文章编号:1671-8860(2005)12-1104-03　　文献标志码:A

关系数据库管理空间数据的方式下数据查询方法的研究

黎展荣[1,2]　杨如军[2]　周新忠[1]

(1　武汉大学遥感信息工程学院,武汉市珞喻路 129 号,430079)
(2　南宁市国土资源局,南宁市东宝路 3 号,530022)

摘　要:从逻辑上概括了基本的二维空间关系,描述新增的查询需求,阐述如何通过关系运算,推演出正确结果,在文章最后给出具体的例子,并对查询的问题作了些简单的总结。

关键词:空间数据;关系数据库;地理信息系统;查询方法
中图法分类号:P208

1　基本空间数据类型和空间关系

1.1　基本空间数据类型

关系数据表中的每一列数据都有特定的数据类型。空间数据类型的种类非常多,最基本的类型无非是点、线、面等。在 Oracle 9i 中,其 Oracle Spatial 考虑了如图 1 所示的空间类型数据,而有一些 GIS 和 CAD 软件所定义的空间数据类型更为复杂,如 CAD 软件中定义有各种样条曲线、多义线、复合对象等。

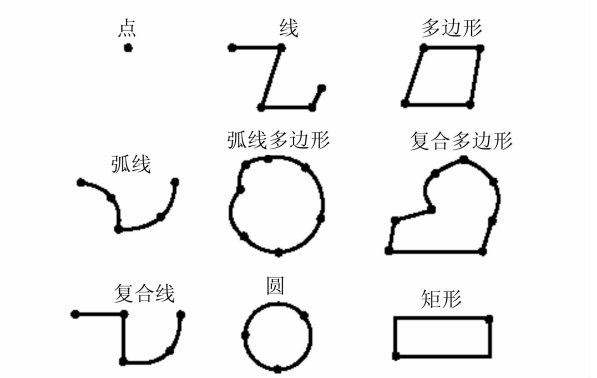


图 1　基本空间数据类型
Fig. 1　Geometric Types

1.2　空间关系

关系数据库管理数据的出发点是从对象的关系出发,本文只讨论基本的点、线、面数据,从其空间分布的特点和组合情况,就不难发现空间关系的多样性和复杂性。

Egenhofer 教授及其同事们所做的研究中,对基本空间对象的拓扑关系作如下的描述。DISJOINT:相离(不相交);TOUCH:相接触;OVERLAPBDYDISJOINT:两个实体相交,但它们的边缘不相交;OVERLAPBDYINTERSECT:两个实体相交,并且它们的边缘也相交;EQUAL:两个实体完全互相重合;CONTAINS:包含;COVERS:一个实体被另一个实体的边界范围包含;INSIDE:涵义与 CONTAINS 正好相反。"*A* INSIDE *B*" 意味者 "*B* CONTAINS *A*";COVEREDBY:涵义与 COVERS 正好相反。"*A* COVEREDBY *B*" 意味者 "*B* COVERS *A*";ON:*A* 实体位于 *B* 实体之上,并且实体 *B* COVERS *A*;ANYINTERACT:实体 *A* 与 *B*,不属于 *A* DISJION *B* 的情况,就是 ANYINTERACT。

笔者认为,这种关系的描述只是基本情况,若要拓展,还可以更复杂,如 CONTAIN 的情况,还可以细分为内含和内邻接两种,如图 2 所示。在 Oracle Spatial 中,除了定义上述的空间关系外,还采用了 Nine-Intersection Model 的方式来描述空间关系,并且通过诸如 SDO_RELATE 或 SDO_GEOM.RELATE 等方法来获取空间关系类型,其具体内容请参阅 Oracle Spatial 产品的相关文档。

收稿日期:2005-11-18。

图 2　内含

Fig. 2　Contain

2　空间数据的组织与空间查询

点、线、面是基本的空间数据要素,但要再现更复杂丰富的信息,还需要大量的扩展。在 ArcGIS 软件里,feature 是绘图的最基本对象,Data 中有多个 feature,Data Frame 中包含了多个 Data,Data 可以是来自 shape file 的数据,也可以是来自 SQL Server、Oracle 这样数据库链接中的一个表。Data 中的数据类型是单一的,它可以加到 Layers 中,在一个视图中来显示。Data 是单一类型的 N 个 feature 的集合($N\geqslant 0$),Data Frame 是 N 个 Data 的集合($N\geqslant 0$),一个 Layer Group 对应于一个 Data 的数据,一幅 Map View 是 N 个 Layer Group 的集合($N\geqslant 0$)。

用户可以不关心空间数据在物理上的组织,而应从认知上形成一个清晰稳定的概念模型,而这些正是最终建立严密物理模型的基础。

基于以上的认识,概念模型可作如下的定义。

1) 定义最基本地理实体:GeoAtom{点、线、面…}。

2) 定义 GeoSet{obj|obj∈GeoAtom}。

3) 定义 GeoSet_1{obj|obj∈GeoSet}。

4) ……

5) 依次类推,有 GeoSet_n{obj|obj∈GeoSet_$n-1$},$n>1$。

6) 定义 TGeoSet{Geoobj、GeoSet_1、GeoSet_2、….. GeoSet_n}。

7) Type(obj)取得 obj 的类型。

8) 定义 GeoSet$T${obj|Type(obj)∈TGeoSet()}。

9) 定义 GeoData{obj| obj∈GeoSet$T$()}。

10)定义 GeoData_1{obj| obj∈GeoData()}。

11) ……

12) 定义 GeoData_n{obj| obj∈GeoData_$n-1$()}。

上面的定义没有将更具体的对象限制条件列出来,但这些定义已经足够复杂地来进行地理空间世界的描述。

这里定义的空间地理概念是嵌套的可扩展的对象模型,空间数据结构树如图 3 所示。

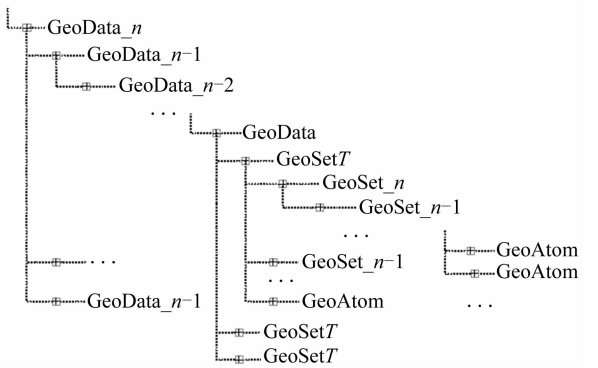


图 3　空间数据结构树

Fig. 3　Spatial Data Tree

对于这样的树型,在实际应用中应考虑定义几个层次、定义怎样的名称、增加的约束条件。

对于类型为 GeoAtom 的 obj$A$、obj$B$。用函数 Dist(obj$A$,obj$B$)来表示 obj$A$ 与 obj$B$ 的距离,OverLay(obj$A$,obj$B$)表示 obj$A$ 与 obj$B$ 的重合情况,其取值如表 1 所示。

表 1　OverLay

Tab. 1　OverLay

点与点	点与线	点与面	线与线	线与面	面与面
相离,0	相离,0	相离,0	相离,0	相离,0	相离,0
重合,1	重合,1	点落在面边缘,1 点落在面内,2	完全重合,1 部分重合,2 相交,3	内含,1 重合,2 相交,3 相触,4	内含,1 完全重合,2 相交,3 相触,4

3　根据空间关系进行关联操作与数据更改

在关系数据模型中,关联运算非常重要,是在两个二维表间进行的,对于地理实体,还可以延拓这样关系运算的内容,那就是根据空间关系进行关联操作,这与根据函数关系进行表关联的思想非常相似,只是运算的方式大不相同。

关系数据库中的关联操作,为 $R\overset{\infty}{\underset{条件}{}}S$。

设关系表 R(id,SF$d1$)、S(id,SF$d1$)。其中 id 为长整型标识符,SF$d1$ 为变长数据类型字段,存放空间信息。

在一般的关系数据库应用中,关联的条件大多是=、≥、≤、≠等关系运算,但空间关系就复杂得多。如要求的条件为:R 中的地理实体与 S 中的地理实体,它们的距离为 0,即 Dist(R.SF$d1$,S.SF$d1$)=0。

这样的关联,可以理解成:① 作 R 与 S 的笛

卡儿乘积，可以得到集合 A：$A=R\infty S$；② 在乘积结果中，根据条件进行选择，可以得到集合 B：$B=C|_{(\mathrm{Dist}(R.\mathrm{SFd1},S.\mathrm{SFd1})=0)}(A)$。

其他的空间关系也是类似的。

4 空间关系与属性关系的联合操作

在实际应用中，空间属性和非空间属性是经常结合的，而采用“对象关系型”管理空间数据的方式，在这点就有更为优越的集成性。

这里给出一个比较复杂的例子，要处理的数据有：土地分类图斑、道路、河流、行政区。建立 4 个基本的数据表：Tb_Land、Tb_Road、Tb_River、Tb_Region，如图 4 所示。每个数据表都有自增长类型的字段 id 作为主键。

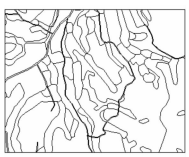
1）检查土地分类图斑中是否有重叠的图斑。

① 集合 A=Tb_Land。

② 集合 B=Tb_Land。

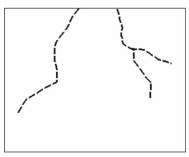
③ 根据空间关系，对 A 与 B 进行关联。

关联条件：obj$A\in A$，obj$B\in B$，OverLay(obj$A$，obj$B$)$\in\{1,2,3,4,7\}$。

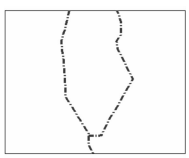


1 土地分类图斑(Tb_Land)
1）面状分布，不同图斑有地类属性；
2）图斑不允许重叠，并且铺满整个区域。
用面类型 GeoAtom 来表示图斑，一个图斑实体具有下列属性：
1）空间属性(SFd)；
2）土地分类(class)。

2 道路(Tb_Road)
1）线状分布，每段道路有属性；
2）道路可以相交，但不允许重叠。
用线类型 GeoAtom 来表示道路，一个道路实体具有下列属性：
1）空间属性(SFd)；
2）路宽(Width)；
3）类型(Type)。



3 河流(Tb_River)
1）河流呈线状分布；
2)河流允许相交，但不能有重叠。
用线类型 GeoAtom 来表示河流，一个河流实体具有下列属性：
1）空间属性(SFd)；
2）河流宽度(Width)。



4 行政区划(Tb_Region)
1）面状分布；
2）没有重叠，并且铺满整个区域。
用面类型 GeoAtom 来表示行政区，一个行政区实体具有下列属性：
1）空间属性(SFd)；
2）行政区名称(Name)。

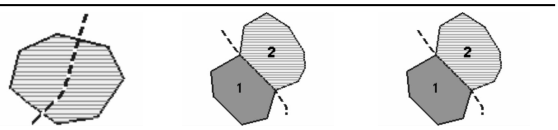
图 4 数据表
Fig. 4 Tables of Data

④ 在 R 上，对 A. id、B. id 作投影，$R_1=\Pi_{(A.\mathrm{id},B.\mathrm{id})}(R)$。

那么 R_1 就反映了 Tb_Land 中，哪个 id 字段对应的图斑有重叠。

2）土地分类图斑面积的扣除。

对于土地分类图斑，面积计算规则限定如图 5 所示。



河流穿越，图斑要扣除河流面积，河流面积为公共部分长度×宽度	河流与 1、2 图斑的公共边重合，这时，只有图斑 1 需要扣除河流的面积	河流与 1、2 图斑的公共边重合，这时，只有图斑 1、2 分布扣除河流的面积的 1/2

图 5 计算规定
Fig. 5 Calculate Rules

要得到各土地分类的面积总和有多种方法，关键是能正确判断线与面的关系，包含部分的线，临边部分的线，扣除方法是不同的，只有区分清楚不同的情况，才能得到正确的结果。

3）不考虑重合的情况，统计每个行政区中的土地分类面积。

① 给出一个行政区面 obj$R$；

② 集合 A=Tb_Land；

③ 在 A 中选择与 obj$R$ 有重叠的记录，obj$A\in A$，$B=C|_{\mathrm{Dist}(A,\mathrm{obj}R)=0}(A)$；

④ 根据 obj$R$，更改集合 B 中的空间数据，obj$B\in B$，用 obj$R\cap$obj$B$ 来替换对应记录的空间数据；

⑤ 利用关系数据库查询语言，对 B 中的分类字段进行统计。

4）其他命题。

① 检查是否有重叠的道路；

② 考虑面积扣除情况，统计各行政区的土地分类面积；

③ 道路所占的图斑的分类面积统计。

参 考 文 献

1 吴立新，史文中. 地理信息系统原理与算法. 北京：科学出版社，2003

2 郭仁忠. 空间分析. 北京：高等教育出版社，2001

3 王 珊，李盛恩. 数据库基础与应用. 北京：人民邮电出版社，2002

4 陈常松. 面向数据共享的 GIS 语义表达理论的初步研究：[博士论文]. 北京：中国科学院，1999

(下转第 1119 页)

表,类之间关系图如图 2 所示。

1) 过程类

- 过程实例类:WEProcessInstance;
- 子过程实例类:WESubProcessInstance;
- 过程定义类:WDProcess。

2) 节点类

- (通用)节点实例类:WEActivityInstance;
- 手动节点实例类:WEManualActivityInstance;
- 自动节点实例类:WEAutoActivityInstance;
- 子过程节点实例类:WESubProcessActivityInstance;
- 路由节点实例类:WERouteActivityInstance;
- 节点定义类:WDActivity。

3) 工作项类

- 工作项类:WEWorkItemInstance;
- 子工作项类:WESubWorkItemInstance。

4) 对象管理类

- 对象管理器:WEWorkflowObjectManager。

5) 持久化类

- 过程持久化类:WEProcessInsPersist;
- 节点持久化类:WEActivityInsPersist;
- 工作项持久化类:WEWorkItemInsPersist。

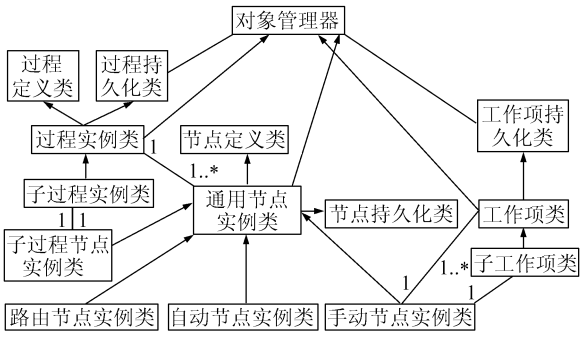


图 2　引擎中类关系图

Fig. 2　Class Relationship Map in the Engine

类的设计采用了面向对象的继承方法,在一定程度上屏蔽了节点类型之间的差别。如一个节点挂起的操作,在调用过程中不必区分是什么类型的节点对象,只要调用通用的接口就可以了。对象管理器在负责装配对象时需要对不同的类型创建不同的实例。

在工作流运行的对象生存周期中,当流程被创建后,流程便从静态的定义状态变为动态可流转状态。流程定义存在两种逻辑单元:过程和节点的定义,以及过程的数据变量。当流程成为可流转的过程时,生成过程实例,而节点是否能成为实例状态,则要由过程流转的路径来决定。而可流转的实例对象也有活跃的内存状态和钝化的数据库状态两种状态。以节点为例,当前节点完成后,过程根据定义路由到下一个节点,这时需要将该节点从定义态转成为可流转的状态,过程向 Object Manager 发出创建节点的请求,Object Manager 根据请求类型生成相应空的对象实例,对象实例根据定义标识进行原始数据初始化,并获得过程传递来的相关数据,接着将所有的数据通过持久化对象写入数据库,并启动逻辑功能服务,创建工作项或进行路由。当前任务完成后,如果节点完成则将自己从运行表中删除,并退出内存状态,否则直接退出内存状态。仅退出内存的节点,当过程再次需要时,向 Object Manager 发出 load 对象的请求。Object Manager 通过持久化层将数据从数据库中取出,创建对象并装配数据,使其成为内存对象。对于完成的节点,在被删除前会将完成的信息记录日志。过程和过程相关数据、节点和节点相关数据、工作项都会被记入日志。

2.3.2　工作流服务

结合以上类设计构建工作流引擎服务,具体包括以下几个方面。

1) 工作流执行服务(workflow execution service)。工作流执行服务是系统的核心部分,主要负责流程实例的运转和对象的状态转换,支持群集运算。

2) 管理/监控服务(admin service)。负责对各服务的监控,管理各服务的状态,可自动对异常状况进行报告。

3) 定时服务(scheduling service)。定时服务根据系统的配置,周期性地启动或调度相应的系统进程来完成某些特殊的任务。如间隔一定的时间,检测流程中是否有已经超时的节点、更新应用服务的注册信息、与业务系统交换数据等等。

4) 归档服务(archive service)。归档服务是将已完成的流程实例及其日志记录从当前运行库导出的过程。用户也可以选择导出某时刻以前启动的所有流程实例。归档服务通过保证引擎数据库工作在一个健康的记录数内,来保障数据安全和引擎的运行效率。

5) 网关服务(gateway service)。网关服务是对消息传递系统的补充。众所周知,不是所有发生在工作流管理系统和其他外部系统之间的通信