现在转向图8，示出了用于方便以合成方式构造查询综合的计算机实现的方法800。从参考标号802开始，可获得类集(或表达式)以及与该类集(或表达式)相关联的控制变量。一般而言，该获得的数据是查询表达式的第一个查询子句，通常是FROM子句的产物。在参考标号804处，可接收包括在查询表达式中的一组查询子句的当前查询子句。可以理解，与要求综合是整体式的常规语言不同，不需要对当前查询子句，或在很大程度上对作为整体的查询表达式施加句法排序限制。

[0597] 在参考标号806处，可根据当前查询子句来修改类集。简言之，作为输入接收的类集可以基于与当前查询子句相关联的查询运算符的方针来变换。可以理解，当前查询子句

可以根据与当前查询子句相关联的期望类型而非基于整体式句法模板来求值。接着，在参考标号808处，可将经修改的类集通过流水线传送到下一查询子句。因此，当前查询子句的输出可用作下一查询子句的输入。

[0598] 在参考标号810处，可基于当前查询子句来确定控制变量的范围。例如，某些类型的查询子句不便于对现存控制变量进行任何改变，而其它查询子句则可方便引入新的控制变量(在某些情况下仅对于特定的一系列下一查询子句)，而还有一些则可方便在(可任选地)引入新控制变量的同时将现存控制变量带出范围。在参考标号812处，可将对当前查询子句的范围中控制变量的访问提供给下一查询子句。

[0599] 现在参考图9，所示是可用于执行所公开的体系结构的示例性计算机系统的框图。为了为本发明各个方面提供附加上下文，图9和下列讨论旨在提供对其中可实现本发明的各方面的合适的计算环境900的简要概括的描述。另外，尽管本发明以上是在可在一个或多个计算机上运行的计算机可执行指令的一般上下文中进行描述的，但是本领域的技术人员将认识到，本发明也可结合其它程序模块和/或作为硬件和软件的组合来实现。

[0600] 一般而言，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、组件、数据结构等等。此外，本领域的技术人员可以理解，本发明的方法可用其它计算机系统配置来实施，包括单处理器或多处理器计算机系统、小型机、大型计算机、以及个人计算机、手持式计算设备、基于微处理器的或可编程消费电子产品等，其每一个都可操作上耦合到一个或多个相关联的设备。

[0601] 本发明的所示方面也可以在其中特定任务由通过通信网络链接的远程处理设备执行的分布式计算环境中实践。在分布式计算环境中，程序模块可以位于本地和远程存储器存储设备中。

[0602] 计算机通常包括各种计算机可读介质。计算机可读介质可以是可由计算机访问的任何可用介质，并包括易失性和非易失性介质、可移动和不可移动介质。作为示例而非限制，计算机可读介质可以包括计算机存储介质和通信介质。计算机存储介质可包括以用于存储诸如计算机可读指令、数据结构、程序模块或其它数据这样的信息的任意方法或技术来实现的易失性和非易失性、可移动和不可移动介质。计算机存储介质包括但不限于，RAM、ROM、EEPROM、闪存或其它存储器技术、CD-ROM、数字多功能盘(DVD)或其它光盘存储、磁盒、磁带、磁盘存储或其它磁存储设备、或可以用来储存所期望的信息并可由计算机访问的任何其它介质。

[0603] 通信介质通常以诸如载波或其它传输机制等已调制数据信号来体现计算机可读指令、数据结构、程序模块或其它数据，且包含任何信息传递介质。术语“已调制数据信号”指的是其一个或多个特征以在信号中编码信息的方式被设定或更改的信号。作为示例而非限制，通信介质包括有线介质，诸如有线网络或直接线连接，以及无线介质，诸如声学、RF、红外线和其它无线介质。上述中的任意组合也应包括在计算机可读介质的范围之内。

[0604] 再次参见图9，用于实现本发明的各方面的示例性环境900包括计算机902，计算机902包括处理单元904、系统存储器906和系统总线908。系统总线908将包括但不限于系统存储器906的系统组件耦合到处理单元904。处理单元904可以是市场上可购买到的各种处理器中的任意一种。双微处理器和其它多处理器体系结构也可用作处理单元904。

[0605] 系统总线908可以是若干种总线结构中的任一种，这些总线结构还可互连到存储

器总线(带有或没有存储器控制器)、外围总线、以及使用各类市场上可购买到的总线体系结构中的任一种的局部总线。系统存储器906包括只读存储器(ROM)910和随机存取存储器(RAM)912。基本输入/输出系统(BIOS)储存在诸如ROM、EPROM、EEPROM等非易失性存储器910中,其中BIOS包含帮助诸如在启动期间在计算机902内的元件之间传输信息的基本例程。RAM 912还可包括诸如静态RAM等高速RAM来用于高速缓存数据。

[0606] 计算机902还包括内部硬盘驱动器(HDD)914(例如,EIDE、SATA),该内部硬盘驱动器914还可被配置成在合适的机壳(未示出)中外部使用;磁软盘驱动器(FDD)916(例如,从可移动磁盘918中读取或向其写入);以及光盘驱动器920(例如,从CD-ROM盘922中读取,或从诸如DVD等高容量光学介质中读取或向其写入)。硬盘驱动器914、磁盘驱动器916和光盘驱动器920可分别通过硬盘驱动器接口924、磁盘驱动器接口926和光盘驱动器接口928连接到系统总线908。用于外置驱动器实现的接口924包括通用串行总线(USB)和IEEE 1394接口技术中的至少一种或两者。其它外置驱动器连接技术也在本发明的预期中。

[0607] 驱动器及其相关联的计算机可读介质提供了对数据、数据结构、计算机可执行指令等的非易失性存储。对于计算机902,驱动器和介质容纳适当的数字格式的任何数据的存储。尽管上面对计算机可读介质的描述涉及HDD、可移动磁盘和诸如CD或DVD等可移动光学介质,但本领域的技术人员应该意识到,在示例性操作环境中也可以使用可由计算机读取的其他类型的介质,如ZIP驱动器、磁带盒、闪存卡、盒式磁带等等,并且此外,任何这种介质都可以包含用于执行本发明的方法的计算机可执行指令。

[0608] 多个程序模块可存储在驱动器和RAM 912中,包括操作系统930、一个或多个应用程序932、其它程序模块934和程序数据936。所有或部分操作系统、应用程序、模块和/或数据也可被高速缓存在RAM 912中。应该明白,本发明可以用各种市场上可购买到的操作系统或操作系统的组合来实现。

[0609] 用户可以通过一个或多个有线/无线输入设备,例如键盘938和诸如鼠标940等定点设备将命令和信息输入到计算机902中。其它输入设备(未示出)可包括话筒、IR遥控器、操纵杆、游戏手柄、指示笔、触摸屏等等。这些和其它输入设备通常通过耦合到系统总线908的输入设备接口942连接到处理单元904,但也可通过其它接口连接,如并行端口、IEEE 1394串行端口、游戏端口、USB端口、IR接口等等。

[0610] 监视器944或其它类型的显示设备也经由诸如视频适配器946等接口来连接到系统总线908。除了监视器944之外,计算机通常包括诸如扬声器、打印机等其它外围输出设备(未示出)。

[0611] 计算机902可使用经由有线和/或无线通信至一个或多个远程计算机,诸如远程计算机948的逻辑连接在网络化环境中操作。远程计算机948可以是工作站、服务器计算机、路由器、个人计算机、便携式计算机、基于微处理器的娱乐设备、对等设备或其它常见的网络节点,并且通常包括以上相对于计算机902描述的许多或所有元件,尽管为简明起见仅示出了存储器/存储设备950。所描绘的逻辑连接包括到局域网(LAN)952和/或例如广域网(WAN)954等更大的网络的有线/无线连接。这一LAN和WAN联网环境常见于办公室和公司,并且便于诸如内联网等企业范围计算机网络,所有这些都可连接到例如因特网等全球通信网络。

[0612] 当在LAN网络环境中使用时,计算机902通过有线和/或无线通信网络接口或适配器956连接到局域网952。适配器956可以便于到LAN 952的有线或无线通信,并且还可包括

其上设置的用于与无线适配器956通信的无线接入点。

[0613] 当在WAN连网环境中使用时，计算机902可包括调制解调器958，或连接到WAN 954上的通信服务器，或具有用于通过WAN 954，诸如通过因特网建立通信的其它装置。或为内置或为外置的调制解调器958以及有线或无线设备经由串行端口接口942连接到系统总线908。在网络化环境中，相对于计算机902所描述的程序模块或其部分可以存储在远程存储器/存储设备950中。应该理解，所示网络连接是示例性的，并且可以使用在计算机之间建立通信链路的其它手段。

[0614] 计算机902可用于与操作上设置在无线通信中的任何无线设备或实体通信，这些设备或实体例如有打印机、扫描仪、台式和/或便携式计算机、便携式数据助理、通信卫星、与无线可检测标签相关联的任何一个设备或位置(例如，公用电话亭、报亭、休息室)以及电话。这至少包括Wi-Fi和蓝牙™无线技术。由此，通信可以如对于常规网络那样是预定义结构，或者仅仅是至少两个设备之间的自组织(ad hoc)通信。

[0615] Wi-Fi，即无线保真，允许从家里沙发、酒店房间的床上或工作的会议室连接到因特网而不需要线缆。Wi-Fi是一种类似蜂窝电话中使用的无线技术，它使得诸如计算机等设备能够在室内和室外，在基站范围内的任何地方发送和接收数据。Wi-Fi网络使用称为IEEE 802.11(a、b、g等等)的无线电技术来提供安全、可靠、快速的无线连接。Wi-Fi网络可用于将计算机彼此连接、连接到因特网以及连接到有线网络(使用IEEE 802.3或以太网)。Wi-Fi网络在未许可的2.4和5GHz无线电波段内工作，例如以11Mbps(802.11a)或54Mbps(802.11b)数据速率工作，或者具有包含两个波段(双波段)的产品，因此该网络可提供类似于许多办公室中使用的基本10BaseT有线以太网的真实性能。

[0616] 现在参考图10，所示是可用于执行所公开的体系结构的示例性计算机编译系统的示意性框图。系统1000包括一个或多个客户机1002。客户机1002可以是硬件和/或软件(例如，线程、进程、计算设备)。例如，客户机1002可以通过使用本发明来容纳cookie和/或相关联的上下文信息。

[0617] 系统1000还包括一个或多个服务器1004。服务器1004也可以是硬件和/或软件(例如，线程、进程、计算设备)。服务器1004可以例如通过使用本发明来容纳线程以执行变换。在客户机1002和服务器1004之间的一种可能的通信能够以适合在两个或多个计算机进程之间传输的数据分组的形式进行。数据分组可包括例如cookie和/或相关联的上下文信息。系统1000包括可以用来使客户机1002和服务器1004之间通信更容易的通信框架1006(例如，诸如因特网等全球通信网络)。

[0618] 通信可经由有线(包括光纤)和/或无线技术来促进。客户机1002操作上被连接到可以用来存储对客户机1002本地的信息(例如，cookie和/或相关联的上下文信息)的一个或多个客户机数据存储1008。同样地，服务器1004可在操作上连接到可以用来存储对服务器1004本地的信息的一个或多个服务器数据存储1010。

[0619] 以上所描述的包括各实施例的示例。当然，出于描绘各实施例的目的而描述组件或方法的每一个可以想到的组合是不可能的，但本领域内的普通技术人员可以认识到，许多进一步的组合和排列都是可能的。因此，本详细描述旨在涵盖所有这些落入所附权利要求书的精神和范围内的更改、修改和变化。

[0620] 特别地，对于由上述组件、设备、电路、系统等执行的各种功能，除非另外指明，否

则用于描述这些组件的术语(包括对“装置”的引用)旨在对应于(除非另外指明)执行所描述的执行此处在各实施例的示例性方面中所示的功能的组件的指定功能(例如,功能上等效)的任何组件,即使这些组件在结构上不等效于所公开的结构。在这一点上,也可认识到各实施例包括用于执行各方法的动作和/或事件的系统以及具有用于执行这些动作和/或事件的计算机可执行指令的计算机可读介质。

[0621] 另外,尽管可相对于若干实现中的仅一个来公开一个特定特征,但是这一特征可以如对任何给定或特定应用所需且有利地与其它实现的一个或多个其它特征相组合。此外,就在说明书或权利要求书中使用术语“包括”和“含有”及其变体而言,这些术语旨在以与术语“包含”相似的方式为包含性的。

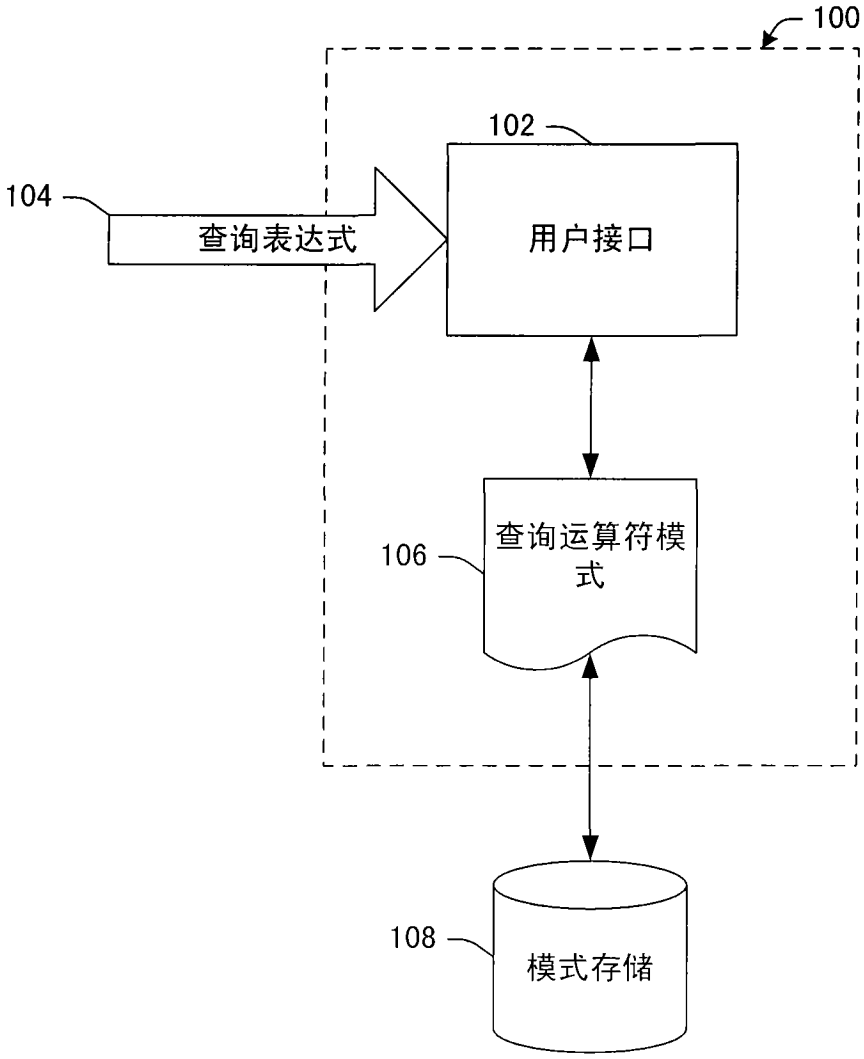


图1

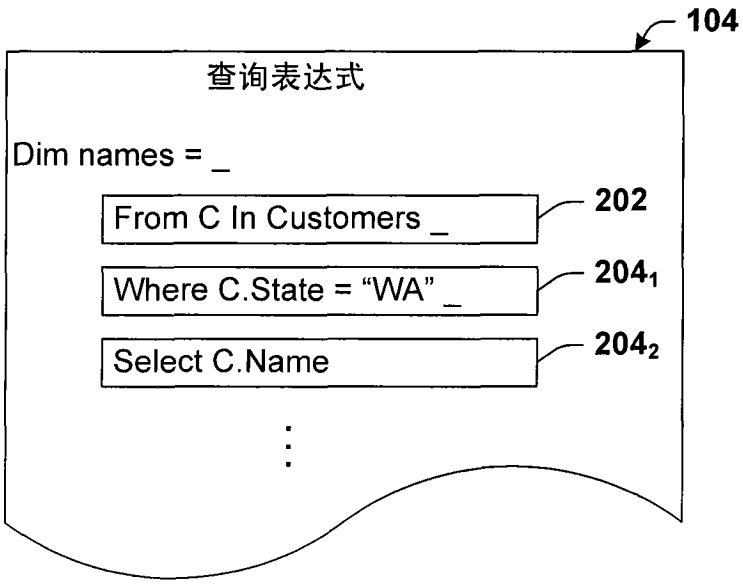

104
查询表达式
Dim names = _
From C In Customers _
202
Where C.State = "WA" _
204₁
Select C.Name
204₂

图2A

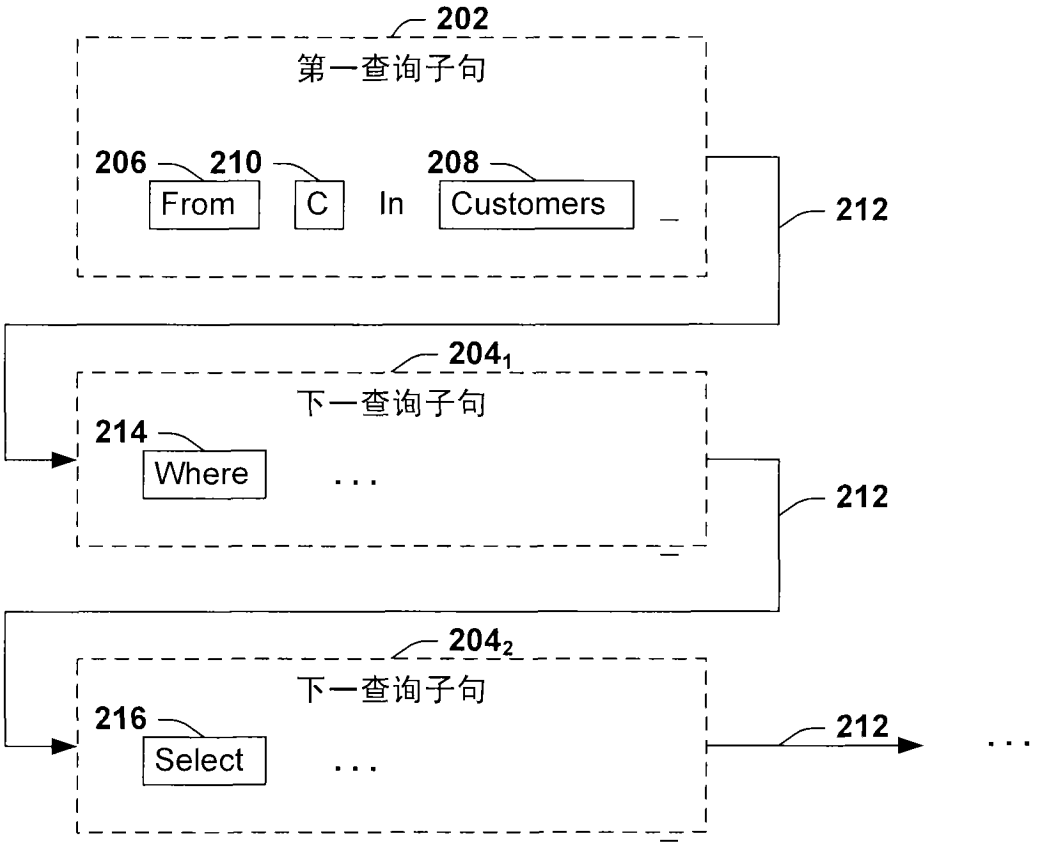

202
第一查询子句
206
210
208
From
C
In
Customers
_
212
204₁
下一查询子句
214
Where
...
212
204₂
下一查询子句
216
Select
...
212
...

图2B

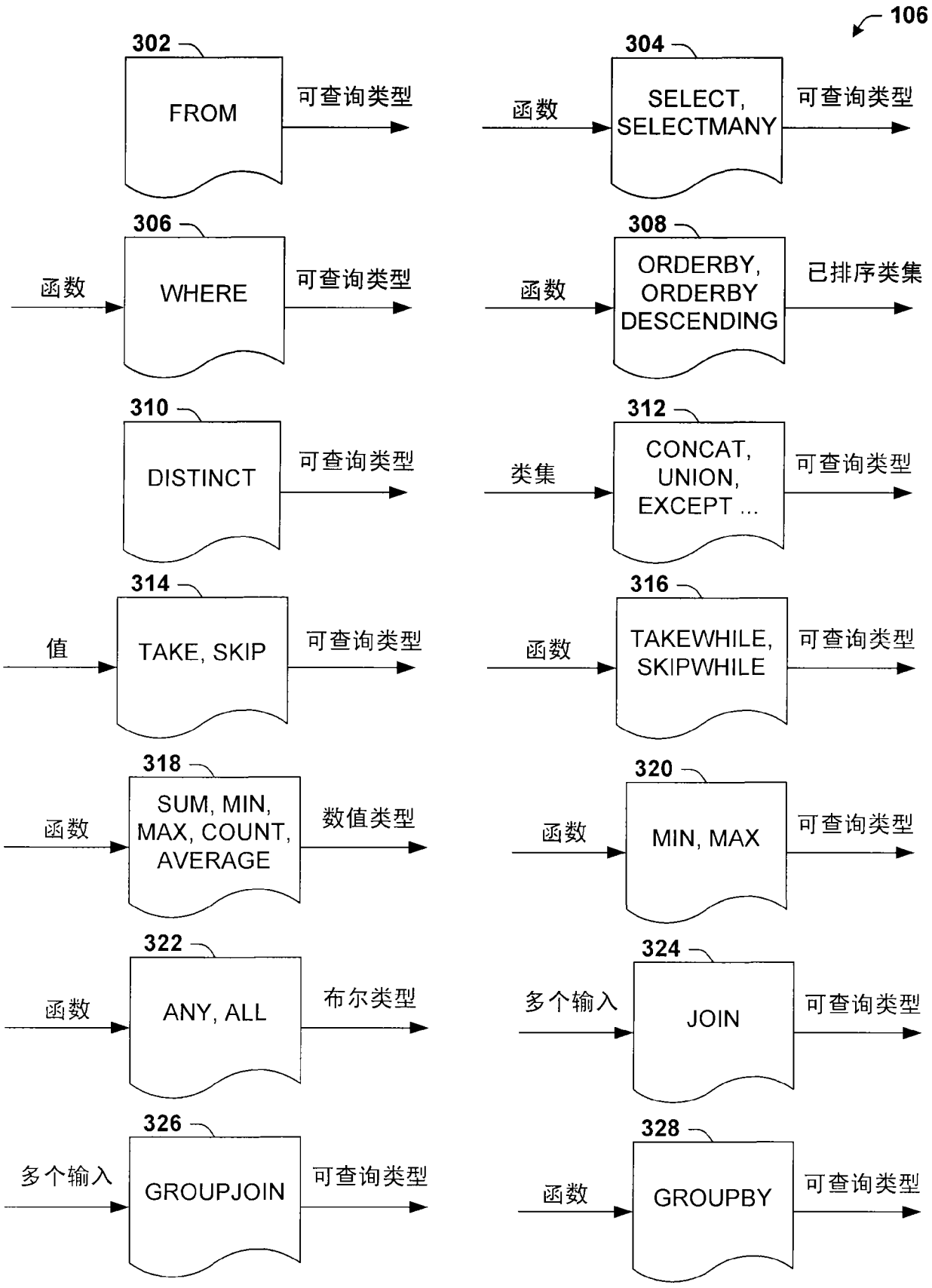

106
302
FROM
可查询类型
304
函数
SELECT, SELECTMANY
可查询类型
306
函数
WHERE
可查询类型
308
函数
ORDERBY, ORDERBY DESCENDING
已排序类集
310
DISTINCT
可查询类型
312
类集
CONCAT, UNION, EXCEPT ...
可查询类型
314
值
TAKE, SKIP
可查询类型
316
函数
TAKEWHILE, SKIPWHILE
可查询类型
318
函数
SUM, MIN, MAX, COUNT, AVERAGE
数值类型
320
函数
MIN, MAX
可查询类型
322
函数
ANY, ALL
布尔类型
324
多个输入
JOIN
可查询类型
326
多个输入
GROUPJOIN
可查询类型
328
函数
GROUPBY
可查询类型

图3

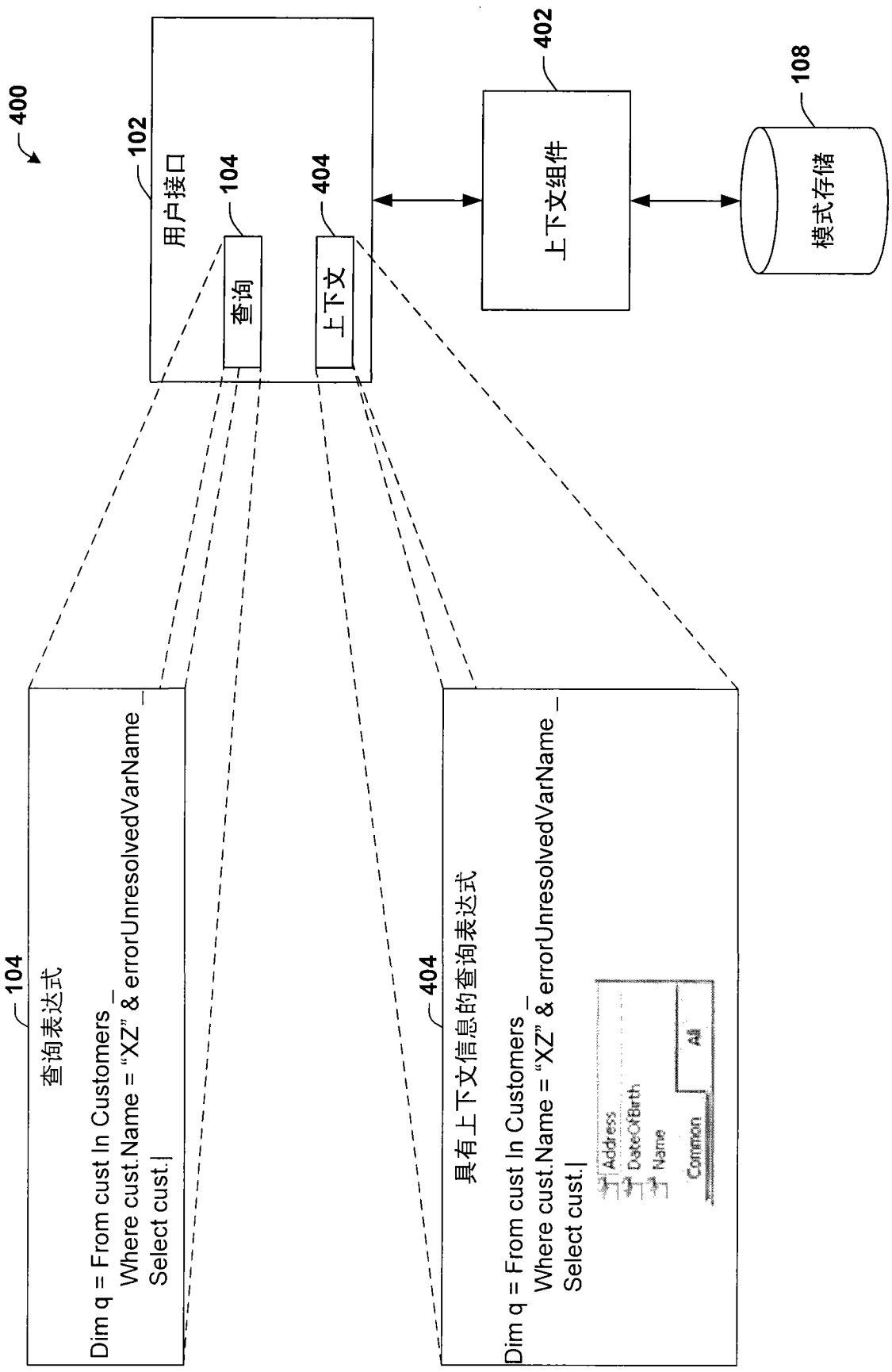

400
102
用户接口
104
查询
404
上下文
402
上下文组件
108
模式存储
104
查询表达式
Dim q = From cust In Customers _
Where cust.Name = "XZ" & errorUnresolvedVarName _
Select cust.|
404
具有上下文信息的查询表达式
Dim q = From cust In Customers _
Where cust.Name = "XZ" & errorUnresolvedVarName _
Select cust.|
Address
DateOfBirth
Name
Common
All

图4

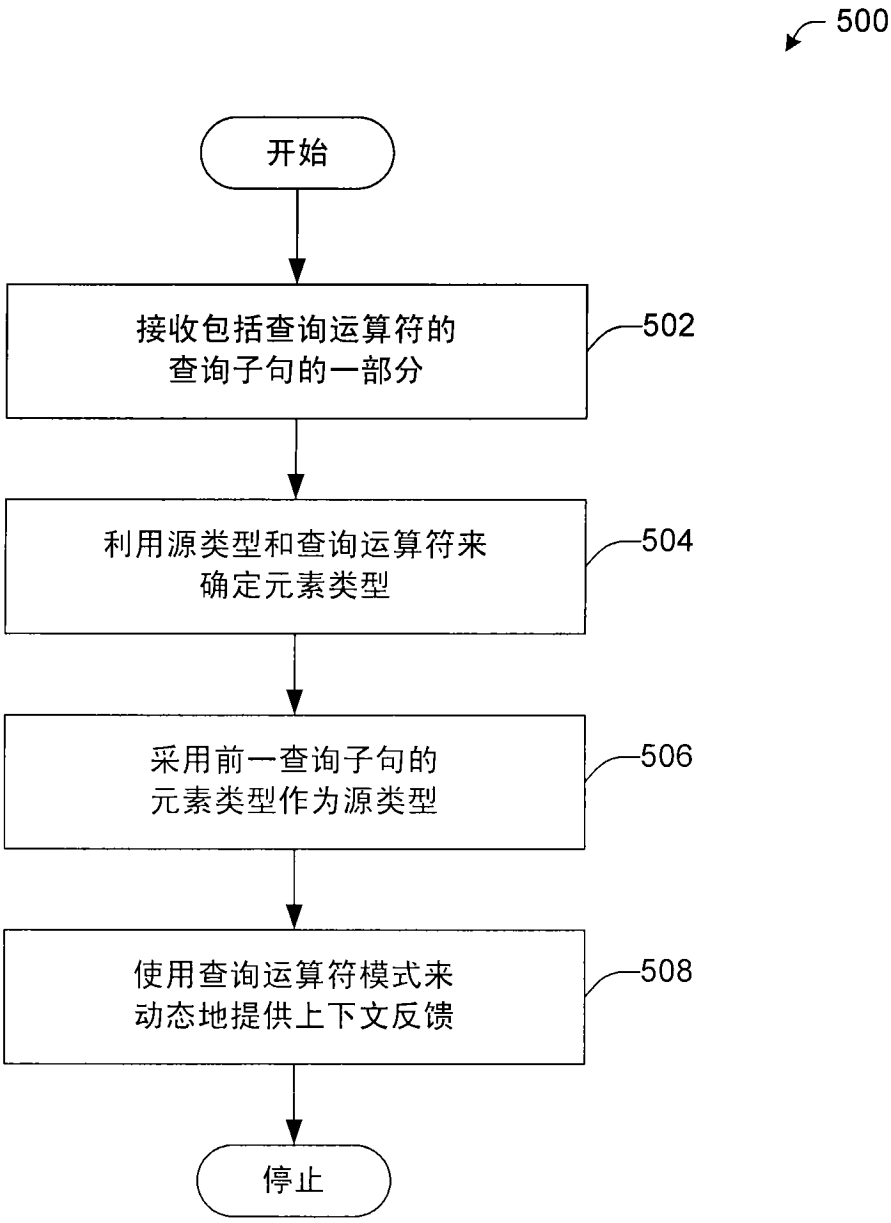

500
开始
接收包括查询运算符的查询子句的一部分
502
利用源类型和查询运算符来确定元素类型
504
采用前一查询子句的元素类型作为源类型
506
使用查询运算符模式来动态地提供上下文反馈
508
停止

图5

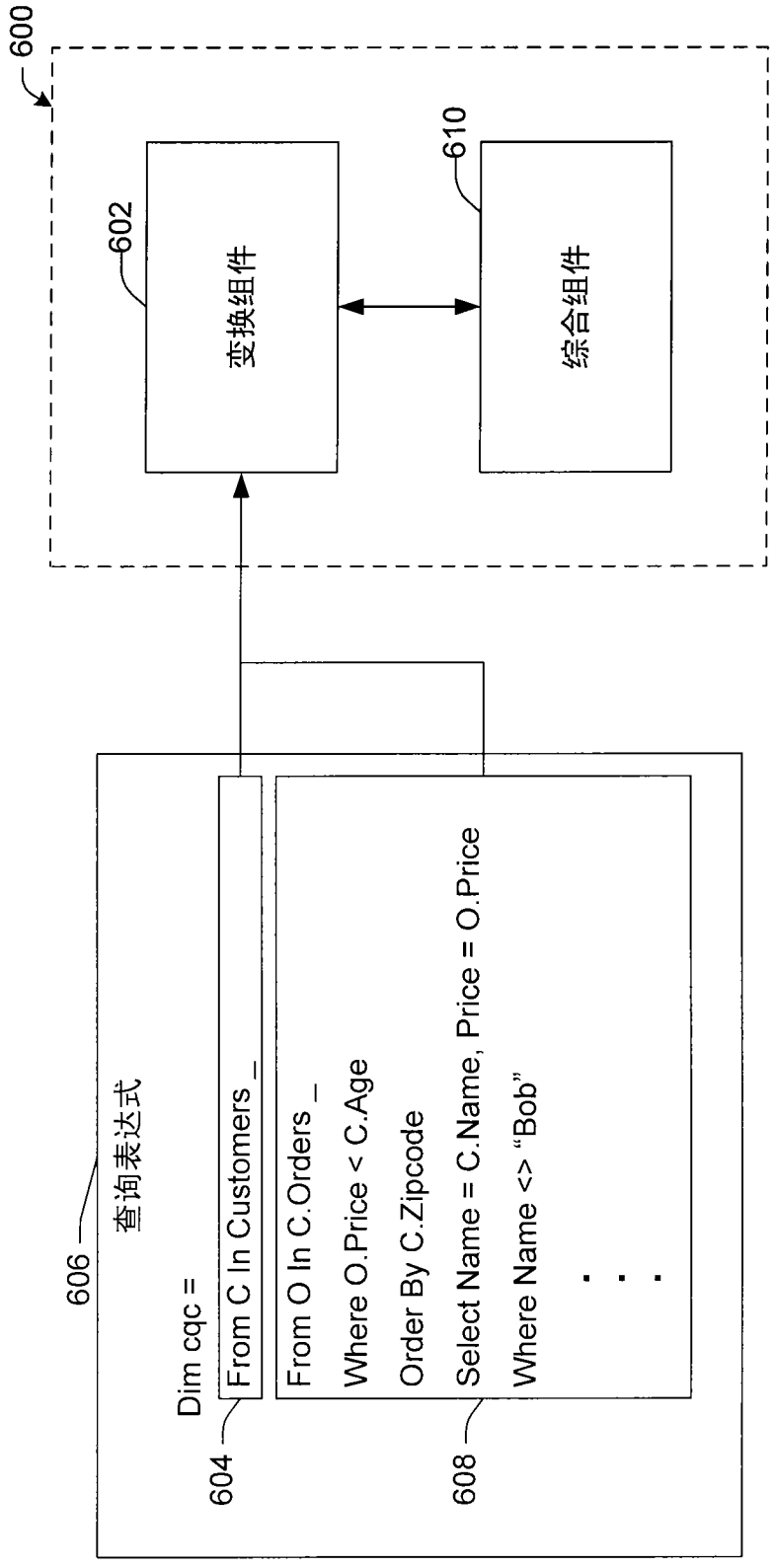

600
602
变换组件
610
综合组件
606
查询表达式
Dim cqc =
From C In Customers _
From O In C.Orders _
Where O.Price < C.Age
Order By C.Zipcode
Select Name = C.Name, Price = O.Price
Where Name <> "Bob"
. . .
604
608

图6

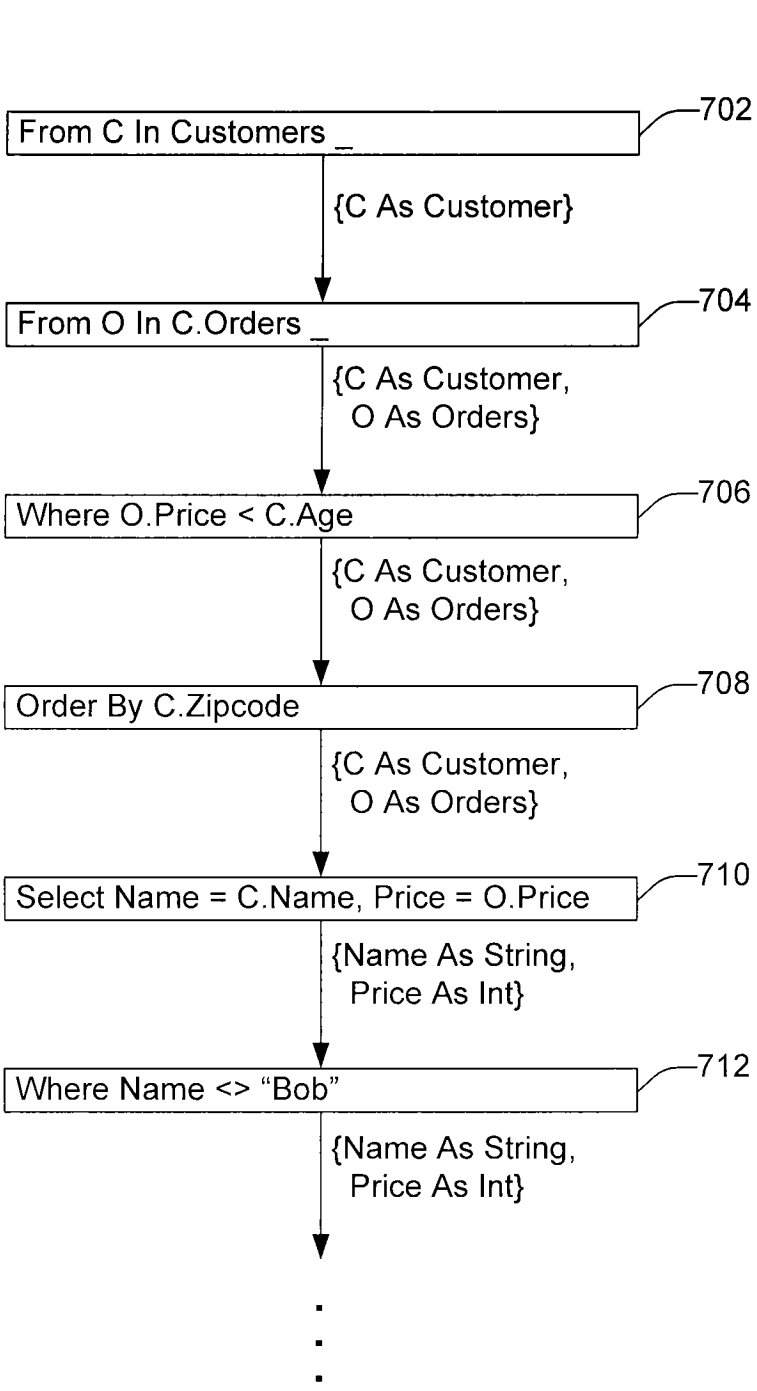

700
From C In Customers _
702
{C As Customer}
From O In C.Orders _
704
{C As Customer,
O As Orders}
Where O.Price < C.Age
706
{C As Customer,
O As Orders}
Order By C.Zipcode
708
{C As Customer,
O As Orders}
Select Name = C.Name, Price = O.Price
710
{Name As String,
Price As Int}
Where Name <> “Bob”
712
{Name As String,
Price As Int}

图7

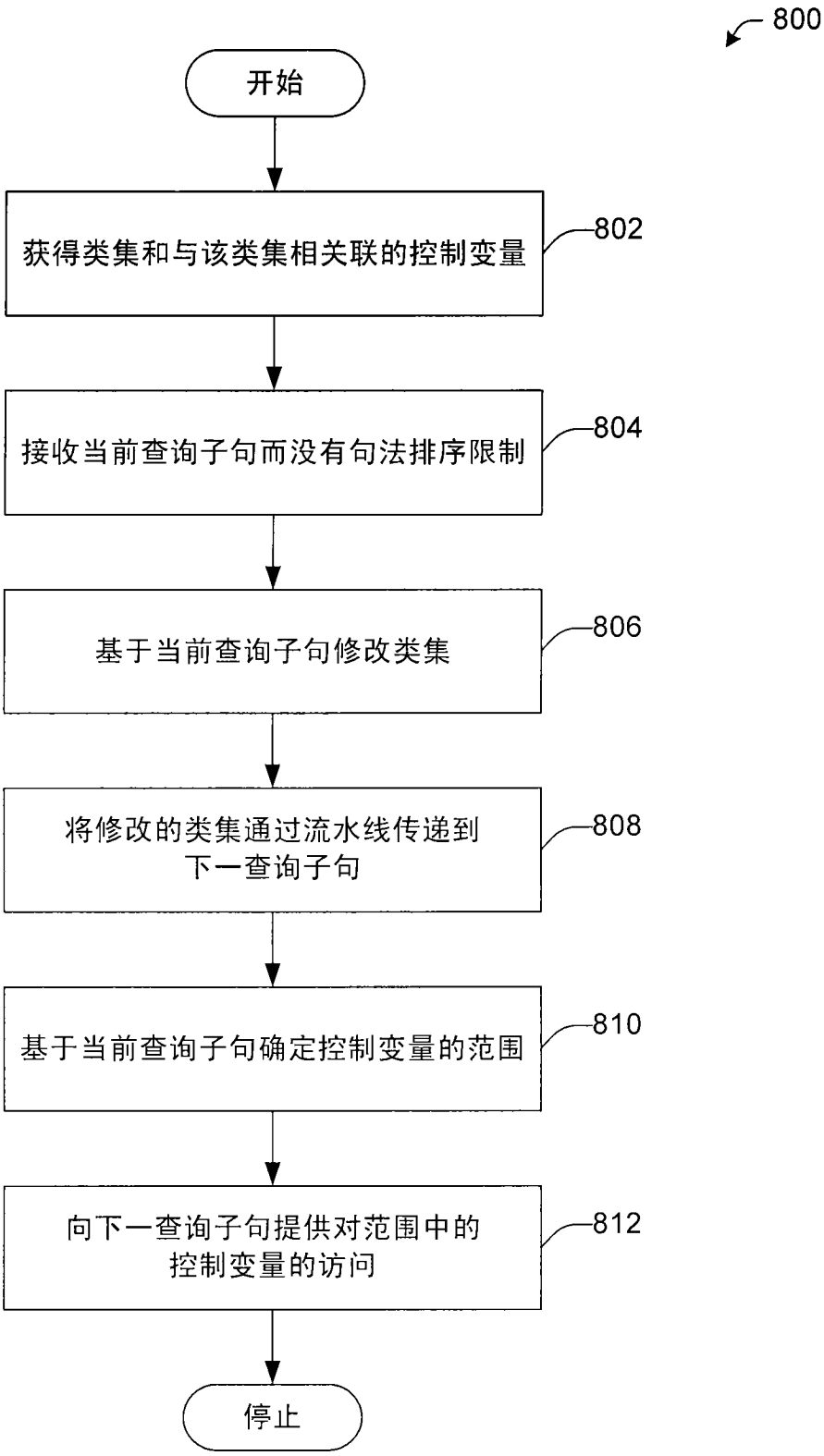

800
开始
获得类集和与该类集相关联的控制变量
802
接收当前查询子句而没有句法排序限制
804
基于当前查询子句修改类集
806
将修改的类集通过流水线传递到
下一查询子句
808
基于当前查询子句确定控制变量的范围
810
向下一查询子句提供对范围中的
控制变量的访问
812
停止

图8

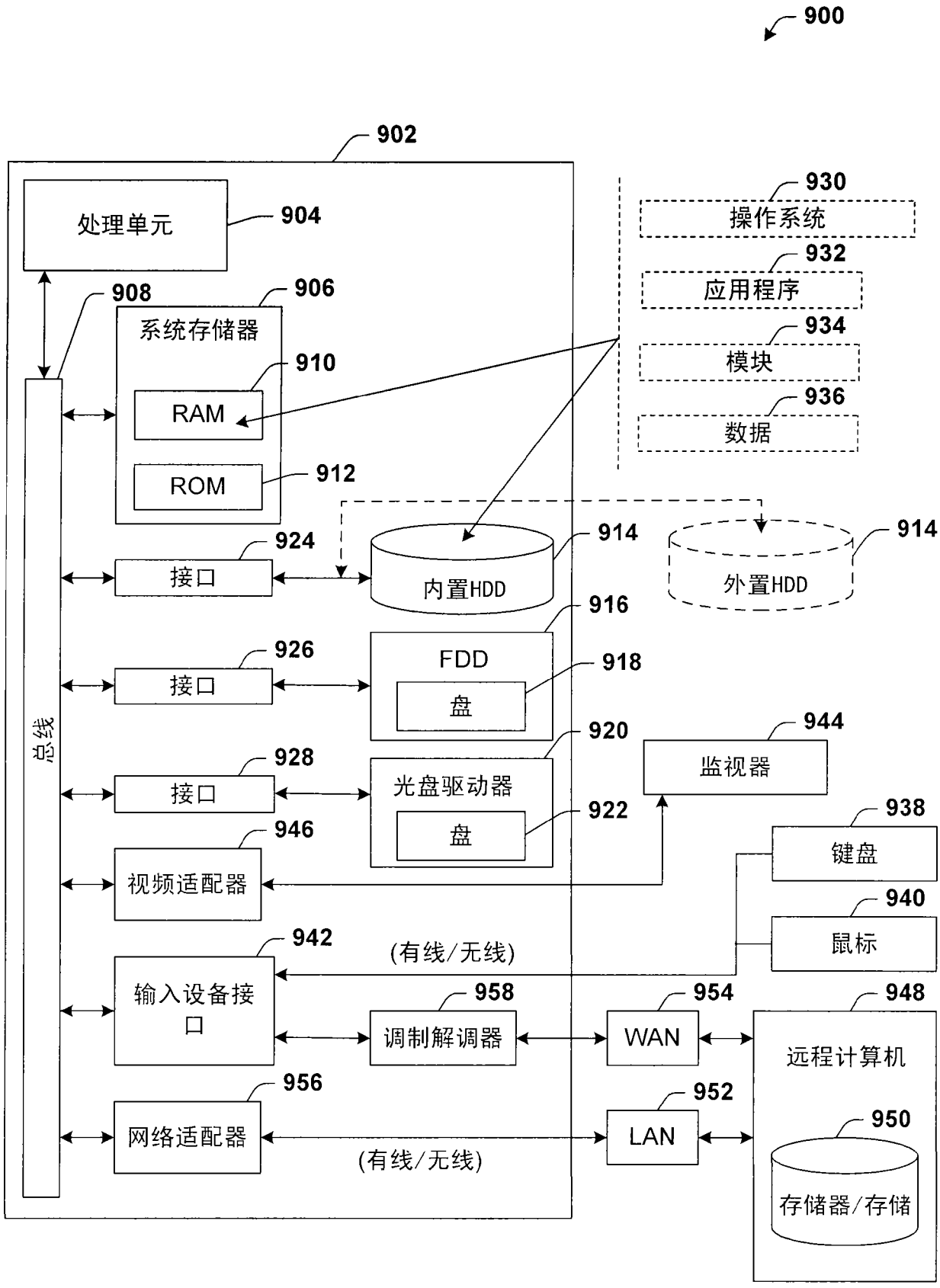

900
902
处理单元
904
908
906
系统存储器
910
RAM
ROM
912
930
操作系统
932
应用程序
934
模块
936
数据
924
接口
914
内置HDD
914
外置HDD
916
926
接口
FDD
918
盘
920
928
接口
光盘驱动器
922
盘
944
监视器
946
视频适配器
938
键盘
940
鼠标
942
(有线/无线)
输入设备接口
958
调制解调器
954
WAN
948
远程计算机
956
网络适配器
952
LAN
950
存储器/存储
(有线/无线)
总线

图9

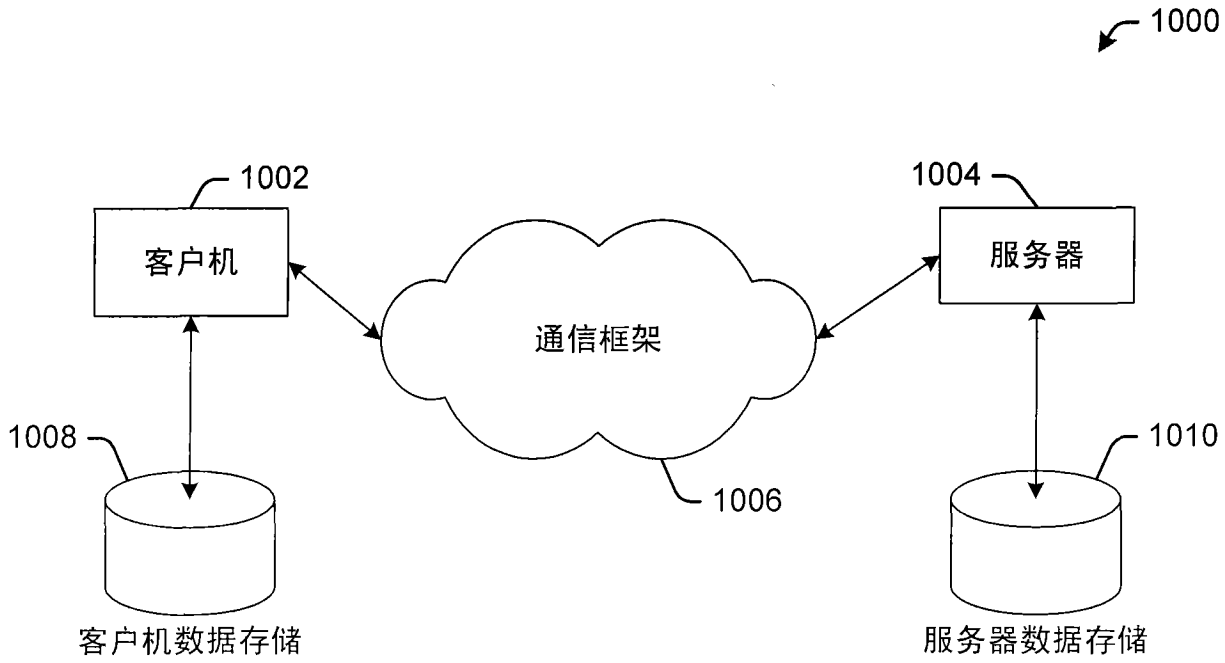


图10