データベースにおける意味的情報の構造的表現

片山 紀生　　高須 淳宏　　安達 淳

東京大学大学院工学系研究科　　学術情報センター研究開発部

全文データベースでは、データが多様な構造および多様な意味的関係を持っている点が問題となる。我々はこの問題を、データベースにおける意味的情報の管理の問題と考え、意味的情報を表現するためのデータモデルとして「意象モデル (Semantic Object Model)」を提案する。意象モデルは、意象、意関、意約の三種類の構成要素から成る。意象は実体やデータなど個々の意味的な対象を、意関は意象間の意味的な関係を、意約はデータベースの意味的な制約を、それぞれ表現する。意象モデルによって構築される意味表現は、意味的な一意性に基づいているという点で直接的であり、意象、意関、意約によって組織的に表現されるという点で構造的である。

Structural Representation of Semantic Information in Databases

Norio KATAYAMA　　Atsuhiro TAKASU　　Jun ADACHI

Graduate School of Engineering
University of Tokyo

Research and Development Department
National Center for Science Information Systems

The major difficulties in implementing full-text database systems originates in the diversity of data structures and semantic relationships. We regard the difficulties as an issue of semantic information management, and propose a novel data model named *SEMANTIC OBJECT MODEL*. It realizes structural representation of semantic information in databases. The semantic object model comprises three components: semantic objects, semantic relationships, and semantic constraints. Semantic objects represent individual semantic entities. Semantic relationships represent semantics among semantic objects. Semantic constraints manage semantic integrity of a database. The characteristics of the model is derived from that it is designed based on the concept of semantic uniqueness.

1　はじめに

現在、データベースシステムを様々な分野に適用することを目的として研究がさかんに行われている。特に、近年では、オブジェクト指向データベース技術によって、CAD, CAM, CASE 分野への応用が実現され、現在も研究が進められている。そのような新しい応用分野のひとつとして、文献の全文情報 (文章、ページイメージ、論理構造など) を提供する全文データベースシステムが注目されている。

全文データベースシステムにおいては以下の二つの具体的な問題がある。

- 多様な文書構造の管理

  文書には、章や節といった論理構造と、ページや図の配置などの物理構造とがある。しかも、これらの構造は文書によって異なっており、すべての文書に適合するような一様な構造は定義できない。

- 多様な意味的関係の管理

  文書の論理構造では、構成要素間に様々な関係が存在する。また、文書相互の間にも引用関係が存在する。さらに、著者や収録雑誌などに関する情報も含めて考えれば、全文データベースが管理すべき意味的関係は多種多様なものとなる。

これらの問題の解決は、関係データベースやオブジェクト指向データベース [1] といった従来のデータベースでは難しいと考えられる。というのは、これらのモデルでは、スキーマの設計時にリレーションやクラスという形で、文書、章、ページといった実体の構造を予め決定することが必要だからである。全文データベースの場合、個々の文書が持っている構造は一様でないため、スキーマ設計時に実体の構造を決定することは困難である。

またハイパーテキストは、文書構造の多様性については対応できるものの、多様な意味的関係の管理には適さない。データベース技術を融合しようとするハイパーテキストデータベース [2] と呼ばれるアプローチもあるが、意味的関係に対して一般的な枠組を与える研究は見られない。

我々は、上記の問題を次世代データベースにおける意味的情報の管理の問題と考え、意味的情報を直接的に表現するためのデータモデルが必要であると考えた。本論文では、この要求を満足するデータモデルとして 意象モデル (Semantic Object Model) を提案する。意象モデルは、意味データモデルをより一般化したものであり、情報の意味的な一意性を規範として意味表現を構築する。意象モデルは、データベースの意味的情報を直接的にしかも構造的に表現することができるため、スキーマ、一貫性制御、並行処理制御といったデータベースにおける意味的な問題について検討する際、構造的な基盤を提供することができる。

2　意象モデルの概要

まず、意象モデルの概要を説明する。

例えば、図 1 で示されるデータベースを考える。この例では、三つのリレーションが定義されている。「著者」および「文書」は、実体についてのリレーションであり、「文書–著者」は「著者」と「文書」の間の関連についてのリレーションである。

このデータベースを意象モデルによって表現すると図 2のようになる。円で表されたものは、文書などの実体や、文字列、ページイメージなどのデータを表現するオブジェクトであり、意象 (意味対象: Semantic Object) と呼ぶ。それぞれの意象は、データベースにおいて意味的に一意な情報を表現する。

意象の間にネットワーク状に張られているリンクは、意象間の意味的な関係を示すものであり、意関 (意味関係: Semantic Relationship) と呼ぶ。それぞれの意関に付けられている名前は、リンクが表している意味を一意に識別するためのものである。意関の名前を述語とみることによって、図 3 のように図 2 を述語の羅列によって表現することもできる。この例では、2 項関係しか含まれていないが、一般的には n 項関係を定義できる。

さらに、意象モデルの構成要素として、意約 (意味制約: Semantic Constraint) がある。意約は、意象や意関の間に存在する意味的な制約を記述したものである。意味的な制約とは、空値に関する制約、定義域に関する制約、クラス継承に関する制約、意味的関係相互の依存関係などのことを指す。意象モデルでは、意関が述語によって表現される

著者

著者 ID	名前	所属
1	N.Katayama	U-Tokyo
2	A.Takasu	NACSIS
3	J.Adachi	NACSIS

文書

文書 ID	タイトル	刊行年
1	Database	1992
2	Hypertext	1993
3	Semantics	1993

著者–文書

著者 ID	文書 ID
1	1
2	2
3	2
3	3

図 1: サンプルデータベース

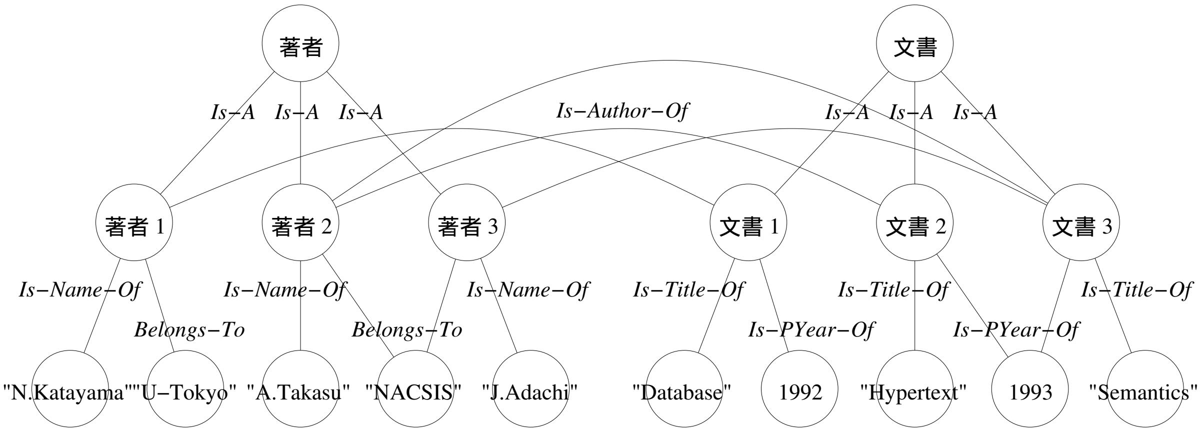


図 2: 図 1のデータベースの意象モデルによる表現

```
Is-A [ Class = (Entity)"文書", Instance = (Entity)"文書-1" ]
Is-A [ Class = (Entity)"文書", Instance = (Entity)"文書-2" ]
Is-A [ Class = (Entity)"文書", Instance = (Entity)"文書-3" ]

Is-Title-Of [ Title = (String)"Database", Of = (Entity)"文書-1" ]
Is-Title-Of [ Title = (String)"Hypertext", Of = (Entity)"文書-2" ]
Is-Title-Of [ Title = (String)"Semantics", Of = (Entity)"文書-3" ]

Is-A [ Class = (Entity)"著者", Instance = (Entity)"著者-1" ]
Is-A [ Class = (Entity)"著者", Instance = (Entity)"著者-2" ]
Is-A [ Class = (Entity)"著者", Instance = (Entity)"著者-3" ]

Is-Name-Of [ Name = (String)"N.Katayama", Of = (Entity)"著者-1" ]
Is-Name-Of [ Name = (String)"A.Takasu", Of = (Entity)"著者-2" ]
Is-Name-Of [ Name = (String)"J.Adachi", Of = (Entity)"著者-3" ]

Is-Author-Of [ Author = (Entity)"著者-1", Of = (Entity)"文書-1" ]
Is-Author-Of [ Author = (Entity)"著者-2", Of = (Entity)"文書-2" ]
Is-Author-Of [ Author = (Entity)"著者-3", Of = (Entity)"文書-2" ]
Is-Author-Of [ Author = (Entity)"著者-3", Of = (Entity)"文書-3" ]
```

図 3: 図 2の述語による表現

ため、意約を述語論理によって記述することができる。例えば、意関 *Is-Title-Of* の Of 項の定義域が、図 2の「文書」であることを記述すると下のようになる。

```
Is-A [ Class = "文書", Instance = Y ]
   ←  Is-Title-Of [ Title = X, Of = Y ].
```

意象モデルの特徴を挙げると次のようになる。

- 実体やデータはもちろん、集合やクラスなどの抽象概念までも一様に、内部構造を持たない意象として表現する。
- 個々の意象は、意味的に一意な情報を表現する。
- 意象相互の意味的な関係を意関として定義し、n 項述語によって表現する。
- データベースがネットワーク構造になり、航行型問合せが可能である。
- 意味的な関係が n 項述語で表現され、述語論理による宣言的問合せが可能である。
- 意象および意関に関する意味的な制約を意約として管理する。

意象モデルの最も大きな特徴は、意象の定義が、意味的な一意性に基づいている点である。このことから、意象モデルでは、実体、データ、抽象概念といった意味的な対象は、すべて一様に意象としてモデル化する。これは関係データモデルやオブジェクト指向データモデルなど従来のデータモデルにはない性質である。

意象モデルよって構築される意味表現は、意味的な一意性に基づいているという点で直接的であり、意象、意関、意約という三つの構成要素によって組織的に表現されるという点で構造的である。

3 意象モデルの概念

3.1 意象

意象は、データベースにおける意味的情報の最小単位であり、個々の意象は、データベースにおける一意な意味的対象を表現する。意象として表現されるのは以下のものである。

実体

個々の文書、個々の人間、個々のページといった実世界における「もの」を意象として表現する。

抽象概念

クラスや集合といった抽象概念も意象として表現する。文書クラスや人間クラスも、一意な意味を持っているので、それぞれ意象として表現する。

物理データ

数値データ、文字列、テキスト、ラスタイメージなど物理データも意象として表現する。というのは、それぞれのデータも一意な意味を持っているからである。たとえば、数値型の `1` は「数字の 1 」という意味を持っている。また、文字列型の `"database"` は「文字が “database” と並んだもの」という意味を持っている。これらの例からわかる通り、データが持つ意味はその値によって決定される。意象は一意な意味ごとに生成されるので、物理データに対する意象は、データの値ごとに生成されることになる。つまり、意象とデータ値との関係は 1 対 1 になるのである。この性質は意象の概念の大きな特徴である。

3.2 意関

意象間には意味的な関係が存在する。たとえば、「文書-1」の著者が「著者-1」であるとき、「文書-1」を表す意象と「著者-1」を表す意象の間には、*Is-Author-Of* という意味的な関係が存在する。このような意味的な関係を表現するのが意関である。意関は下のように n 項述語によって記述される。

```
Is-Author-Of [ Author = , Of = ]
```

ここで `Is-Author-Of` は述語名であり、括弧で挟まれた部分はラベルの付いた項の並びである。`Author` および `Of` が項のラベルである。

意関も意象と同様に意味的な一意性に基づいて定義される。すなわち、ある意関が表現する意味は一意であり、逆にある意味的関係を表現する意関は唯一でなければならない。例えば、*Is-Author-Of* という意関は、項の値にいかなるものが代入されようと、著者と文献との間の意味的関係しか表現しない。

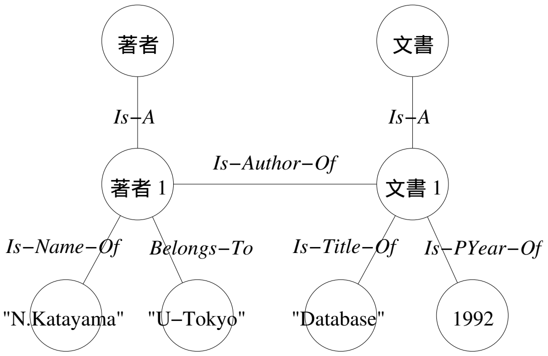


図 4: 著者と文書の意味的な関係

実際に、意関の項に値を代入することによって、意象間に存在する意味的なリンクを表現することができる。例えば、二つの意象、「著者-1」と「文書-1」の間に *Is-Author-Of* の関係があるときには、下のように記述する。

```
Is-Author-Of [ Author = (Entity)"著者-1",
               Of     = (Entity)"文書-1" ]
```

ここに、`(Entity)"著者-1"` および `(Entity)"文書-1"`は、意象を特定するための便宜的な記法である。

意関を用いて意象相互の意味的な関係を記述していくことにより、データベースが格納する情報の意味表現を構築することができる。

3.3 意約

意象と意関によって、データベースが持つ意味的情報を表現することが可能であるが、更新において意味的な一貫性を維持するためには、さらに意味的な制約を指定する必要がある。例えば、意関 *Is-Author-Of* の Author ラベルの値は、この意関が表す意味から当然、図 4のように、「著者」という抽象概念を表す意象と *IS-A* 関係を持っていなければならない。

このような意味的な制約は、意象や意関が表現する意味情報にとって本質的なものであり、意象モデルでは、これらの制約を意約と呼び、データモデルの構成要素のひとつとして位置付ける。

意象モデルでは、意関が n 項述語によって表現されるため、意約は述語論理によって記述することができる。例えば、上記の *Is-Author-Of* に関する制約は下のように記述される。

```
Is-A [ Class = "著者", Instance = X ]
   ←  Is-Author-Of [ Author = X, Of = Y ].
```

3.4 意象モデルの特徴

実体と物理データとを区別しない

意象モデルでは、意味表現としては実体と物理データとを区別せず、いずれも意象として取り扱う。これは、一意な意味を表現するという意味で、実体も物理データも違いがないからである。例えば図 5 は、文書とそのキーワードの関係を意象モデルによって表現したものである。意象モデルでは、データの値ごとに意象を定義するため、文書とキーワードの関係は図のように多対多となる。したがって、実体と物理データとは、意関について対等であり、実体から物理データを検索する場合も、物理データから実体を検索する場合も一様に、意象の検索として処理することができる。

実体と抽象概念とを区別しない

意象モデルでは、個々の実体と、集合やクラスなど抽象的な概念とを区別しない。例えば図 6 は、個々の文書と、文書クラスおよび論文クラスとの関係を示したものである。オブジェクト指向データベースでは、これらの *Is-A*、*Is-Kind-Of* といった関係を管理する機能は、データモデルの中に取り込まれている。

意象モデルでは、これらの関係をモデルに取り込むことはせず、他の関係と同様に扱う。ただし、それぞれの関係にともなう意味的な制約、例えば上位クラスと下位クラスに関する制約などについては、意約としてデータベース設計者が与えるものとする。例えば、下位クラスに属するインスタンスは上位クラスにも属するというオブジェクト指向モデルでは当然の制約を、意象モデルでは図 7 のように明示的に記述することになる。

実体構造の明示的な定義が必要ない

意象モデルでは、実体は意象によって表されるが、意象は構造を持っていない。したがって、実体の構造をスキーマにおいて明示的に定義する必要がない。また、実体がデータや他の実体とどの

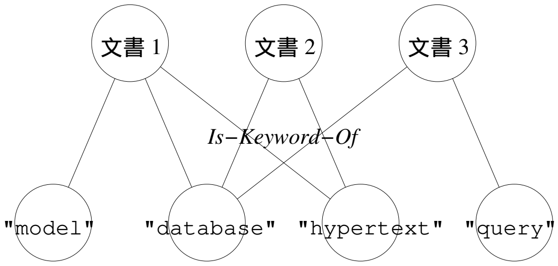


図 5: 意象モデルによる文書のキーワードの表現

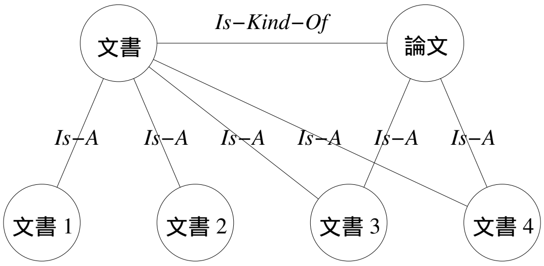


図 6: 意象モデルによる IS-A 関係の表現

```
Is-Kind-Of [ Superclass = X, Subclass = Z ]
          ←  Is-Kind-Of [ Superclass = X, Subclass = Y ],
             Is-Kind-Of [ Superclass = Y, Subclass = Z ].

Is-A [ Class = S, Instance = X ]
          ←  Is-Kind-Of [ Superclass = S, Subclass = C ],
             Is-A [ Class = C, Instance = X ].
```

図 7: IS-A 関係に関する一貫性制約の記述

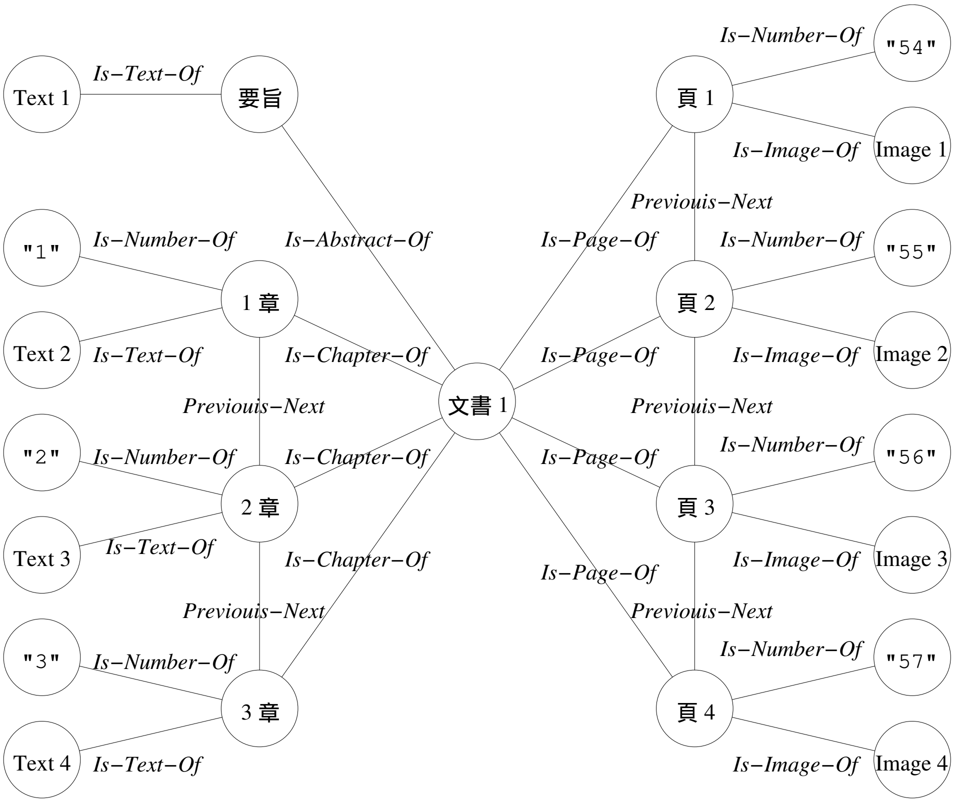


図 8: 意象モデルによる文書構造の表現

ような意味的関係を持つかについても、スキーマにおいて確定する必要がない。

この性質は、全文データベースにおける文書構造のように、多様な構造を持った実体を格納する場合に有効である。例えば、図8は、ひとつの文書の論理構造と物理構造とを表現したものである。これらの構造は、文書ごとに異なるが、意象モデルでは、意象と意関とを組合せることによって、柔軟に表現することができる。

意関を n 項述語で表現する。

意象間の意味的関係を n 項述語で表現することによって、述語論理による問合せ処理、更新処理、一貫性制約の記述が可能になっている。すなわち、宣言的な問合せを実現することができる。これは、意味ネットワークやハイパーテキストといった意味的な関係を簡単なリンクによって表すモデルには見られない特徴である。

ネットワーク型の意味表現を持つ

意象モデルによる意味表現は、ネットワーク型の構造を持つ。したがって、意象をノード、意関をリンクとみなすことによって、ハイパーテキストのような航行による問合せを実現できる。

意象と意関について意味的な制約を記述する

意味的な制約をデータモデルの中に取り込んでいることも、意象モデルの特徴のひとつである。意象および意関についての意味的な制約を、意約として管理する。

意味表現の構造的な要素である意象および意関から、意味的な制約についての機構をすべて意約に集めている点が重要である。このことによって、意象および意関は一意な意味を表現するだけの単純な機構となり、意象、意関、意約という三つの構成要素の機能分担が、直交性の高いものになる。

3.5 意象モデルの問題点

意象モデルの問題点は、その意味表現が複雑なネットワーク構造になることである。これは、意象が構造を持たないこと、および、意味的な一意性に基づいて直接的に表現していることに由来しており、意味表現という性質上やむをえないことである。データベースを効率的に実装するという目的では、関係データモデルやオブジェクト指向データモデルなど、実体の構造をスキーマにおいて確定するモデルの方が適していると考えられる。しかし、次世代データベースにおける意味的情報に関する問題を検討するためには、意象モデルのような直接的な意味表現を実現するデータモデルが必要であると考える。

3.6 関連研究との比較

意象モデルは、意味的な一意性に基づいている点で、関係データモデルを始めとする従来のデータモデルと異なっている。さらに、関数型データモデル [3, 4] に代表される意味データモデルに対しては、n 項述語によって意味的関係を記述している点、意約の概念を持っている点が異なっており、オブジェクト指向データモデル [1] に対しては、実体が構造を持たない点、クラスのような抽象的な概念をモデルの中に含めない点が異なっている。演繹データベース [5] とは、述語論理による表現を用いる点は共通しているが、意象というデータや実体を抽象化する機構を持っている点が異なっている。ハイパーテキストデータベース [2] については、ネットワーク型の構造を持っている点が共通しているが、意象の概念を持っている点、n 項述語による表現を用いている点が異なっている。

4 意象モデルの実装に関する検討

4.1 意味的情報と物理データの分離

意象モデルでは、数値、文字列、テキストなど全ての物理データは、内部構造を持たない意象として取り扱われ、しかも意象とデータ値との関係は1対1となる。したがって意味的情報という観点からは、すべての物理データは値の固定した定数のように取り扱われる。この性質から、図9のように、物理データを意味的情報から分離して管理する方法が考えられる。

意味的情報を格納する部分を意味リポジトリと呼び、物理データを格納する部分をデータリポジトリと呼ぶ。

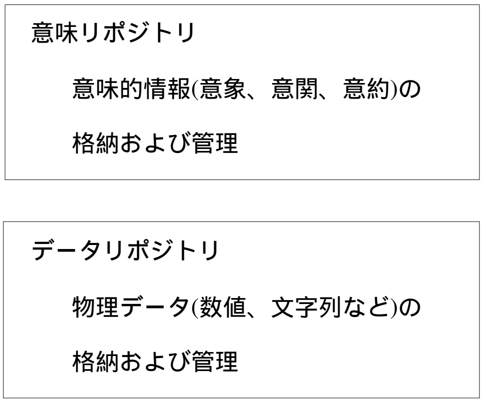


図 9: 意味的情報と物理データの分離

4.2　意味リポジトリ

意味リポジトリは、意象、意関、意約に関する情報を格納する。問合せを評価するために、意味リポジトリは、意象表と意関表の二種類の表を生成する (図 10)。

意象表は、三つの属性 (意象 ID、リンク ID、意関 ID) を持つ。この表は、航行型問合せを実現するために必要であり、特定の意象と結合しているすべての意象を検索するのに使われる。

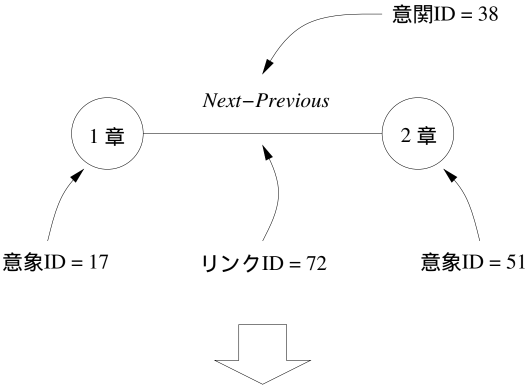
意関表は、個々の述語ごとに作られる。この表は、宣言的問合せにおいて述語を評価するのに使われる。

4.3　データリポジトリ

データリポジトリは、数値、文字列、SGML 文書、ページイメージなど物理的なデータを格納する。オブジェクト指向プログラミングのように、同じデータ型を持つデータはクラスとしてまとめられる。個々のクラスは、以下の四つの基本的機能を持たなければならない。

物理データの登録

この機能は、物理データをデータリポジトリに格納し、その物理データに対応する意象を生成する。意象モデルでは、意象は内部構造を持たない。従って、構造を持つデータは、意象と意関とによって組み立てられねばならない。例えば、SGML 文書のような構造を持ったデータを格納するときには、まず文書はパーズされ、その構成要素が抽出される。そして、個々の構成要素に対して、意象が生成され、構成要素の間に意関を定義することによって、データの構造を表現する。



意象表

意象ID	リンクID	意関ID
17	72	38
51	72	38

Next−Previous

リンクID	Next	Previous
72	51	17

図 10: 意味リポジトリの表

物理データの取得

この機能は、特定の意象に対応する物理データを取得するためのものである。意象モデルでは、意味リポジトリにおける問合せの結果は意象として返されるので、問合せ結果を利用者が見るためには、データリポジトリから対応する物理データを取り出さなければならない。

物理データの検索による意象の特定

意味リポジトリに対して述語による問合せを行うとき、物理データを定数として利用するためには、その物理データに対応する意象を特定しておく必要がある。物理データと意象との間のマッピングは、データリポジトリによって管理されるので、データリポジトリにおいて物理データを検索すれば、そのデータを表す意象を特定することが

```
Result [ Document = D, Title = T ]
              ←  Is-A [ Class = (Entity)"文書", Instance = D ],
                 Is-Title-Of [ Title = T, Of = D ],
                 Is-Substring-Of [ Substring = (String)"database", Of = T ].
```

図 11: 述語を定義する方法による問合せの例

できる。物理データを検索する手段としては、文字列に関しては照合による検索、テキストデータに関しては全文検索、画像情報に関してはブラウジングによる検索などが考えられる。

物理データ固有の意関の管理

物理データには、数値間の大小関係や、文字列間の部分一致関係など、データ相互の間に本質的な関係が存在する。意象モデルでは、これらの関係も意味リポジトリにおける意関の一種として取り扱い、問合せにおいて評価の対象となる。

例えば、文字列の部分一致関係は、意味的な関係の一種として、下のような意関として表現される。

`Is-Substring-Of [ Substring = , Of = ]`

この意関は、意味リポジトリが直接管理する意関と同様に、問合せ記述で用いることができるが、実際の評価はデータリポジトリによって行われる。

4.4 問合せ処理

意象モデルでは、航行型問合せと宣言的問合せが可能であるが、宣言的問合せは、述語論理によって記述される。特に、問合せ用の新しい述語を定義する形で、目的とする意象を選び出すところに特徴がある。宣言的問合せは次の三つの処理によって組み立てられる。

- 意象の特定

  物理データや実体など、問合せにおいて定数として利用する意象を特定する。

- 述語の定義

  データベースにおいて定義されている述語を組み合わせて、目的とする意象のみが取り出されるような述語を定義する。

- 物理データの取得

  述語の定義によって意象が選択されたら、その意象に対応する物理データを取り出す。

述語による問合せの例を図 11に示す。これは、タイトルに “database” の文字列が含まれるすべての文書を見つけ出すものである。

`(Entity)"文書"` という表現は、`"文書"` という名前の実体を表す意象を特定するための記述である。`(String)"database"` という表現は、`"database"` という文字列に対応する意象を特定するための記述である。

Is-Substring-Of という意関は、文字列データ間の本質的な関係であるという意味で他の意関とは異なっている。したがって、*Is-Substring-Of* 意関はデータリポジトリによって評価され、他の意関は意味リポジトリによって評価される。

4.5 更新処理

意象モデルにおける更新処理は次のように分類される。

意象の生成・削除

意象の生成は、新規データを登録したとき、新たな実体あるいは新たな抽象概念を定義するときに行われる。従来のデータベースで言えば、タプルの追加やオブジェクトの追加に相当する。

リンクの生成・削除

意象モデルでは、属性値の変更がリンクの変更として表現される。これは意象が一意な意味を表現するという性質による。例えば、「著者-1」の所属を、`"U-Tokyo"` から `"NACSIS"` に変更する処理は図 12 のように表現される。

意関の定義・削除

データベースで管理する意味的な関係を追加したり削除する場合に、意関の生成・削除が行われる。これは、従来のデータベースで言えば、スキーマの変更に相当する。

意約の定義・削除

意約を定義する場合、その時点でのデータベースの内容が新しい意約を満たしているかどうか

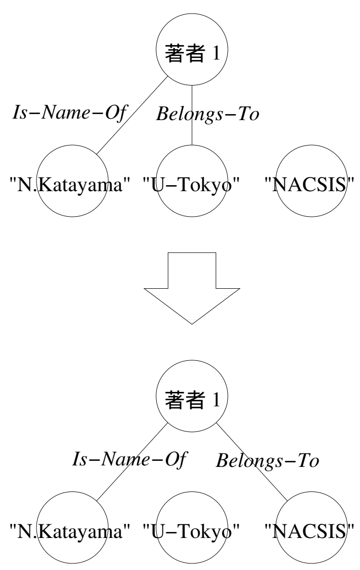


図 12: リンクの変更による更新処理

検証する必要がある。もし満たしていない場合には、その意約の定義は棄却される。

5 今後の課題

データベース言語の設計

意象、意関、意約の定義、手続き型の処理の記述、アプリケーションプログラムとのインタフェースなど、データベース言語にとって必須となる部分について検討を行う必要がある。意象モデルの直接的かつ構造的な意味表現としての性質を活かすことによって独自のデータベース言語の設計が可能であると考えている。

否定および空値に関する検討

意象モデルでは、意象間にリンクが定義されていない場合、以下の三つの状態が考えられる。

否定 : 該当する意味的関係が存在しないので、リンクが定義されていない状態。

不明 : リンクを定義すべき意象が不明であるため、リンクが定義されていない状態。

不当 : 実体の構造など意味的な理由でリンクが存在し得ない状態。

述語論理による問合せ処理において、これらの相違がどのような影響を及ぼすのか検討を行う必要がある。

並行処理制御方式に関する検討

意象モデルでは、実体の構造あるいは属性値の更新は、意象間のリンクの変更となって表現され、トランザクション間の衝突は意関に集中する。この性質を利用して意象モデルに適した並行処理制御方式を検討できるものと考えている。

最適化手法に関する検討

意象モデルは概念的なモデルであり、実装に対する配慮は行っていない。そのため、プロトタイプの構築など実装を目指す場合には、記憶構造や問合せ処理について、最適化手法を検討する必要がある。

6 あとがき

多様な構造、多様な意味的関係を持ったデータに対するデータモデルとして、意象モデルの提案を行った。意象モデルは、意味的な一意性を規範としており、データベースにおける意味的情報の直接的かつ構造的な表現を実現する。

今後はプロトタイプの開発を行い、意象モデルの妥当性について検討を進めていく考えである。

参考文献

[1] Kim, W., “Introduction to Object-Oriented Databases,” MIT Press, 1990.

[2] 銭晴, 谷崎正明, 原伸一郎, 田中克己, “ハイパーテキストデータベースシステム TextLink / Gem におけるオブジェクトとスキーマの動的、段階的な構築機能”, 信学技法, DE92-39.

[3] 有澤博, “意味データモデルの最近の動向,” 情報処理, Vol. 32, No. 9, pp. 1023-1031, Sep. 1991.

[4] 石川英彦, 石山政浩, 有澤博, “データベース集合演算の高速化,” 信学技法, DE92-25.

[5] 横田一正, 西尾章治郎, “演繹・オブジェクト指向データベース,” 情報処理, Vol. 31, No. 2, pp. 234-243, Feb. 1990.