多媒体对象查询语言及其查询处理*

田增平 党华锐 周傲英 施伯乐

(复旦大学计算机科学系 上海 200433)

摘要 文章研究了多媒体数据库的查询需求,提出结构化的多媒体对象查询语言MOQL(multi-media object query language).它能够支持基于类型、结构特征、同步关系、时态关系和内容信息的多媒体查询.以DB2数据库为存储机制,定义了一组代数算子和变换规则,利用它们可以将用户定义的MOQL查询变换为代数表达式,进行代数优化,并将代数查询表达式转换为能够在DB2数据库上运行的DB2SQL和C++查询过程.

关键词 多媒体数据库,查询语言,查询优化.

中图法分类号 TP311

自从"多媒体"概念出现以来,多媒体信息查询技术就成为数据库等相关领域研究的重要课题,取得了很多成果.随着研究工作的进展,多媒体信息的查询需求以及查询语言的功能结构也逐渐明确起来.D. Woelk[1]和C. Goble等人[2]指出,基于结构和类型的查询是多媒体信息查询的重要特征.T. D. Little[3]和A. Karmouch等人[4]提出多媒体数据库应当支持基于多媒体数据表现的时间同步关系的查询.基于结构和类型的多媒体查询使得多媒体数据组织的元数据可以被查询,为用户提供了更大的信息查询空间.

多媒体数据库将传统数据库基于属性值的查询与信息检索系统基于内容的查询结合起来.E. Bertino等人的MULTOS[5]支持对图像、文本、声音的存在性条件查询.M. Flicker等人[6]开发的QBIC系统能根据图像的形状、纹理、颜色等内容进行查询.A. Gupta等人[7]开发的VIMSYS系统能根据给定的特征查询图像.Wu等人[8]开发的CORE系统能够按文本的相似度和模糊度等查询图像.E. Oomoto等人[9]研制的OVID系统能根据动态视频影像的描述特征进行查询.内容查询使得传统数据库的确定查询变为近似或模糊查询.

在研究前人工作时我们注意到,以往提出的各种多媒体查询语言都是针对专门的应用,使用范围较窄,而且各种语言的语法形式与人们熟悉的数据库语言如SQL差别较大,不易学习和使用,另外,各种语言的描述能力也不尽相同.本文以数字图书馆应用为背景,研究了多媒体数据库查询的需求,提出结构化的多媒体对象查询语言MOQL(multi-media object query language).它能够支持基于类型、结构特征、同步关系、时态关系和内容信息的多媒体查询.我们以DB2数据库为存储机制,定义了一组代数算子和变换规则.利用它们可以将用户定义的MOQL查询变换为代数表达式,进行代数优化,最后,将代数查询表达式转换为能够在DB2数据库上运行的查询过程.本文的工作已用于我们开发的中国历史地图数字图书馆系统.

本文第1节讨论多媒体查询的需求.第2节介绍MOQL语言依赖的多媒体对象数据模型.第3节给出MOQL语言的定义.第4节讨论MOQL语言的查询处理和优化技术.第5节总结全文工作,提出需要进一步研究的问题.

---

* 本文研究得到国家自然科学基金、国家863高科技项目基金和国家教育部博士点基金资助.作者**田增平**,1968年生,博士,讲师,主要研究领域为面向对象数据库,多媒体数据库.**党华锐**,1964年生,博士,副教授,主要研究领域为数据库,网络通信.**周傲英**,1965年生,博士,教授,博士生导师,主要研究领域为数据库理论及应用.**施伯乐**,1936年生,教授,博士生导师,主要研究领域为数据库理论及应用.

本文通讯联系人:田增平,上海200433,复旦大学计算机科学系

本文1998-04-16收到原稿,1998-08-04收到修改稿

1 多媒体信息的查询需求

在多媒体信息查询技术的研究中，根据系统目标和应用类型的不同，提出了各种不同的需求. 概括来说，多媒体数据库系统应当支持基于类型、结构、多媒体数据表现的同步关系、多媒体对象内容及时态信息的查询.

(1) 基于类型和结构的查询，支持用户按照对象类型和内部结构特征查询多媒体对象. 在多媒体数据库中，描述多媒体数据组织的类型和结构信息是用户所关心的，对查询多媒体对象具有重要作用. 例如，“查询所有历史地图”和“查询所有含有声音的历史地图”等是基于类型和结构的多媒体查询.

(2) 基于同步关系的查询是以多媒体对象表现的时间特性或不同属性表现的时序关系为条件的查询. 多媒体对象表达的信息不但取决于各种属性数据的内容，还取决于这些数据表现的时序关系. 因而，支持基于同步关系的查询是多媒体查询特有的要求. 例如，“查询伴音解释先于动态视频影像结束的电视片断”等均属于同步关系查询.

(3) 基于内容的查询根据非结构化数据，如图像、动态视频影像和文本等的语义信息查询多媒体对象. 传统数据库查询基于对属性值谓词布尔组合的准确计值. 在多媒体数据库中，用户希望根据图像、视频影像和文本等非结构化对象所富含的语义信息进行查询.

(4) 基于时态信息的查询能够根据某些时间条件查询多媒体对象. 我们研究的多媒体数据库用于存储有关历史地图的多媒体信息，因而，支持基于时态信息的查询是必不可少的功能.

多媒体数据库的查询需求多种多样，而且随新型应用的出现在不断发展和变化，因此，它们是无法完全列举的. 本文根据数字图书馆应用的需要，结合前人的研究成果，提出上列 4 种主要功能要求，并设计了能够满足这些要求的 MOQL 语言.

2 多媒体对象数据模型

MOQL 语言基于我们设计的多媒体对象数据模型，该模型具有如下特点：

(1) 多媒体信息的内容和行为建模. 多媒体对象、类型、类以及类层次和方法等是多媒体对象数据模型的主要概念. 一个多媒体对象由对象标识符(OID)、一个元组类型值、各同步表现属性的时间调度及一个有效历史时间区间组成. 类型、类及类层次是多媒体对象的组织概念. 类型定义一组具有相似结构的多媒体对象的公共结构. 类是一组对象的集合. 类层次是类的组织形式，系统中所有的类形成一个有向无环图. 子类继承父类的结构、行为和表现特征. 方法描述了多媒体对象的行为特征. 系统中支持两类方法，即服务器方法和平台无关方法. 服务器方法与一般对象系统的方法相似，任何用户对方法的调用都会使相应方法在服务器上执行. 平台无关的方法主要是多媒体对象的表现过程. 任何用户对多媒体对象的表现要求都导致相应方法在用户机器执行.

(2) 多媒体信息的表现特征建模. 表现特征建模主要体现在表现的空间和时间特性建模方面. 为了实现表现的平台无关性，我们引入 Java 语言定义多媒体对象的表现过程. 以 Java 语言定义对象方法，实现对象方法的跨平台执行，是数据库与 Internet 网络相结合的新的尝试.

(3) 在表现的空间和时间建模方面，也做了有益的尝试. 多媒体对象表现的空间特征由 Java 语言定义的表现方法描述，即用过程而不是用数据来表达空间特征. 用户可以利用具有完全计算功能的 Java 语言(而不是某种 Form)来安排多媒体对象的表现空间. 对多媒体对象表现的时间特性采用数据描述而不是过程方式建模. 多媒体对象创建者不但要定义多媒体对象的内容，还要定义内容表现的时间顺序关系.

3 多媒体对象查询语言 MOQL

3 种数据库访问形式

MOQL 支持 3 种数据库访问形式，即按 OID 查询、按名称查询以及 SELECT 查询.

(1) 按 OID 查询

在多媒体数据库系统中，每个对象、类型定义、类定义以及类的外延都被视为内部对象. 每个内部对象都由

一个唯一的 OID 标识.用户可以通过 OID 直接检索所要查询的对象信息,例如,#130,#208.

(2) 按名称检索

在多媒体数据库系统中,允许显式地给对象、类外延、类型或类的定义命名,这个名字与对象取值无关.一个对象可以有 1 个或多个名字,但是,一个名字只能赋予一个对象.用户可以根据对象的名称获得对象的有关信息.例如,"M_QuanTu"如果是明代全图对象的名字,则用户利用它就可获得明代全图的有关信息.

(3) SELECT 语句

SELECT 语句支持用户基于内容、时间信息、结构和同步关系的多媒体查询.例如,下列查询:"找出文字解释中含有'历史悠久'且图像颜色与 sampmap 的颜色的相似度大于 0.7 的清代历史地图,要求它们不但含有动画(animClass 为动画数据类型),而且声音解释与文字解释同时开始表现,查询结果按照与 sampmap 图像颜色的相似度,从大至小输出前 5 个对象",可以写成如下的 MOQL 语句:

```
SELECT m
FROM m IN mapClass
WHERE SUBSTRING (m.exptxt,"历史悠久")
        and Grade(m.map.color, sampmap.color)>=0.7
WHEN IN_t(m, 'qingDynasty')
STRUCT m CONTAINS (animClass)
SYNCH m.exptxt STARTS m.expvoc
ORDER [5] ASC (Grade)
```

其中 SELECT,FROM,WHERE,WHEN,STRUCT,SYNCH 和 ORDER 为保留字;SUBSTRING 和 Grade 为用户或系统定义的谓词和函数;IN_t,CONTAINS 和 STARTS 为内部谓词.

3.2 SELECT 语句

SELECT 语句是用户访问多媒体数据库的主要形式.它支持基于内容、时间信息、结构和同步关系的查询,其形式如下:

```
SELECT (路径表达式)
FROM 变量名称 IN 类名称
[WHERE  内容条件 ]
[WHEN   时间条件 ]
[STRUCT 结构条件 ]
[SYNCH  同步条件 ]
[ORDER  [返回结果数目] 结果排序条件 ]
```

其中 SELECT FROM 子句说明查询的目标及范围,WHERE,WHEN,STRUCT 和 SYNCH 分别说明查询结果应当满足的内容条件、时间条件、结构条件和同步条件.

(1) SELECT 及 FROM 子句

SELECT 及 FROM 子句说明查询的结果及查询范围.查询结果可以由一个或多个路径表达式描述.

定义 1. 路径表达式定义如下:

① 若 X 为具有类 T 的对象变量,则 X 为路径表达式,T 为其类型.

② 若 $A_0,A_1,\ldots,A_i(i\geqslant 0)$ 为路径表达式,A_i 是取值类型为 T 的属性,A 是类型 T 的属性,则 $A_0,A_1,A_2,\ldots,A_i.A$ 亦为路径表达式,其类型为 A 的类型.

③ 若 $A_0,A_1,\ldots,A_i(i\geqslant 0)$ 为路径表达式,A_i 是值域为类 C 的属性,f 是类 C 上的方法,f 的参数类型为 $T_1,\ldots,T_n$,而 $a_1,\ldots,a_n$ 分别是具有类型 $T_1,\ldots,T_n$ 的值,则 $A_0,A_1,\ldots,A_i.f(a_1,\ldots,a_n)$ 亦为路径表达式,f 的返回值类型为其类型.

(2) WHERE 及 ORDER 子句

WHERE 及 ORDER 子句说明查询目标应当满足的内容条件以及返回给用户的查询结果数目及其顺序.

定义 2. 内容条件式定义如下：

① 若 P_1 和 P_2 为路径表达式，$op\in\{>,<,\neq,=,\geqslant,\leqslant\}$为一算术比较符，$a$ 为常量，则 $P_1(op)a$，$a(op)P_1$ 和 $P_1(op)P_2$ 均为内容条件式(a 与 P_1 的类型兼容)；

② 若 P_1 和 P_2 为路径表达式，a 为常量，Q 为用户定义谓词或系统谓词，则 $Q(P_1,a)$和 $Q(P_1,P_2)$为内容条件式. MOQL 支持 DB2SQL 所支持的所有内部谓词；

③ 若 C_1 和 C_2 为内容条件式，则 C_1 and C_2，C_1 or C_2 和 not C_1 都为内容条件式.

ORDER 子句的结构为

ORDER [k] ASC|DESC ($f_1,f_2,\ldots,f_n$)，其中：

- k 为返回给用户的结果对象数目；
- ASC 或 DESC 分别说明按 $f_1,f_2,\ldots,f_n$ 函数值的增序或降序输出结果；
- $f_1,f_2,\ldots,f_n$ 为出现在 WHERE 子句中的模糊度或相似度函数.

在查询模型中，内容相关的模糊性及相似性度量函数作为相应类上的方法，因而，它们蕴含于路径表达式中.

(3) WHEN 子句

WHEN 子句说明用户对查询结果的时间条件要求. MOQL 仅支持历史时间信息的查询. 为了表达和操作时间信息，定义下列算子：

BeginOf(一元)，EndOf(一元)，Extend(二元)，SameTime(二元).

其直观解释如下：

- BeginOf(a)：给出 a 的开始年代，例如，BeginOf('qingDynasty')=1618；
- EndOf(a)：给出 a 的结束年代；
- a Extend b：a 与 b 的时间区间的并；
- a SameTime b：a 与 b 的时间区间的交.

定义时间项如下：

① 对象变量和时间常量，如 'Now'，'qingDynasty' 等为时间项；

② 若 t_1,t_2 为时间项，$f\in$ {BeginOf，EndOf}，$g\in$ {Extend，SameTime}，则 $f(t_1)$，$g(t_1,t_2)$为时间项.

MOQL 中提供 4 个二元时间谓词，$BEFORE_t$，$AFTER_t$，IN_t，$OVERLAP_t$：

- $BEFORE_t(a,b)$：a 的时间区间先于 b 的时间区间；
- $AFTER_t(a,b)$：a 发生的时间区间后于 b 的时间区间；
- $OVERLAP_t(a,b)$：a 的时间区间与 b 的时间区间重迭；
- $IN_t(a,b)$：a 的时间区间包含于 b 的时间区间.

定义 3. 时间条件式定义如下：

① 若 a 和 b 为时间项 $P\in\{BEFORE_t,AFTER_t,IN_t,OVERLAP_t\}$，则 $P(a,b)$为时间条件式；

② 若 S 和 Q 为时间条件式，则 S and Q，S or Q 及 not Q 为时间条件式.

(4) STRUCT 子句

MOQL 语言中引入 STRUCT 子句来说明查询目标的结构特征. 它仅定义了一个结构谓词 CONTAINS 来描述结构条件.

定义 4. 结构条件式定义如下：

① 若 P 为路径表达式，$T_1,\ldots,T_n$ 为类或类型名称，则 P CONTAINS$\{T_1,T_2,\ldots,T_n\}$为结构条件式；

② 若 S_1,S_2 为结构条件式，则 S_1 and S_2，S_1 or S_2 及 not S_1 亦为结构条件式.

(5) SYNCH 子句

根据基于时间间隔(TIB)的同步关系表示方法，多媒体数据表现的同步关系可表示为时间间隔的时序关系. J. F. Allen 指出[10]，两个时间间隔之间有 7 种不重复关系，分别是 BEFORE，MEETS，OVERLAPS，DURING，STARTS，FINISHES 和 EQUALS. MOQL 语言支持基于这些同步关系的多媒体查询.

定义 5. 同步条件式定义如下：

① 若 P_1 和 P_2 为路径表达式，$R\in$ {BEFORE, MEETS, OVERLAPS, DURING, STARTS, FINISHES 和 EQUALS}，则 $P_1(R)P_2$ 为同步条件式；

② 若 P_1 和 P_2 为路径表达式，N 为正整数，$op\in\{>,<,\neq,=,\leqslant,\geqslant\}$，则 Length($P_1$)($op$)Length($P_2$)，Length($P_1$)($op$)$N$ 和 N(op)Length(P_1)为同步条件式；

③ 若 S_1 和 S_2 为同步条件式，则 S_1 and S_2，S_1 or S_2 及 not S_1 为同步条件式.

上述定义中，Length 为系统定义函数，它返回多媒体对象的表现时间长度(该对象在表现调度中定义的表现时间长度).

4 查询处理

MOQL 查询处理系统以 DB2 V2 为存储机制，利用关系表格存储结构化数据，以文件机制存储非结构化的对象数据，如图像、文字和声音等，利用用户定义函数(UDF)实现相似或模糊测度函数，支持基于内容的多媒体查询. 它以 MOQL 语言为源语言，以 DB2 SQL 及 C_{sec}^{++}语言为表达数据存取方案的目标语言. 查询处理由以下 7 部分组成：语法检查、命名变换、查询改写、代数变换、优化处理、生成存取方案和执行存取方案.

4.1 语法检查

语法检查具有 3 种功能：(1) 对用户基于 OID 的查询，获取 OID 的类信息并生成查询结果描述结构，将 OID 及查询结果描述结构送给 OID 检索模块的结果输出过程；(2) 对用户基于对象名称的查询，查询"OID-对象名称"对照表，获取对象的 OID 及其类信息，生成查询结果描述结构，将其与 OID 送给结果输出过程；(3) 对用户的 SELECT 查询，检查用户查询语句是否合乎 MOQL 的语法. 它交给命名变换的 SELECT 查询应当符合 MOQL 语法定义.

4.2 命名变换

命名变换的功能是消除变量，并使所有路径表达式都以类名称开始.

4.3 查询改写

查询改写生成查询结果的描述结构，并将 SELECT 查询的目标改写为路径表达式公共类的对象.

4.4 代数变换

代数变换将上述改写的查询结果变换为一个代数表达式. 由于已有的对象代数不便于表达 MOQL 查询，不易被 DB2 SQL 语言实现，而传统的关系代数又不能充分体现 DB2 SQL 的能力，因而，我们定义下列一组代数算子，作为查询的内部表达语言.

- 投影 PJ[A]：PJ[A](R)=$\Pi_A(R)$；
- 选择 SL[P]：SL[P](R)=$\sigma_p(R)$；
- 自然连接 NJ[A_1,A_2]：NJ[A_1,A_2](R_1,R_2)=$R_1\bowtie_{A_1=A_2}R_2$；
- 并 UN：UN(R_1,R_2)=$R_1\cup R_2$；
- 交 INT：INT(R_1,R_2)=$R_1\cap R_2$；
- 差 DF：DF(R_1,R_2)=R_1-R_2；
- 扩充 EXTEND[A_1,...,A_n;f_1,...,f_n](R)将 R 扩充列 A_1,...,A_n，其取值为函数 f_1,...,f_n 作用于 R 元组的结果(f_1,...,f_n 定义在 R 的属性上)；
- 查找 LOOKUP[A_1,A_2,...,A_{2k},P]：

$$\text{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](R_1,R_2,\ldots,R_{k+1})$$
$$=\Pi_{R_1}(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_5}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$$

- 排序 ORDER[A,K,$ASC|DESC$]是一个过程. ORDER[A,K,$ASC|DESC$](R)将 R 中元组按属性 A 的升序(ASC)或降序(DESC)排序，并只取其前 K 个元组. 它将一个关系变成一个有序表. 虽然这破坏了查询结果的封闭性，但是，由于用户对内容查询的结果具有一定的倾向性，因此，要求结果按某种顺序输出是必要的.

对于 MOQL 中定义的谓词，可以由 DB2 SQL 语句定义如下：

- BEFORE$_t$，AFTER$_t$，IN$_t$，OVERLAP$_t$定义为$\underline{BtP}$，$\underline{AtP}$，$\underline{ItP}$，$\underline{OtP}$；
- CONSTAINS 定义为$\underline{CoP}$；
- BEFORE，MEETS，OVELAPS，DURING，STARTS，FINISHES 和 EQUALS 定义为$\underline{BsP}$，$\underline{MsP}$，$\underline{OsP}$，$\underline{DsP}$，$\underline{SsP}$，$\underline{FsP}$ 和$\underline{EsP}$.

4.5 优化处理

代数优化基于下列变换规则：

(1) PJ[A](OP(R_1,R_2))↔OP(PJ[A](R_1),PJ[A](R_2)),OP∈{INT,UN,DF}；

(2) UN(SL[P_1](R),SL[P_2](R)↔SL[P_1 or P_2](R)；

(3) INT(SL[P_1](R),SL[P_2](R))↔SL[P_1 and P_2](R)；

(4) SL[P](OP(R_1,R_2))↔OP(SL[P](R_1),SL[P](R_2))；

(5) PJ[S](SL[P](R))↔SL[P](PJ[S](R))，S 含有 P 中所有属性；

(6) PJ[S](NJ[A_1,A_2](R_1,R_2))↔NJ[A_1,A_2](PJ[S](R_1),PJ[S](R_2))，A_1 和 A_2 同属于 S；

(7) EXTEND[U,f](PJ[S](R))↔PJ[S,U](EXTEND[U,f](R))，f 函数使用的属性在 S 中；

(8) EXTEND[U,f](SL[P](R))↔SL[P](EXTEND[U,f](R))；

(9) EXTEND[U,f](OP(R_1,R_2))↔OP(EXTEND[U,f](R_1),EXTEND[U,f](R_2))；

(10) EXTEND[U,f](LOOKUP[A_1,A_2,...,A_{2k},P](R_1,R_i,...,R_{k+1}))↔LOOKUP[A_1,A_2,...,A_{2k},P](EXTEND[U,f](R_1),R_2,...,R_{k-1})；

(11) PJ[S](LOOKUP[A_1,A_2,...,A_{2k},P](R_1,...,R_{k+1}))↔LOOKUP[A_1,A_2,...,A_{2k},P](PJ[S](R_1),R_2,...,R_{k+1}))，S 包含 A_1；

(12) SL[Q](LOOKUP[A_1,A_2,...,A_{2k},P](R_1,R_2,...,R_{k+1}))↔LOOKUP[A_1,A_2,...,A_{2k},P](SL[Q](R_1),R_2,...,R_{k+1})).

定理 1. 根据各个算子的定义，上列规则是正确的.

证明：见附录.

4.6 生成存取方案

存取方案生成过程将代数表达式中的代数算子翻译成含有动态 SQL 语句和 C$_{++}^{++}$过程的 DB2 数据库访问过程.

各算子对应的 SQL 语句过程如下：

- PJ[A](R)：SELECT A FROM R；
- SL[P](R)：SELECT * FROM R WHERE P；
- NJ[A_1,A_2](R_1,R_2)：SELECT * FROM R_1 r_1, R_2 r_2 WHERE $r_1.A_1=r_2.A_2$；
- UN(R_1,R_2)：R_1 UNION R_2；
- INT(R_1,R_2)：R_1 INTESECT R_2；
- DF(R_1,R_2)：R_1 EXCEPT R_2；
- EXTEND[A_1,...,A_n;f_1,...,f_n](R)：SELECT *,$f_1(B_1)$ AS A_1,...,$f_n(B_n)$ AS A_n FROM R；(B_1,...,B_n 为 R 的属性)
- LOOKUP[A_1,A_2,...,A_{2k},P](R_1,...,R_{k+1})：

```
SELECT *
FROM R1
WHERE A1 IN (SELECT A2
                FROM R2
                WHERE A3 IN (...(SELECT A2k FROM Rk+1 WHERE P)...))
```

ORDER[A_1,K,$ASC|DESC$](R)：是一个含有动态 SQL 语句的 C 语言过程.

4.7 执行存取方案

将上述生成的含有动态SQL语句的查询方案在DB2系统中运行,得到满足条件的对象OID.

5 小结与讨论

本文主要讨论多媒体数据库查询的需求、MOQL语言及其查询处理.MOQL语言能够支持类型、结构、同步关系、时间关系及内容查询的、简单的多媒体数据库查询语言.本文讨论了以DB2数据库作为存储机制的MOQL语言查询处理技术.查询处理过程中,以用户定义的MOQL查询作为源语言,以DB2数据库上运行的含有动态SQL语句的DB2查询过程作为目标语言,定义了一组代数算子作为查询优化的内部表现机制,给出了一组变换规则以便进行代数优化.

未来的工作包括:(1) 提高MOQL语言的表达能力.(2) 定义更完备的代数算子集和优化规则集.(3) 基于代价模型的优化技术.(4) 研究各种非结构化媒体的内容查询方法.

参考文献

1 Woelk D, Kim W. An extensible framework for multimedia information management. Data Engineering, 1987,(3):115~121

2 Goble C, Docherty M O *et al*. The Manchester multimedia information system. In: Goos G, Hartmanis J eds. Proceedings of the International Conference Extending Database Technology. Berlin: Springer-Verlag, 1992. 39~55

3 Little T D. A frame work for synchronization delivery of time-dependent multimedia data. Multimedia Systems, 1993, 1(1):87~94

4 Karmouch A, Emery J. A playback schedule model for multimedia documents. IEEE Multimedia, 1996,(3):50~61

5 Megalni C, Rabitli F *et al*. Conceptual modeling of multimedia documents. IEEE Computer, 1991,24(10):23~29

6 Flicker M, Sawhney H *et al*. Query by image and video content: the QBIC system. IEEE Computer, 1995,28(9):22~31

7 Gupta A, Weymouth T E, Jain R. Semantic queries with pictures: the VIMSYS model. In: Lohman M, Sernadas A, Camps R eds. Proceedings of the 17th International Conference Very Large Databases. Barcelona: Morgan Kaufmann Publishers, Inc., 1991. 69~79

8 Wu J K *et al*. CORE: a content-based retrieval engine for multimedia information systems. Multimedia Systems, 1995, 3(1):43~52

9 Oomoto E, Tanaka K. OVID: design and implementation of a video-object database system. IEEE Transactions on Knowledge and Data Engineering, 1993,5(4):629~643

10 Allen J F. Maintaining knowledge about temporal intervals. Communications of ACM, 1983,26(11):832~843

附录:

定理1的证明:

(1) 由关系代数理论知变换规则(1)~(6)是正确的.下面主要证明规则(7)~(12)的正确性.

(2) 规则(7)左部: $\mathrm{EXTEND}[U,f](\mathrm{PJ}[S](R))$

$=\mathrm{EXTEND}[U,f](\{s|s=\Pi_S(r)\wedge r\in R\})=\{(s,u)|s=\Pi_S(r)\wedge r\in R\wedge u=f(r)\}$,

规则(7)右部: $\mathrm{PJ}[S,U](\mathrm{EXTEND}[U,f](R))$

$=\Pi_{[S,U]}(\{(r,u)|r\in R\wedge u=f(r)\})=\{(s,u)|s=\Pi_s(r)\wedge r\in R(u=f(r)\}$,

故规则(7)成立.

(3) 规则(8)左部: $\mathrm{EXTEND}[U,f](\mathrm{SL}[P](R))$

$=\mathrm{EXTEND}[U,f](\{r|P(r)\wedge r\in R\})=\{(r,u)|P(r)\wedge r\in R\wedge u=f(r)\}$,

规则(8)右部: $\mathrm{SL}[P](\mathrm{EXTEND}[U,f](R))$

$=\sigma_p\{(r,u)|r\in R\wedge u=f(r)\}=\{(r,u)|P(r)\wedge r\in R\wedge u=f(r)\}$,

故规则(8)成立.

(4) 这里仅证明OP为UN时的情况,其他情况由下列证明易得.

规则(9)左部: $\mathrm{EXTEND}[U,f](\mathrm{UN}(R_1,R_2))$

$=\mathrm{EXTEND}[U,f](\{r|r\in R_1\vee r\in R_2\})$

$=\{(r,u)\mid(r\in R_1\wedge u=f(r))\vee(r\in R_2\wedge u=f(r))\}$,

规则(9)右部：$\mathrm{UN}(\mathrm{EXTEND}[U,f](R_1),\mathrm{EXTEND}[U,f](R_2))$

$=\mathrm{UN}(\{(r,u)\mid r\in R_1\wedge u=f(r)\},\{(r,u)\mid r\in R_2\wedge u=f(r)\})$

$=\{(r,u)\mid(r\in R_1\wedge u=f(r))\vee(r\in R_2\wedge u=f(r))\}$,

故规则(9)成立.

(5) 规则(10)左部：$\mathrm{EXTEND}[U,f](\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](R_1,R_2,\ldots,R_{k+1}))$

$=\mathrm{EXTEND}[U,f](\Pi_{R_1}(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1})))$

$=\Pi_{[R_1,U]}(\mathrm{EXTEND}[U,f](R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1})))$

$=\Pi_{[R_1,U]}(\{(t,u)\mid t\in(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))\wedge r=\Pi_{R_1}(t)(u=f(r))\})$,

规则(10)右部：$\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](\mathrm{EXTEND}[U,f](R_1),R_2,\ldots,R_{k+1})$

$=\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](\{(r,u)\mid(r\in R_1\wedge u=f(r)\},R_2,\ldots,R_{k+1})$

$=\Pi_{[R_1,U]}(\{(r,u)\mid(r\in R_1\wedge u=f(r))\}\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$

$=\Pi_{[R_1,U]}(\{(t,u)\mid t\in(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))\wedge r=\Pi_{R_1}(t)\wedge u=f(r)\})$,

故规则(10)成立.

(6) 规则(11)左部：$\mathrm{PJ}[S](\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](R_1,\ldots,R_{k+1}))$

$=\mathrm{PJ}[S](\Pi_{R_1}(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1})))$

$=\Pi_S(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$,

规则(11)右部：$\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](\mathrm{PJ}[S](R_1),R_2,\ldots,R_{k+1}))$

$=\Pi_S(\mathrm{PJ}[S](R_1)\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$

$=\Pi_S(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$,

故规则(11)成立.

(7) 规则(12)左部：$SL[Q](\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](R_1,R_2,\ldots,R_{k+1}))$

$=SL[Q](\Pi_{R_1}(R_1\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1})))$

$=\Pi_{R_1}(SL[Q](R_1)\bowtie_{A_1=A_2}R_2\bowtie_{A_3=A_4}R_3\bowtie_{A_5=A_6}\cdots\bowtie_{A_{2k-1}=A_{2k}}\sigma_p(R_{k+1}))$

$=\mathrm{LOOKUP}[A_1,A_2,\ldots,A_{2k},P](SL[Q](R_1),R_2,\ldots,R_{k+1})$

=规则(12)右部.

由(1)～(7)知，定理 1 成立. □

Multimedia Object Query Language and Its Query Processing

TIAN Zeng-ping　DANG Hua-rui　ZHOU Ao-ying　SHI Bai-le

(*Department of Computer Science　Fudan University　Shanghai*　200433)

Abstract　In this paper, the authors study the query requirements of multimedia database and propose a structured multimedia object query language (MOQL). The language can support queries based on type, structure, synchronous relations, temporal relations, and contents of multimedia data. Taking the DB2 relational database system as a storage mechanism, they define a set of algebraic operators and some transformation rules, by which the MOQL queries defined by the users can be transformed into a set of algebraic formula and optimized. Finally, the algebraic formula can be transformed into execution plan in DB2SQL and C++ procedures, which can be run on DB2 system.

Key words　Multimedia database, query language, query optimization.