

4M-1

フレーム型知識表現による
関係データベースの記述に対する検討

伊藤 秀昭

(財)日本情報処理開発協会 開発研究室

1. はじめに

データベースシステム(DBS)を知識表現システムと結合し、これを用い知識型システムを実現する試みが盛んに行われている。例えば、あるデータベース言語を通じデータベース(DB)をアクセスする方法、データベース管理システム(DBMS)を拡張することにより知識表現システムとして実現する方法、などが種々提案されている[KERS]。一般に、DBSは、あるデータモデルを提供し、これを用いデータの検索および更新を可能とするシステムである。ここで、データモデルは、データ構造、このデータ構造を操作するための演算およびデータに対する一貫性制約、などより成る。これらは開発されるシステムの目的および実現方法にも依存することになるが、DBMSまたは知識型システムにおいて記述、定義することが必要である。

現在、我々はフレーム型知識表現システムFKBUSの開発を進めている[ITO]。このシステムは、関係型データモデルに基づくDBMSを統合化し、これを用いて知識型システムを開発するためのツールとして機能することを目的の一つとするシステムである。これを達成するためのアプローチとして、FKBUSによりデータモデルの一部を記述および定義し、システム内部においてデータの管理メカニズムの一部を実現および提供するというアプローチを採用している。本稿においては、データ構造の記述、SQL文とフレームの関係、などについて述べる。なお、本システムは、SUN3ワークステーションにより稼働し、Sun Common LispおよびC言語により記述されている。また、DBMSとしてInformixを用いている。

2. データ構造の記述のためのフレームについて

関係スキーマの集合であるデータベーススキーマおよび関係スキーマを構成する属性の記述について述べる。この種のメタ情報はDBMSが備える固有のDD/D[LEON]に定義されている。しかしながら、ここでは、問題分野の構造を記述することが困難、DB管理のための手続きと知識型システムにおける問題解決のための手続きの混乱、などが問題点となっている[ISO]。一方、本システムにおいては、固有にDBの構造を記述することによる知識型システムとの親和性、メタ情報を操作することが可能であるという柔軟性、などの理由によりメタ情報の一部はFKBUSの備える知識ベースに定義することとした。

FKBUSの提供する知識ベースは、フレーム型データ構造であり、その表現単位はフレームと呼ばれ、個々のフレームは複数のスロットより成る。また、スロットはいくつかのファセットより成る。このデータ構造を利用し、対象とするDBの提供するデータ構造を定義し利用するためのフレームには大きく、次の二種のフレームがある。

(1) データ構造(表)を記述するためのフレーム。これを

データ構造記述フレーム(DDF)と呼ぶ。この種のフレームには、①ビューを記述するためのフレーム、②DBに定義されている関係である基底関係を記述するためのフレーム、③属性であるフィールドを記述するためのフレーム、の三種がある。

(2) 問題分野を記述するフレームにおいてDBに定義された関係を参照するフレーム。この種のフレームを問題記述フレーム(PDF)と呼ぶ。

まず、PDF(図1)の構造について述べる。これらのフレームにおいていくつかのスロットは、DBを管理するためのシステムスロットである。これら以外に問題分野に応じたスロットおよび各々のフレームの目的に応じたスロットが定義される。ここでは、いくつかの主たるシステムスロットについて述べる。Data-relation-nameスロットには、参照するDDFを示すポインタが記入される。スロットrelation-typeはそのPDFがいかなる種類の関係を利用するのかということを示している。これにはある関係を基に他の関係の合成であることを示すcomposite、定義されているビューであることを示すviewおよび実表であることを示すgroundの三種がある。ここで、relation-typeの値としてcompositeが指定されたならばoriginal-tableスロットが定義され、そこには合成される関係を定義するために基となるPDF名またはDDF名が記入される。スロットfield-namesには、その関係スキーマを構成するフィールド(属性)の名前から成るリストが記入される。このような属性はスロットとして定義される。この種のスロットは、そのフィールドのデータ型、デフォルトおよびドメインを記入するためのファセットなどを有する。さらに、これらのスロットは、問題分野に応じて設定されるスロットとなる。

次に、DDFの構造(図2)について述べる。この種のフレームは、全てフレーム型instanceを有するフレームである。スロットrelation-nameはDBに定義される表の名前が記入される。スロットstruct-relation-nameには、相当するPDFへのポインタが記入される。スロットrelation-typeは、そのDDFが実体集合であることを示すentity、関連集合であることを示すrelationship、またはビューであるviewを値とする。

PDFおよびDDFは、上記のシステムスロットの他に、いくつかのスロットがそのフレームの記述する対象の目的に応じて定義される。

3. SQL文の利用および一貫性制約について

3.1 データ構造記述フレームおよびSQL文

DDFにおいて基底関係を定義するフレームは、(1)エディタによる定義、(2)SQL文による定義(ここでは、create table文)、の二種の方法により定義される。(1)のエディ

```

frame-name : struct-r
frame-type : class
data-relation-name : data-r
relation-type : { composite
                  view
                  ground
original-table : compositeの場合指定される
field-names : 属性名の列
[属性の定義スロット]
field-1 : { [data-type] : スロット値の型
            [value] : 指定されたならば下位フレームに継承される
            [domain] : 定義域
            [default] : 暗黙値
            [reference] : 部分の如何なる属性を参照するのを示すポインタ
            }
parts : 他関係を記述するフレームへのポインタ
join-keys : 結合演算のための条件を記述するスロットへのポインタ
key-1 : 具体的な結合処理のための条件の記述
        } compositeのとき

```

ただし、parts、join-keys はrelation-type がcomposite である場合に設定される。また、自動的に下位に受け継がれるスロットは属性を定義するスロットのみである。

図1 問題記述フレームのデータ構造

```

frame-name : data-r
frame-type : instance
data-relation-name : rel-r ; 実際の関係名
struct-relation-name : struct-r ; もし存在するならば相当するフレームへのポインタ
table-type : { entity
               relationship
               view
field-names : 属性名の列
[属性の定義スロット]
field-1 : { [value] フィールド記述フレームへのポインタ
            [unique]
            }
primary-key : 主キー
index :
index-type : } インデックスの定義
index-field : } 指定されたならば定義する
connected-entity : relationship であるとき参照するフレームへのポインタ
connections : 上記の参照関係を記述するスロットへのポインタ
con-k : 参照関係を記述するスロット
foreign-keys : 外部キー
sql-statement : view であるとき生成するための文を記述するフレームへのポインタ

```

図2 データ構造記述フレームのデータ構造

ィタでは、基底関係を定義するためのコマンドにより、相当するデータ構造が定義される。このとき、DDFにおけるシステムスロットの値の設定の順序および値に対する管理は、このエディタにより実現されている。エディタにおける種々のパラメタの設定に関する動作は、SQLの構文に従う制御により管理されている。この機能が提供されていることにより、利用者はよりSQLと密接にDBの定義を行うことが可能となる。

(2)の方法によるものは、ある与えられたSQL文を解析し、そこから得られるテーブル名、テーブルを構成する属性などを先に述べたDDFとして定義する。これを実現するために、SQL文を解析するためのSQLサブフレームシステムがFKBUS知識ベースに設けられている(図3参照)。この種のフレームにおいて定義されているスロットは、個々のSQL文の構文に基づき設定されている。例えば、SELECT文では、取り出されるフィールドを記述するスロット、取り出される関係を記述するスロット、などが定義されている。このサブフレームシステムを構成し、それから得られる情報をDDFのスロットに設定する。これは、文法が変更された場合にも対処することが可能であるという柔軟性をシステムに提供するためである。

ただし、現時点ではアクセス権の制御およびDBの更新

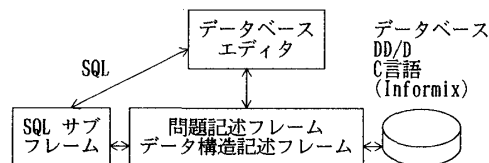


図3 構成要素の概念的構造

に関するものなどは未だ実現していない。

3.2 データの更新について

本システムにおけるDBに格納されたデータの検索および更新は基本的にはタプルに基づく操作のみが提供されている。これは、基底関係だけではなくビューに対しても同様である。このタプルは現在子を示すポインタにより指定されている。タプルの更新に係わるチェックおよび実際の更新はフレームに付加される手続きにより記述される(フレームに付加されたattached procedures: AP)。このAPを定義するために次のような6種のスロットがある。

- ① tuple-appended; ある関係にタプルが付加されたときに起動される手続き。
- ② tuple-delete; ある関係から現在子により示されるタプルが削除されたときに起動される手続き。
- ③ tuple-modify; ある関係から現在子により示されるタプルが更新されたときに起動される手続き。
- ④ check-append, check-delete, check-modify; これらは上記①～③に先立ち起動される手続きであり、この手続きを活性化し評価の結果が偽以外ならばそれぞれに相当するAPが起動される。

これらの手続きは、メッセージ交信による明示的な起動、またはシステムが提供する関数により相当する手続きが起動された際に一種のフレームに付加されたデモンとして起動される。ただし、現時点において個々のAPの記述は利用者の負担となっている。これを容易に記述するための言語および関数群が必要であると考えている。

4. おわりに

本稿においては、関係モデルにおけるデータ構造を記述するための枠組みを中心にして、SQL文とシステムの関連および現時点において実現したデータの更新について概略を述べた。これらの詳細は、機会を別に述べたいと考えている。今後、一貫性制約についてより検討を進めると共に、知識ベースシステムにおけるトランザクションメカニズムについて検討を進めたいと考えている。

[参考文献]

- [ISO]ISO/IEC DIS 10027: "Information Processing Systems - Information Resource Dictionary System (IRDS) Framework", 1989
- [ITO]伊藤, 他: "フレーム型知識表現システムFKBUSにおけるデータベースの統合利用について", 第38回情報全大
- [JANS]Jansen, B., et al: "The Knowledge Dictionary: A Relational Tool for the Maintenance of Expert Systems", Proc. on FGCS, 1988
- [JIS]JIS X 3005: データベース言語SQL, 1987
- [KERS]Kerschberg, L. ed.: "Expert Database Systems, Proc from 2nd Int. Conf.", Benjamin/Commings, 1989
- [LEON]Leon-Hong, B., et al: "Data Dictionary/Directory Systems", John Wiley & Sons, 1982
- [WALD]Wald, J., et al: "Implementing Constraints in a Knowledge Base", In [KERS]