2数据图

任何知识图的基础都是首先将图抽象应用于数据，得到初始数据图。现在，我们将讨论一些在实践中常用来表示数据图的图结构数据模型。然后，我们讨论构成用于查询这些数据图的图查询语言基础的原语。

2.1模型

撇开图表不谈，让我们假设旅游委员会还没有决定如何为景点、事件、服务等相关数据建模。委员会首先考虑使用表格结构——特别是关系数据库——来表示所需的数据，尽管他们不确切知道需要捕获什么数据，但他们开始设计一个初始的关系模式。它们从一个有5列的Event表开始:

Event(name, venue,type,start, end)

其中name和start共同构成表的主键，以便惟一地标识重复发生的事件。但是，当他们开始填充数据时，他们会遇到各种问题:事件可能有多个名称(例如，在不同的语言中)，事件可能有多个地点，他们可能还不知道未来事件的开始和结束日期时间，事件可能有多个类型，等等。随着数据变得更加多样化，逐步解决这些建模问题，它们为事件生成内部标识符，并调整它们的关系模式，直到有:

EventName(id, name), EventStart(id, start), EventEnd(id, end), (1)

EventVenue(id, venue), EventType(id, type)

有了上述模式，组织现在可以用0-n个名称、地点和类型以及0-1个开始日期和结束日期对事件进行建模(不需要表中的关系空/空白单元格)。

在此过程中，为了支持新的数据源，董事会不得不多次增量地更改模式。每一次这样的更改都需要进行昂贵的数据重构、重新加载和重建数据;这里我们只考虑了一张表。旅游局纠结于关系模型，因为他们事先不知道哪些数据需要建模，或者他们将使用哪些数据源。但是，一旦他们到达后一种关系模式，董事会发现他们可以在不做更多更改的情况下集成更多的数据源:在对多样性(1-1、1-n等)的最小假设下，该模式为集成不完整和不同的数据提供了很大的灵活性。

事实上，这个板最终得到的精细、灵活的模式(如(1)所示)是对实体之间的一组二元关系进行建模，这实际上可以被看作是对一个图形进行建模。通过从一开始就采用图形数据模型，董事会可以放弃预先模式的需要，并且可以在任何时候定义任何一对实体之间的任何(二进制)关系。

我们现在介绍三种实际中常用的图形数据模型[14]。

2.1.1有向带标签图表。

一个有向边带标签的图被定义为一组节点——如——以及这些节点之间的一组有向带标签的边，如，对于知识图，节点用来表示实体，边用来表示实体之间的(二进制)关系。图1提供了一个示例的旅游局可以将一些相关的事件数据建模为一个有向的带标签图(正式定义的一个有向带标签图见附录B中定义规则B.1),这个图表包括有关名称、类型、开始和结束日期时间以及活动地点的数据2,向这样的图中添加信息通常需要添加新的节点和边(除了后面讨论的一些例外情况)表示不完全信息需要简单地省略一条特定的边;例如，图表还没有定义Food Truck festival的开始/结束日期-时间。

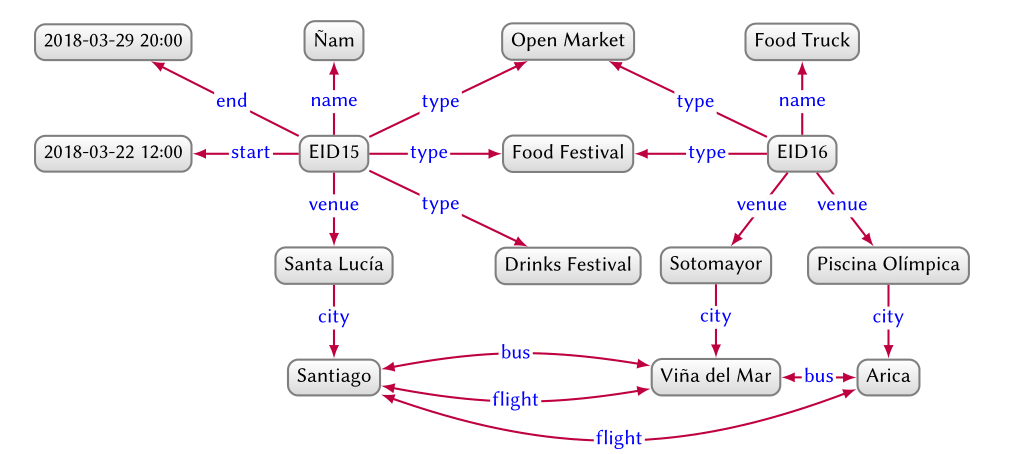


图1所示。描述事件及其场地的有向边标记图。

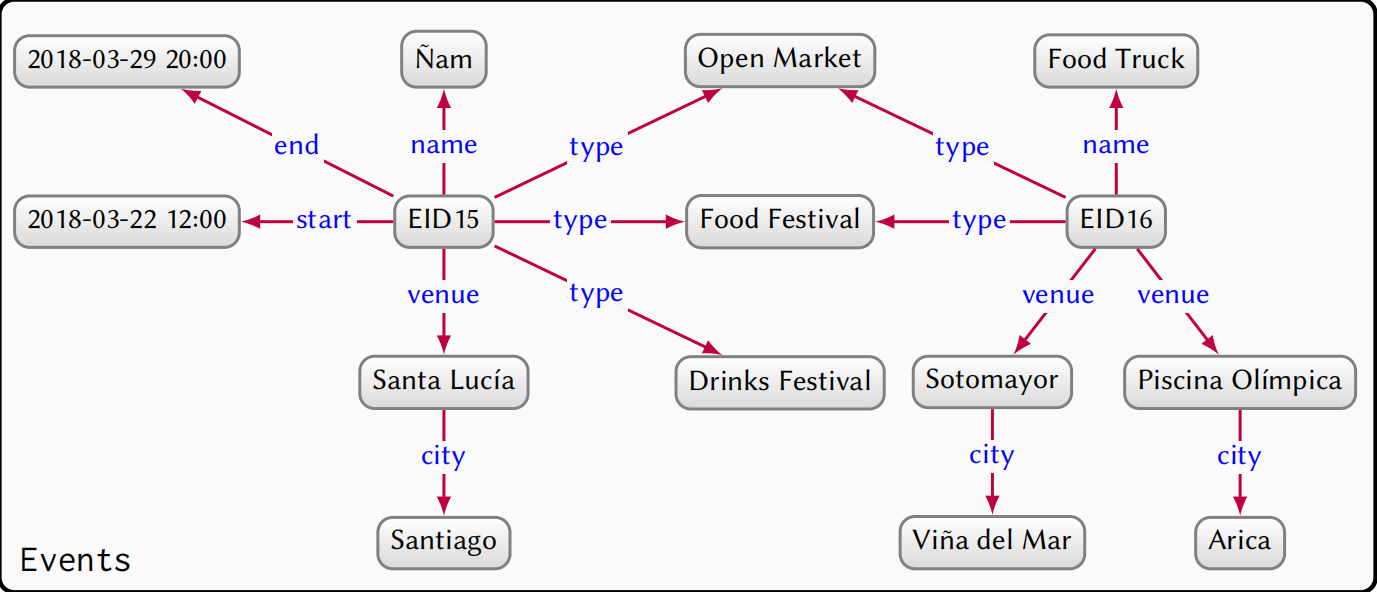
与标准关系模型相比，以这种方式将数据建模为图形为集成新数据源提供了更大的灵活性，在标准关系模型中必须预先定义模式，并在每个步骤中遵循模式。虽然其他结构化数据模型(如树(XML、JSON等))提供了类似的灵活性，但图不需要分层组织数据(例如，地点应该是父类型、子类型或兄弟类型?)它们还允许表示和查询循环(例如，注意Santiago, Arica, and Viña del Mar之间的路线中的有向循环)。

基于有向边缘标记图的标准化数据模型是资源描述框架(RDF)[106]，它已被W3C推荐。RDF模型定义了不同类型的节点，包括国际化资源标识符(IRIs)[129]，它允许在Web上对实体进行全局识别;字面值，它允许表示字符串(带有或不带有语言标记)和其他数据类型值(整数、日期等);和空白节点，这是没有分配标识符的匿名节点(例如，我们可以选择使用空白节点，而不是在RDF中创建像EID15、EID16这样的内部标识符)。当我们讨论与身份相关的问题时，我们将在第3.2节中进一步讨论这些不同类型的节点。

2.1.2图像数据集。

虽然多个有向边标记图可以通过取它们的并集来合并，但通常希望管理多个图而不是一个单片图;例如，管理来自不同来源的多个图可能是有益的，这使得更新或细化来自一个来源的数据成为可能，从而区分不可信的来源和更可信的来源，等等。图数据集由一组命名图和一个默认图组成。每个命名的图是一对图ID和一个图。默认图是一个没有ID的图，如果没有指定ID，则“默认”被引用。图2提供了一个示例事件和路线都存储在两个命名图,和默认图管理元数据的命名图(正式图数据集的定义,参见附录B中定义B.3)。我们强调图的名字也可以用作节点图。此外，节点和边可以在图中重复，不同图中的相同节点通常指向相同的实体，从而允许在合并图时集成实体上的数据。

包含动词的完整边缘标签，如has-vention或is-valid-from，在本文中，出于演示目的，我们将省略此类标签中的“has”和“is”动词，只使用vention或valid-from。



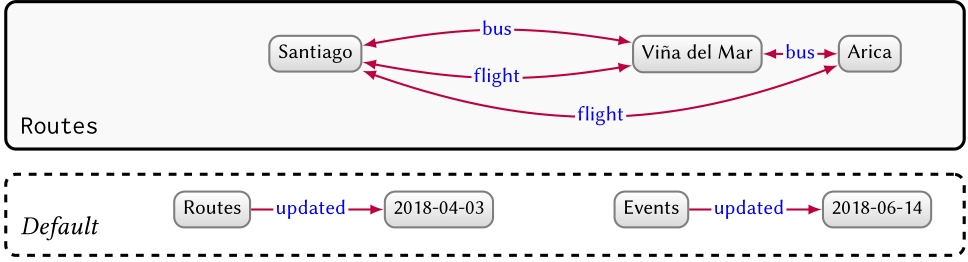


图2所示。图形数据集，包含两个命名的图形和一个描述事件和路由的默认图形

图数据集的一个突出用例是管理和查询由跨越Web的RDF图的相互链接的文档组成的链接数据。在处理Web数据时，跟踪数据的来源变得非常重要[54,125,546]。我们将在后面的3.2节中讨论关联数据，并在3.3节中进一步讨论关联数据的来源。

2.1.3属性图。

属性图的引入为建模更复杂的关系提供了额外的灵活性。考虑集成输入的数据，这些数据提供关于哪些公司提供哪些航班的票价的信息，从而让委员会更好地了解城市之间的可用航线(例如，在国家航空公司上)。在有向边标签图的情况下，我们不能直接注释像这样的边与提供该路线的公司(或公司)。但是我们可以添加一个表示航班的新节点，将其连接到源、目的地、公司和模式，如图3所示。然而，将这种建模应用到图1中的所有数据将涉及对图表的重大更改。另一种选择可能是将不同公司的航班放在不同的命名图中，但是如果已经使用命名图来跟踪图的来源(例如)，这可能会变得很麻烦。

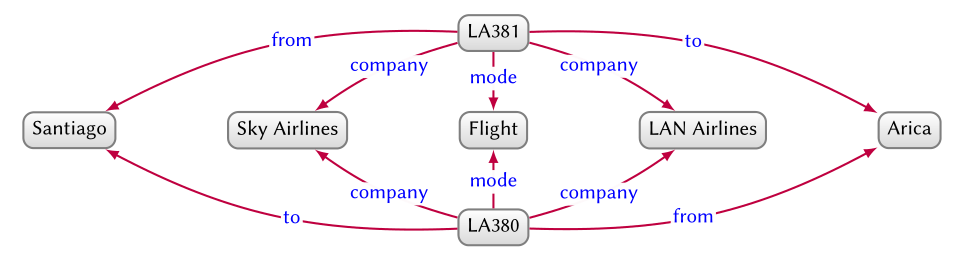
因此，提出属性图模型是为了在将数据建模为图时提供额外的灵活性[14,332]。属性图允许一组属性值对和一个标签同时与节点和边相关联。图4提供了一个例子，再次显示了圣地亚哥和阿里卡之间的航班以及提供这些航线的公司(关于属性图的正式定义，我们参考附录B中的定义B.5)。这一次，我们使用边上的属性-值对来模拟公司以及距离。

图3所示。提供圣地亚哥和阿里卡之间航班的公司的有向边标签图

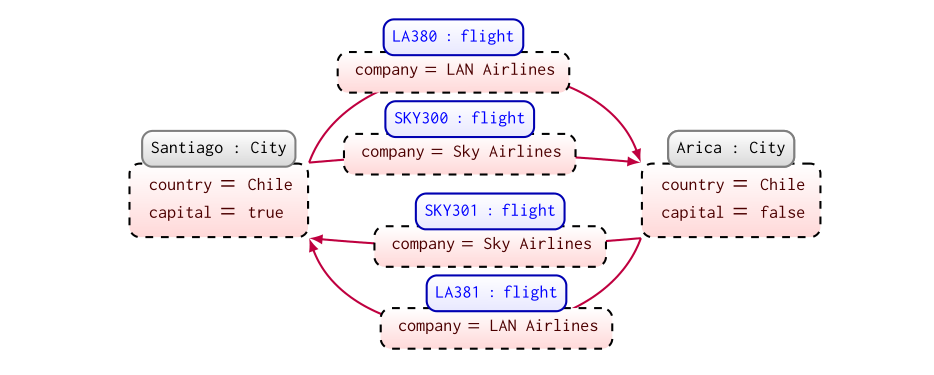


图4所示。提供圣地亚哥和阿里卡之间航班的公司的财产图表

关系的类型由标签飞行捕获。我们进一步建模，圣地亚哥和阿里卡是使用节点标签的城市，以及它们的国家和它们是否为首都，使用节点上的财产-价值对。

属性图最常用于流行的图形数据库，如Neo4j[14，332]。在选择图形模型时，重要的是要注意，属性图可以往返于有向边标记图和图数据集而不丢失信息[16，224]（例如图4）。总之，有向边标记图提供了一个更简单的模型，而属性图提供了一个更灵活的模型。通常情况下，模型的选择将次要于其他实际因素，例如不同模型的可用实现等。

**2.1.4其他图形数据模型**

前三种模型是图形表示的常见示例。其他图形数据模型存在复杂节点，这些节点可能包含单个边[15,212]或嵌套图[15,38]（有时称为超节点[295]）。类似地，超图的数学概念定义了连接集合而不是节点对的复杂边。我们认为，知识图可以基于节点和边采用任何这样的图数据模型：数据通常可以从一个模型转换到另一个模型（见图3和图4）。近年来，实践中最流行的图模型已成为有向边标记图、图数据集和属性图[14]。在本文的其余部分，鉴于有向边标记图的相对简洁性，我们更愿意讨论它们，但大多数讨论自然延伸到其他模型。

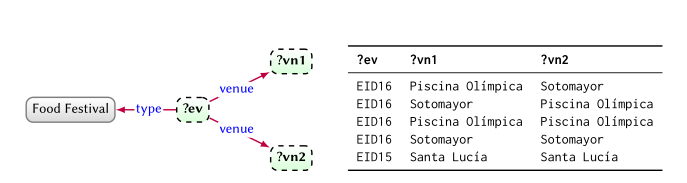


图5.在图1（右）的图形上生成映射的图形模式（左）

**2.2查询**

已经提出了许多用于查询图的实用语言[14]，包括用于RDF图的SPARQL查询语言[207]；Cypher[159]、Gremlin[420]和G-CORE[13]用于查询属性图4。这些查询语言的基础是一些常见的原语，包括（基本）图形模式、关系运算符、路径表达式等[14]。现在，我们从图形模式开始，依次描述查询图形的这些核心功能。

**2.2.1图形模式**

每个结构化图形查询语言的核心是（基本）图形模式[14,96]，它们遵循与被查询的数据图相同的模型（参见第2.1节），另外还允许变量作为术语5。因此，图模式中的5个术语被分为常量（如Arica或venue）和变量（以问号作为前缀），如?event和 ?rel。然后，通过生成从图形模式的变量到数据图中的常数的映射，根据数据图对图形模式进行评估，从而使映射下的图形模式的图像（用指定的常数替换变量）包含在数据图中。

在图5中，我们提供了一个寻找美食节场地的图形模式示例，以及该图形模式生成的与图1数据图的可能映射。在一些呈现的映射（最后两个列出）中，多个变量映射到同一个术语，这可能是可取的，也可能不是，取决于应用。因此，人们提出了许多语义来评估图模式[14]，其中最重要的是：基于同态的语义，它允许多个变量映射到同一个术语，这样图5中显示的所有映射都将被视为结果；以及基于同构的语义，它要求节点和/或边上的变量映射到唯一的术语，从而将图5的后三个映射从结果中排除。不同的实用语言采用不同的语义来评估图形模式，例如，SPARQL采用基于同态的语义，而Cypher采用基于同构（边）的同态语义。

正如我们将在后面的示例中看到的（尤其是图7），图形模式也可能形成循环（无论是有向的还是无向的），并可能用变量替换边标签。通过允许变量替换模型任何位置的术语，可以类似地定义其他模型（如属性图）上下文中的图形模式。我们在附录B.2.1中为有向边标记图和属性图提供了图形模式的形式化及其评估。

**2.2.2复杂的图形模式**

图形模式将输入图形转换为结果表（如图5所示）。然后我们可以考虑使用关系代数来组合和/或变换这样的表，从而从一个或多个图形模式形成更复杂的查询。回想一下，关系代数由接受一个输入表的一元运算符和接受两个输入表的二元运算符组成。一元运算符包括用于输出列子集的投影（π）， 选择（σ）以输出与给定条件匹配的行子集，并重命名列（ρ）。二进制运算符包括并集(∪) 要将两个表的行合并到一个表中，差(−) 删除第二个表中删除第一个表中的行，和连接() 将一个表中的行与另一个表中满足连接条件的行进行扩展。选择和连接条件通常包括等式（=）、不等式(≤), 否定（），析取(∨), 从这些运算符中，我们可以进一步定义其他（语法）操作符，比如交集(∩) 要在两个表中输出行，反连接(▷, 又名不存在）要从第一个表中输出第二个表中没有连接兼容行的行，左连接（，又名可选）执行连接，但保留第一个表中的行，而第二个表中没有兼容行，等等。

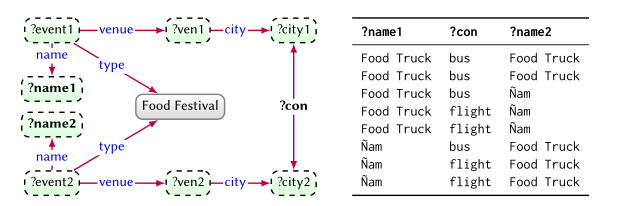


图6.连接查询（左）和图1（右）图上生成的映射

然后，图形模式可以用关系代数的子集（即π、σ、ρ、）表示。例如，假设一个三元关系G（s，p，o）代表一个图，即一个表G有三列s，p，o，图5的查询可以用关系代数表示为：



其中表示自然联接，这意味着在两个表中检查具有相同名称的列对之间的相等性（这里，联接是在主题列s上执行的）。这个查询的结果是一个表，每个变量有一列：ev、vn1、vn2。然而，并非所有在G上使用π、σ、ρ和的查询都可以表示为图形模式；例如，我们不能选择在图形模式中投影哪些变量，而是必须投影所有未固定为常数的变量。

SPARQL[207]和Cypher[159]等图形查询语言允许在图形模式的结果上充分使用关系运算符，从而产生复杂的图形模式[14]。图6展示了一个复杂图形模式的示例，该模式以粗体显示投影变量，选择特定变量以显示在最终结果中。就表达能力而言，具有这种形式的（不受限制的）投影的图模式等同于图上的连接查询。在图7中，我们给出了另一个复杂图形模式的示例，用于查找不在圣地亚哥举办的美食节或饮品节，可以选择返回其名称和开始日期（如果可用）。这样的查询--允许在图形模式之上充分使用关系运算符--等同于对图形的一阶查询。在附录B.2.2中，我们将复杂图模式及其对数据图的评估形式化。

复杂的图形模式可能会导致重复的结果；例如，图6中的第一个结果出现了两次？city1和Arica匹配而和？city2在一个结果中与Viña del Mar匹配，反之亦然。然后，查询语言提供了两种语义：bag语义根据底层映射的多样性保留重复项，而set语义（通常使用不同的关键字调用）从结果中删除重复项。

**2.2.3导航图模式**

区别于图形查询语言的一个关键特性是能够在查询中包含路径表达式。路径表达式r是一个正则表达式，允许在两个节点之间匹配任意长度的路径，它被表示为一条规则查询路径(x, r,y)，其中x和y可以是变量或常量（甚至是相同的项）。基本路径表达式中，r是常数（边标签）。此外，如果r是一个路径表达式，然后r−（逆）6和r∗（克莱恩星：零或更多）也是路径表达式。最后，如果r1和r2是路径表达式，那么r1 | r2（析取）和r1·r2（串联）也是路径表达式。

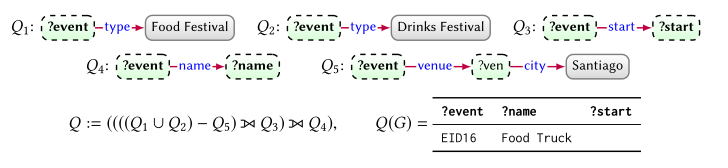


图7.在图1的图上生成映射的复杂图模式（Q）

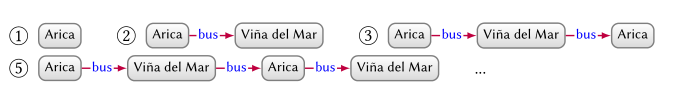


图8.图1（Q（G））中的一些可能路径匹配（Arica、bus\*、？city）

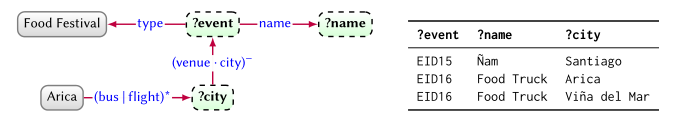


图9.导航图模式（左），在图1（右）的图上生成映射

然后，可以在许多不同的语义下对常规路径查询进行评估。例如，根据图1的图表评估的（Arica、bus\*、？city）可能与图8中的路径相匹配。事实上，由于存在一个循环，因此可能会匹配无限多条路径。出于这个原因，通常会使用受限语义，只返回最短路径，或没有重复节点或边的路径（如Cypher）7。另一个选项不是返回路径，而是返回由匹配路径连接的（有限）节点对集（如SPARQL 1.1）。

然后，可以在图形模式中使用常规路径查询来表示导航图形模式[14]，如图9所示，图9显示了一个查询，该查询搜索可通过巴士或航班从阿里卡（递归）到达的城市中的美食节。

此外，当常规路径查询和图形模式与投影、选择、并集、差分和可选等运算符组合时，结果称为复杂导航图形模式[14]。附录B.2.3提供了（复杂）导航图模式及其评估的定义。

**2.2.4其他功能**

到目前为止，我们已经讨论了图形的任何查询语言的实用性和理论基础的特征[14]。然而，图形的特定查询语言可能支持其他实用功能，例如聚合（分组、计数等），更复杂的过滤器和数据类型运算符（例如，从日期中提取的年份范围查询），联合会通过Web查询远程托管的图形，用于更新图形的语言、对语义蕴涵机制的支持等。有关更多信息，我们参考了各自查询语言的文档（例如[13，207]）和Angels等人[14]的调查。

**B 正式定义**

为了使讨论尽可能容易理解， 本文的主体使用示例驱动的对与知识图相关的主要概念和技术的解释。 在本节中，我们用正式的定义来补充本文的讨论。

**B.1 数据图模型**

我们根据以前的约定（例如， [14]） 定义图形数据模型。虽然在不同的模型中可以使用不同类型的 常量（例如， RDF 允许 IRI 和文字） ，但这些定义使用单个（可数） 无限的常量集， 表示为 Con。 （因此， 我们还抽象出了当前介绍性讨论中不急需的问题， 例如 RDF [235] 中空白节点的存在语义、 文字上的 D 蕴涵 [213]、位置限制 [106] 等）

B.1.1 有向边标记图。我们首先提供有向边标记图的定义。

定义 B.1 （有向边标记图）。有向边标记图是一个元组 G = (V, E,L)，其中 V ⊆ Con 是 一组节点， L ⊆ Con 是一组边标签， E ⊆ V × L × V 是一个集合的边缘。

示例 B.2。参考图 1， 节点集 V 有 15 个元素， 包括 Arica、EID16 等。边集 E 有 23 个三元 组，包括 (Arica,flight,Santiago)。双向边用两条边表示。边标签集 L 有 8 个元素，包括 start、 flight 等。

定义 B.1 没有说明 V 和 L 是不相交的： 尽管在示例中没有出现， 但节点也可以用作边缘标签。 该定义还允许节点和边标签可以在没有任何关联边的情况下出现。任何一个限制都可以在特定应用 中明确说明——如果有必要的话——同时仍然符合有向边标记图。

在下面的一些定义中， 为了便于表示， 我们可以将一组（有向标记） 边 EVLV 视为有向边标记图V、E、L，在这种情⊆况下×，们指的是由 E 诱导的图，假设V 和 分别包)含所有且仅包含 E 中使用边标记图。

B.1.2 图数据集。接下来我们定义一个图形数据集。

定义 B.3（图形数据集） 。命名图是一对 n，G， 其中 是有)向边标记图， n Con 是图名。图数 据集是一对 D∈= GD, N 其中 GD是称为默认图的有向边标记图， N 是空(集或一)组命名图

示例 B.4。 参考图 2， 图数据集 D 由两个命名图和一个默认图组成。默认图形没有与之关联的名 称。两个图名称是事件和路线； 这些也用作默认图中的节点。

尽管定义 B.3 基于有向边标记图，但图数据集的概念可以简单地概括为管理其他形式的图（例如 属性图） 。 RDF 数据集是由 W3C [106] 标准化的图数据集模型的示例， 其中每个图都是 RDF 图， 图名称可以是空白节点或 IRI。图形数据集的定义允许图形名称可以用作图形中的节点或边标签； RDF 数据集也是如此。

B.1.3 属性图。最后， 我们定义一个属性图。

定义B.5（属性图）。属性图是一个元组 G = (V , E,L,P, U , e,l,p)，其中V ⊆ Con 是一组节点 id， E ⊆ Con 是一组边 id， L ⊆ Con 是一组标签， P ⊆ Con 是一组属性， U Con 是⊆一组值， e:E→V\*V 将边 id 映→射到×一对节点 id，l： VE 2L将节点或边 id 映射到∪一组→标签， p:V∪E一2P×U将节点或边ID映射到一组属性值对。

示例 B.6。返回图 4：

• 集合 V 包含 Santiago 和 Arica；

• 集合 E 包含 E1 和 E2；

• 集合 L 包含 City 和 flight；

• 集合 P 包含country, capital, company 和 dist；

• 集合U包含Chile, true, false、Sky Airlines、LAN Airlines、2000km；

• 映射 e 给出，例如， e(E1) = (Santiago, Arica)；

• 映射 l 给出， 例如 l (E1) = {flight} 和 l (Santiago) = {city}；

• 映射 p 给出， 例如 p(E1) = {(company, Sky Airlines), . . . , (dist, 2000 km)} 和 p(Santiago) = {(country, Chile), . . . , (capital, true)}。

定义 B.5 不要求集合 V、E、L、P 或 U 是（成对的） 不相交的： 例如， 我们允许值也是节点。 与之前的一些定义 [14] 不同， 这里我们允许节点或边对给定属性具有多个值。在实践中， 像 Neo4j [332] 这样的系统可能宁愿通过允许一组值来支持这一点。我们将这些变化视为句法。

**B.2查询**

在这里，我们将与图上的查询相关的基本概念形式化，从图模式开始，随后我们将向其中添加关系式运算符和路径表达式。

B.2.1图形模式

我们首先对有向边标记图，然后对属性图，形式化了图模式的概念[14]。对于这些定义，我们引入了一组可数无限的变量Var，范围超过（但不相交于：Con）∩ Var=∅) 常数集。我们通常将常量和变量称为术语，表示和定义为Term = Con ∪ Var.。

**定义B.7**（有向边标记的图形模式）。我们将有向边标记的图模式定义为元组Q=（V′，E′，L′），其中V′⊆ 术语是一组节点术语，L′⊆ 项是一组边项，E′是⊆ V′×L′×V′是一组边（三重模式）。

**例B.8**。回到图5的图形模式：

* 集合V′包含恒定的食物节和变量？事件ven1和？ven2；
* 集合L′包含常数类型和地点；
* the set E′ contains four edges, including (?event, type, Food Festival), etc.

属性图模式以类似方式定义，允许变量位于任何位置。

**定义B.9**（属性图模式）。我们将属性图模式定义为元组Q=（V′，E′，L′，P′，U′，E′，L′，P′），其中V′⊆ 术语是一组节点id术语，E′⊆ 术语是一组边id术语，L′⊆ 术语是一组标签术语，P′⊆ 术语是一组属性术语，U′⊆ 术语是一组值术语，e′：e′→ V′×V′将边id项映射为一对节点id项l′：V′∪ E′→ 2L′将节点或边id项映射到一组标签项，而p′：V′∪ E′→ 2P′×U′将节点或边id项映射到一组属性值项对。

为了定义评估图形模式的结果，我们首先定义部分映射µ：Var→ Con从变量到常量，其域（为其定义的变量集）由dom（µ）表示。给定一个图模式Q，让Var（Q）表示出现在Q中（某个递归嵌套元素）的所有变量的集合。

我们用µ（Q）表示 μ下的Q的图像，意味着任何变量v∈ Var（Q）∩ Q中的dom（µ）替换为µ（v）。观察Var（Q）时⊆ dom（µ），那么µ（Q）是一个数据图（在相应的Q模型中）。

接下来，我们定义了数据图之间包含的概念。对于两个有向边标记的图模式G1=（V1，E1，L1）和G2=（V2，E2，L2），我们说G1是G2的子图，表示为G1⊆ G2，当且仅当V1⊆ V2，E1⊆ E2和L1⊆ L2。39相反，在属性图中，节点通常可以无边定义。对于两个属性图G1=（V1，E1，L1，P1，U1，E1，L1，P1）和G2=（V2，E2，L2，P2，U2，E2，L2，P2），我们说G1是G2的一个子图，表示为G1⊆ G2，当且仅当V1⊆ V2，E1⊆ E2，L1⊆ L2，P1⊆ P2，U1⊆ U2，为所有x∈ 它认为E1（x）=e2（x），并且对于所有的y∈ E1∪ V1它认为l1（y）⊆ l2（y）和p1（y）⊆ p2（y）。

现在我们已经准备好定义图模式的评估

**定义B.10**（图形模式的评估）。设Q为图模式，G为数据图。然后，我们将图模式Q在数据图G上的求值定义为映射集{µ|µ（Q）⊆ G和dom（µ）=Var（Q）}。

**例B.11**。图5列举了在图1的数据图上对所描述的图模式进行评估时给出的所有映射。每个非标题行表示一个映射。

根据语义的选择，评估图形模式的最终结果可能会有所不同：基于同态的语义下的结果被定义为Q（G）。相反，在基于同构的语义下，可能会从结果中排除将两个边变量发送到同一常数的映射和/或将两个节点变量发送到同一常数的映射。从今往后，我们假设更普遍的基于同态的语义。

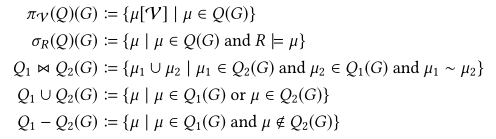
B.2.2复杂的图形模式。我们现在定义复杂的图形模式。

**定义B.12**（复杂图形模式）。复杂的图形模式是递归定义的：

* 如果Q是一个图形模式，那么Q是一个复杂的图形模式。
* 如果Q是一个复杂的图形模式，那么V⊆ 那么πV（Q）是一个复杂的图形模式。
* 如果Q是一个复杂的图模式，R是一个带有布尔和等式连接词的选择条件(∧, ∨, 那么σR（Q）是一个复杂的图形模式
* 如果Q1和Q2是复杂的图形模式，那么Q1 Z Q2，Q1∪Q2和Q1−Q2也是复杂的图形模式。

接下来，我们定义复杂图形模式的评估。首先，给定一个映射µ，对于一组变量V⊆ Var letµ[V]表示映射µ′，使得dom（µ′）=dom（µ）∩ V和µ（V）=所有V的µ′（V）∈ dom（µ′）（换句话说，µ[V]从µ中投影变量V）。此外，让R表示布尔选择条件和µa映射，通过R |=µ，我们表示µ满足布尔条件。最后，我们定义了两个兼容的映射µ1和µ2，表示为µ1∼ µ2，当且仅当所有v的µ1（v）=µ2（v）∈ dom（µ1）∩ dom（µ2）（换句话说，它们将所有公共变量映射到同一常数）。我们现在准备提供定义。

**定义B.13**（复杂图形模式评估）。给定一个复杂的图模式Q，如果Q是一个图模式，那么Q（G）是根据定义B.10定义的。否则：



基于这些查询操作符，我们还可以定义一些额外的语法操作符，比如反连接(▷, aka不存在），或左连接（Z，aka可选）：



我们称这些运算符为语法运算符，因为它们不会向查询语言添加表达能力。

**例B.14**。图7展示了一个复杂的图形模式及其评估。

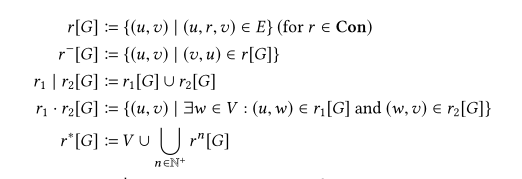
B.2.3导航图模式。我们首先定义路径表达式和常规路径查询。

**定义B.15**（路径表达式）。常量（边标签）c是路径表达式。此外：

* 如果r是一个路径表达式，那么r− （逆）和r∗ （克莱恩星）是路径表达式。
* 如果r1和r2是路径表达式，那么r1·r2（串联）和r1 | r2（析取）是路径表达式。

现在，我们在SPARQL 1.1风格的语义下定义对路径表达式的求值，从而返回路径的端点（开始和结束节点对）[207]

**定义B.16**（路径表达式计算（有向边标记图））。给定有向边标号图G=（V，E，L）和路径表达式r，我们定义r对G的求值，表示为r[G]，如下所示：



其中，r n表示r的第n次串联（例如，r 3=r·r·r）。

属性图G=（V，E，L，P，U，E，L，P）上的路径表达式的计算可以通过调整第一个定义（在r∈ Con）如下：



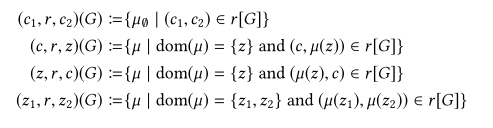
其余的定义则保持不变。

查询语言可能支持其他运算符，其中一些运算符是语法运算符（例如，r+有时用于一个或多个运算符，但可以重写为r·r）∗), 而其他人可能会增加表达能力，如SPARQL[207]的情况，它允许表达式中的有限形式的否定（例如，！r，r是常数或常数的逆，匹配任何未标记r的路径）。

接下来，我们定义一个常规路径查询及其求值。

**定义B.17**（常规路径查询）。常规路径查询是一个三元组（x，r，y），其中x，y∈ 骗局∪ Var和r是一个路径表达式。

**定义B.18**（常规路径查询评估）。设G表示有向边标记图c，c1，c2∈ Con表示常数，z，z1，z2∈ Var表示变量。然后，常规路径查询的计算定义如下：



∅ 表示空映射，以便dom（µ）=∅ （加入身份）。

**定义B.19**（导航图形模式）。如果Q是一个图形模式，那么Q是一个导航图形模式。此外，如果Q是一个导航图模式，（x，r，y）是一个常规路径查询，那么qz（x，r，y）是一个导航图模式。

然后，导航图模式的求值定义遵循之前的连接定义和常规路径查询求值的相应定义（分别适用于有向边标记图或属性图）。同样，复杂的导航图模式——及其评估——是通过使用定义B.12中相同的运算符，按照定义B.13中相同的语义，以自然的方式扩展该定义来定义的。