解説

3. 複合オブジェクトに基づく演繹データベース†

三浦孝夫†† 塩谷 勇†††

1. 前書き

演繹データベースの目的は，データベースをより高度なものにすることである[1),2)]．データベースが事実のみを蓄えるのに対して，演繹データベースでは規則や法則などのより表現能力の高い記述をも許す．処理対象となるデータは第一階論理の一つのモデル(外延)に対応し，これらから演繹されるものを質問の答とする．第一階論理は演繹推論システムと記述言語の両方の役割を果たしており，このことが論理プログラミングとの関連を与えている（論理プログラミングは，プログラム（節集合）の計算過程を第一義的なものと考える点が異なる）．第一階論理は，両者の共通の概念的枠組みを与えているのである．

データベース研究の多くは関係モデルを基にしている．表形式構造をとり，形式化と自然な対応がついたためである[6)]．現在，データベースの研究ではより複雑な構造を扱う試みがなされている．この原因として非正規 (NF²) 関係の提案[7)]と，データモデル分野でのさまざまな試み，特に意味データモデルやオブジェクト指向データベースの研究[8),9)]があげられる．現在，データベースが積極的な情報の構造化を試みていることは，演繹データベースへも強い影響を与えている．本稿では現状を紹介するが，筆者の力量不足と急速な進展から，触れることのできないものがあることをお断わりしたい．

2.では，複合オブジェクト研究の出現背景を示し，**3.**では非正規関係データベースの定義と操作言語について述べる．**4.**では，データベースの述語論理による形式化，特にデータベース論理について論じる．**5.**は，ホーン論理を拡張して直接データ構造を取り扱う論理系を考察する．**6.**は要約である．

2. 複合オブジェクトの出現背景

論理プログラムでは，データの表現は基礎原子式（事実，外延情報）が単位となる．この表現方法は，述語記号の後に一定の数の項が並び，各位置には固定した役割が与えられる．例 2.1 は，従業員情報を表したものである．

【例 2.1】

Emp (Name, Ext, Role, Dept, Sals)

Emp は従業員を記述する述語，Name は氏名，Ext は内線番号，Role は役職，Dept は所属，Sals は給与歴で年度・金額の組とする．**表-1** はこの事実集合を表す．■

関係データベースでは，対象とする情報は述語記号（関係名）を用いてモデル化される．しかし，演繹データベースでは，述語論理に基づくため，関数を用いてもよい．たとえば**表-2** では，部署関数が部署名と所在地を，また給与歴関数が年度と金額の二つの項目を引数とする．この形式を用いることで，意味まとまりの単位を構造的に捉え，意図の表現をより直観的なものとすることができる．ここでは関数に特別の働きを与えず，その形式だけに注目している．

構造化には一般的な「関数形式」以外にも，汎用のデータ構造記法を新たに項とする「データ構造形式」が考えられる．データ構造形式は関数形式の一種であるが，データ構造のもつ意味に対応した特定の解釈を

表-1 従業員の事実集合

Emp (山田, 100, 部長, システム, 1985-10)
Emp (山田, 100, 部長, システム, 1986-14)
Emp (山田, 150, 部長, プラント, 1985-10)
Emp (山田, 150, 部長, プラント, 1986-14)
Emp (鈴木, 110, 課長, システム, 1986-7)
Emp (鈴木, 110, 課長, システム, 1987-9)
Emp (渡辺, 170, ヒラ, プラント, 1988-5)

† Deductive Databases Based on Complex Objects by Takao MIURA (SANNO College, Department of Management and Informatics) and Isamu SHIOYA (Tokyo Denki University, Faculty of Sciences and Engineerings).

†† 産能大学経営情報学部
††† 東京電機大学理工学部

表-2 関数形式による従業員表現

Emp (山田, 100, 部長, 部署 (システム, 3F), 給与歴 (1985, 10))
Emp (山田, 100, 部長, 部署 (システム, 3F), 給与歴 (1986, 14))
Emp (山田, 150, 部長, 部署 (プラント, 4F), 給与歴 (1985, 10))
Emp (山田, 150, 部長, 部署 (プラント, 4F), 給与歴 (1986, 14))
Emp (鈴木, 110, 課長, 部署 (システム, 3F), 給与歴 (1986, 7))
Emp (鈴木, 110, 課長, 部署 (システム, 3F), 給与歴 (1987, 9))
Emp (渡辺, 170, ヒラ, 部署 (プラント, 4F), 給与歴 (1988, 5))

表-3 順序組構造による従業員表現

Emp (山田, 100, 部長, [システム, 3F], {(1985, 10), (1986, 14)})
Emp (山田, 150, 部長, [プラント, 4F], {(1985, 10), (1986, 14)})
Emp (鈴木, 110, 課長, [システム, 3F], {(1986, 7), (1987, 9)})
Emp (渡辺, 170, ヒラ, [プラント, 4F], {(1988, 5)})

表-4 関数形式による汎化従業員表現

Emp (山田, 100, 部長 (山本, 大場秘書), システム, 1985-10)
Emp (山田, 100, 部長 (山本, 大場秘書), システム, 1986-14)
Emp (山田, 150, 部長 (加藤, 川合秘書), プラント, 1985-10)
Emp (山田, 150, 部長 (加藤, 川合秘書), プラント, 1986-14)
Emp (鈴木, 110, 課長 (業務課), システム, 1986-7)
Emp (鈴木, 110, 課長 (業務課), システム, 1987-9)
Emp (渡辺, 170, ヒラ (鈴木), プラント, 1988-5)

もつ．たとえば,「順序組」関数の場合, 各情報表現の位置情報があらかじめ定められた意図を表すという同質性を用いて, 簡潔な表現を得る. **表-3** では,「部署」や「給与歴」を位置で表し, 関数記号をもたない. 構造項表現で構造を質問対象とするには, これを値とする変数を導入すればよい. 述語によるモデル化では高階論理が必要となる.

述語論理では, 関数形式の柔軟さを用いて, 各情報は同質性を保ちながら, より詳細・固有の情報を表すことができる. たとえば, **表-4** では, 部長ならば副部長・秘書, 課長ならば課名, ヒラならば上司名というように, 役職ごとに個別の情報を示している. これは汎化または部分型と呼ばれる[10]. 個々の情報の関数記号は種別を表すタグとして働いていることに注意したい. 反面, この柔軟性は, データ構造形式にはなく, タグ情報を別にもたねばならない.

表-5 従業員関係

Name	Ext	Role	Dept	Sals
山田	100	部長	システム	1985-10
山田	100	部長	システム	1986-14
山田	150	部長	プラント	1985-10
山田	150	部長	プラント	1986-14
鈴木	110	課長	システム	1986-7
鈴木	110	課長	システム	1987-9
渡辺	170	ヒラ	プラント	1988-5

非正規関係データベースには, これまで二つのアプローチ・出現理由がある[7]. 本来, 関係は単純値の順序組からなる. 例 2.1 に対応して, **表-5** で関係スキーマおよび実現値を表す. ここでは, 給与歴属性値は従業員氏名に対して繰り返し出現する. この冗長性 (給与歴回だけの繰り返し) を解決するため, 関係モデルでは高次正規化, すなわち関係の属性単位での分解を行うが, そのため構造値で結び付く情報が分解によって希薄になるという欠陥があった. 非正規関係は, 属性間の構造によって冗長性を防ぐ目的を有する[7),11)]. **表-6** は表-2 から, 述語名, 関数形式, データ構造形式などを取り除き, その形式を属性記述に反映させたものに対応する. 順序組化と集合化を繰り返して適用することにより, 属性名を節点とする木として型が構成される (どの属性名も高々1回しか生じない). 属性値の内部構造はこの構成に従わねばならない. このように, 関係をより効率よく表現するためにデータ構造を導入した方式を表現指向アプローチという. この特長は構造のない元の関係と同じ意味を有することである.

これに対して, 情報構造がもつ意味をそのままデータ構造に反映させ, それ自体を独立した「もの」として扱う方法が, 値指向アプローチである. **表-7** では, 表-6 に集合値属性「好物」を追加した. たとえば {豆

表-6 非正規関係による従業員表現 (1)

Name	Ext	Role	Dept	Sals
			[部名, 場所]	{[年度, 金額]}
山田	100	部長	システム, 3F	1985, 10 1986, 14
山田	150	部長	プラント, 4F	1985, 10 1986, 14
鈴木	110	課長	システム, 3F	1986, 7 1987, 9
渡辺	170	ヒラ	プラント, 4F	1988, 5

表-7 非正規関係による従業員表現 (2)

Name	Ext	Role	Dept	Sals	好物
			[部名, 場所]	{[年度, 金額]}	
山田	100	部長	システム, 3F	1985, 10 1986, 14	{豆腐, ミソ}
山田	150	部長	プラント, 4F	1985, 10 1986, 14	{豆腐, ミソ}
鈴木	110	課長	システム, 3F	1986, 7 1987, 9	林檎 林檎
渡辺	170	ヒラ	プラント, 4F	1988, 5	メロン

腐，ミソ｝は，組み合わせたものが好物であって，豆腐やミソは個別には違うことを表す．つまり，各集合値は構成要素とは別の意味をもつ．

非正規関係データベースとデータ構造形式を許す演繹データベースを統合するため，データ構造による情報の取扱・表現形式（複合オブジェクトという）の研究が発展してきた．本質的には関数形式を拡張したものだが，表以外にも式や木によって表す．また，ここで対象とする構造化は，順序組化・集合化のほかにも，和を許すものもある[12]．

関係データベースでは，出現当初から代数系操作と（一階述語）論理系操作が提案され，二つの表現能力の比較検討がなされた．C. Zaniolo は上述の関数形式を直接データに埋め込み，関数形式を扱うように拡張した選択・射影・和・結合を定義した[13]．この拡張関係代数は，関数形式を許す非再帰ホーン節論理と等価になる．つまり，このホーン節論理を基準にした完備性が定義できる．次の例 2.2 は，表-4 に対して，関数形式の選択とその引数を問う論理プログラムである．

【例 2.2】 構造情報の問い合せ

部長職 (Name, Sec)←

Emp(Name, _, 部長 (_, Sec), _) ■

この提案以降，非正規関係データベース上の関係代数とホーン論理の拡張との関連が研究されている．また定理証明の観点から，操作言語の表現能力を問う基準として，証明手続きとしての完全性が論じられている．しかしデータ構造を論理に導入した一般的な枠組みでは証明完全性が成立しない[14]．このため，完全な部分クラスが提案されている．

3. 非正規関係

非正規 (NF^2) 関係は，有限長のテーブル形式の表現で，その各順序組（タプル）の属性値は単純値または非正規関係をとる．ただし，順序組の属性値は同質（すなわち，同じ型）であり，各値の構造化の方法は同じである．通常，対象となる構造の構成は順序組化と集合化の二つに基づく．順序組化と集合化が交互に繰り返されるものを非正規関係，そうでないものを複合オブジェクトと呼ぶものもあるが[11]，ここでは非正規関係に統一する．表-6 や表-7 は非正規関係の例である．関係に構造を導入することは，データベース設計時に認識・検出されねばならない．表現指向・値指向の区別は構造化の意図を正しく記述する基準となっている[7]．非正規関係の単純値属性上で順序組の値が一意に定まるならば，この非正規関係を階層関係または分割正規形と言う[15),16]．

関係データベースと同じように，非正規関係には二つの操作言語が提案されている．非正規関係論理[17]に基づくデータベース質問は，$\{t|\phi(t)\}$ の形式で表される．ここで，$\phi(t)$ は順序組変数 t を自由変数とする式で，$R(t)$, $s\varepsilon t$, $d\theta s$, $s\theta t$, $s=\{u|\phi(u)\}$ の基本形の結合（$\exists\forall\vee\wedge\neg\rightarrow$）である．ここで，$d$ は定数，t, s は変数，θ は比較子で＝または算術比較であり，R は属性名，ϕ は $u, t, ..$ を自由変数にもつ式で s を含まないとする．非正規関係 R に対するデータベース質問 $\{t|\phi(t)\}$ の解とは，ϕ を満たすすべての順序組 t の集合をいう．非正規関係が同じ任意の二つのデータベースに対して，データベース質問Qが同じ答を生成するならば，Qは領域独立であると言う．関係データベースと同様に，式の構文形式から領域独立性を判定することとは決定不能である．そこで，どの変数の値もそれが生じる部分式内で規定されるという構文則を与え，これを満たすとき安全と言う．同様に，変数値を得るためのべき集合計算を許さない安全な式を，強意に安全と言う．

次に，非正規関係上の関係代数[16)~18]を考える．これまでの関係代数との違いは，ネスト，アンネスト，べき集合の三つの操作である．ここで，属性集合 S に対しネスト操作 $\nu_{T=S}(R)$ とは，式 R に生じる S以外の属性で値の等しい順序組を一つにまとめ，S 上集合値を取るように表す操作である．このとき新しい属性を T: $\{S\}$ とする．アンネスト $\mu_{W=S}(R)$ は，式 R が S: $\{T\}$ 上集合値を取るとき，これを要素ごとに繰り返して展開する操作を言う．属性名 S の代わりに型 W が表れる．

安全な論理操作，領域独立な論理操作および拡張関係代数が等価であること，べき集合操作を除いた拡張関係代数が強意に安全な論理操作と等価であることが知られている[16),17]．

次の例では，二つの非正規関係に対して，従業員とその給与値集合の組を求める質問を，安全な論理と代数で表現したものである．

【例 3.1】 非正規データ操作

Emp(Name, Ext, Role, Dept : [DName, Loc],
Sals : {[Year, Amnt]})

例 2.1 の詳細化

PaidTax(Name, Sals : {[Year, Amnt, Tax]})

従業員と税金・給与歴，Emp に含まれないものも

ある

（a） 論理操作

$\{u \mid \exists x(\mathrm{Emp}(x) \wedge u[1]=x[1] \wedge$
$u[2]=\{y \mid \exists z \exists w(\mathrm{Emp}(z) \wedge u[1]=z[1] \wedge$
$w \in y[5] \wedge y=w[2]\})))\}$
$\vee \exists x(\mathrm{PaidTax}(x) \wedge u[1]=x[1] \wedge$
$u[2]=\{y \mid \exists z \exists w(\mathrm{PaidTax}(z) \wedge$
$u[1]=z[1] \wedge w \in y[2] \wedge y=w[2]\}))$
$\}$

（b） 代数操作

$\nu_{\mathrm{Amnt}}(\pi_{\mathrm{Name,S.Amnt}} \mu_{\mathrm{S=Sals}}(\mathrm{Emp})$
$\cup \pi_{\mathrm{Name,S.Amnt}} \mu_{\mathrm{S=Sals}}(\mathrm{PaidTax}))$ ■

関数のないホーン節論理プログラム（たとえば Datalog）を質問言語とする演繹データベースでは，推移閉包の計算などの再帰質問[20]や，前提部に否定を含む場合の形式的意味についての研究がなされている．後者では，否定記号に関して述語にある半順部が与えられるときに層状論理プログラムと呼び，小さいものから順に計算することで（順序の設定に依存しない）一意な極小モデルが得られることが知られている[45]．

非正規関係上の安全な論理操作は，推移閉包を表現できる．例 3.2（b）の第1式は領域を定めており，第2式は（a）の規則1に相当し，第3式は規則2に相当している．第4式は，最小性を表現している．ここで，この式は強意に安全ではない．すなわち，代数的に表現すればべき集合演算を用いていることが分かる．一般に，非正規関係上の安全な論理操作は，層状論理プログラムと等価であることが知られている[17]．このほか，データ構造の操作について情報容量の観点から完全性を論じた研究もある[12),21]．

【例 3.2】 推移閉包

$P(xy)$：x は y の親，$V(xy)$：x は y の祖先

（a） 論理プログラムによる表現

$V(xy) \leftarrow P(xy)$
$V(xy) \leftarrow P(xz), V(zy)$

（b） 非正規関係論理操作による表現

領域 (t)：$\forall x(x \in y[1] \rightarrow \exists z(P(z) \wedge x[1]=z[1])$
$\wedge \exists w(P(w) \wedge x[2]=w[2]))$

直系 (t)：$\forall x(P(x) \rightarrow x \in t[1])$

閉包 (t)：$\forall x \forall y(x \in t[1] \wedge y \in t[1] \wedge x[2]$
$=y[1] \rightarrow \exists z(z \in t[1] \wedge z[1]=x[1] \wedge$
$z[2]=y[2]))$

最小性：$V=\{x \mid \forall t$(領域 $(t) \wedge$ 直系 $(t) \wedge$
閉包 $(t) \rightarrow x \in t[1]\}$ ■

4. データベース論理

4.1 データベース論理

述語論理にデータ構造を導入し，データベースをその解釈と考えるアプローチが B. E. Jacobs によって提案された[14]．この形式化により，非正規関係だけでなく，階層データモデル・ネットワーク（網）データモデルなどを統一的に捉えることができる．この形式化は，スキーマ・操作言語・制約条件の三つを定義することでなされる．スキーマとは，$S=(R_1 \cdot\cdot R_n)$, $n>0$ の形の定義式の有限集合である．ただし，S, R_i は相異なる名前で，どの名前も高々1回しか左辺に生じず，各名前にはネストレベルと呼ばれる自然数が対応するとする．式集合中の右辺にしか生じない名前を0次名，そうでないものを高次名という．左辺にしか生じない名前を外部名，そうでないものを内部名という．スキーマを図示するため，グラフ技法を導入する．このグラフでは，名前を頂点に，式 $S=(R_1 \cdot\cdot R_n)$ に対して S から各 R_i への有向辺を対応させる．例 4.1 は従業員情報の表現を示す．

【例 4.1】

a） 関係スキーマ

Emp＝(Name, Ext, Role, Sals)

従業員の氏名 (Name), 内線 (Ext), 役職 (Role), 給与歴 (Sals) を表す．

b） 階層スキーマ

Emp＝(Name, Ext, Role, Sals)
Sals＝(Year, Amnt)

従業員の氏名 (Name), 内線 (Ext), 役職 (Role), 給与歴（高次名），給与歴は年度 (Year), 金額 (Amnt) からなる

c） 網スキーマ

Emp＝(Name, Ext, Dept)
Dept＝(DName, Emp)

従業員の氏名 (Name), 内線 (Ext), 所属部署（高次名），部署の名前 (DName), 代表者 (Emp)
■

例 4.1 に対応するスキーマグラフを図-1 に示す．簡単に分かるように，関係スキーマはレベルが1の木グラフ集合，階層スキーマは木グラフの集合，網スキーマはサイクル（長さゼロのループ）のないグラフで表される．

データベース論理の式 $S=(R)$ は非正規関係では S：$\{[R]\}$ に相当し，データベース論理に生じる名前

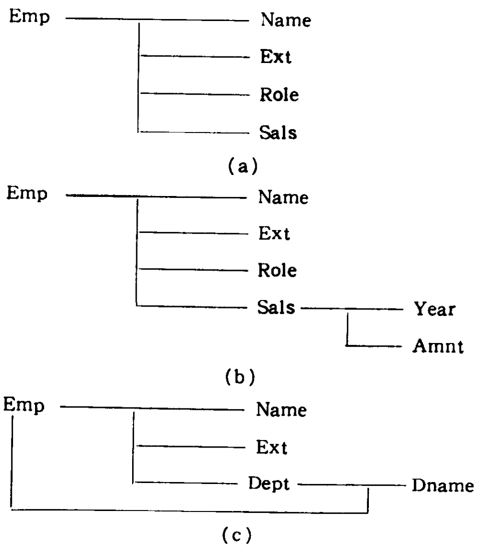

図-1 例 4.1 のスキーマグラフ

は，非正規関係ではある条件をもつ構造型に対応する（このため型と呼ぶこともある）．例 4.2 は表-6 の非正規関係を表す．

【例 4.2】 データベース論理のスキーマ
Emp＝(Name, Ext, Dept, Sals)
Dept＝(DName, Loc)
Sals＝(Year, Amnt) ■

データベース論理での質問式は，特殊な型構造（スキーマ）を有する第一階多類論理式である．変数・定数・関数や述語記号 p は型付されており，スキーマでの式 $S=(T_1..T_n)$ があるとき，s, t_i をそれぞれ定義型 S, T_i の項とすると，式は（1）$p(t_1..t_n)$,（2）$t_i=t_i'$,（3）$t(t_1..t_n)$,（4）$t\in t'$, ただし項 t の型は，t' の型のスキーマグラフ上での真の子孫のいずれかの結合（∨∧¬∃∀→ による）である．その解釈では，0次名Aに対して原子値集合 D, 高次名 S に対して S の式の右辺 $(R_1..R_n)$ の各 R_i の解釈の直積 $R_1\times..\times R_n$ を対応させる．式の定義では，（3),（4）の形が本質である．（3)の式 $t(t_1..t_n)$ が真とは，各項の解釈値 t', t_i' が $t'=(t_1'..t_n')$ となるときとする．また（4)の形の式 $t\in t'$ が真とは，$T'=(..T..)$ がスキーマで与えられ，t, t' がそれぞれ型 T,T' の項であれば，t' は集合値であり，t はその要素のいずれかの成分として生じる．T が T'の右辺に生じる（子と言う）のでなければ，T' の子孫 S で T の親となるものに対して型Sの項 s があり，$t\in s$, $s\in t'$ を満たすものがある．ただし，どの型もネストレベルで与えられた自然数が対応し，この展開はその数以内で終了するという制限を付ける．

第一階述語論理と同様，充足性，モデル，データベース質問，論理帰結，公理系や（Modus Ponens などの）推論規則が定義される．例 4.3 は従業員氏名と所属部署を求める質問である．

【例 4.3】 データベース論理の質問
従業員氏名とその所属部署
W(Nm, DN)：∃Emp1 ∃Ext1 ∃Dept1 ∃Sals1 ∃Loc2
(Emp1(Nm, Ext1, Dept1, Sals1)∧Dept1(DN, Loc2)) ■

データベース論理に（第一階論理と同様）公理系や導出規則を与えれば，多類論理の拡張として第一階述語論理に変換できるようにみえる．しかし，証明可能ならば論理帰結である（健全性）が，逆（完全性）が成立しない．これを例 4.4 に示す．制約集合は従業員 (n) の所属部署 (n) とリーダ $(n+1)$ の連鎖を定数で表す．明らかにどの有限部分集合もそのモデルを作ることができるから充足可能である．しかし，制約集合自体は無限の連鎖を表せ，どのようなネストレベルでもこれを超え，充足可能ではない．この結果コンパクト性（閉式集合の充足可能性と任意の有限部分集合の充足可能性とが等価であること）が成立せず，完全性が得られない．つまりこの性質を有する第一階論理への埋め込みができない．

【例 4.4】 証明可能でないデータベース
Emp＝(Dept, Emp)
従業員（Emp）はリーダ（Emp）のいる部署（Dept）に属するとする
制約条件：$\{Emp_n(Dept_n, Emp_{n+1}) | n=0, ..\}$ ■

4.2 データベース論理に基づくアプローチ

データベース論理の問題は構造情報の表現力に関するものと完全性の不成立にある．構造化については，順序組化と集合化の交互の出現だけを扱い，また記法上の区別ができないことが問題である．完全性の不成立は，多類論理での型解釈に起因する．ここには二つの問題が含まれ，データがサイクルを有するとき（例 4.3で，2人の従業員が互いに相手のメンバになるケース）と，無限個の要素を含む領域の取扱い（例 4.4 のリスト）である．前者は証明手続きで問題を起こす可能性があり，後者は有限で扱えない．

そこで，型（類）の解釈を限定し，証明手続きを完全にする試みがある．F論理[22),23)]はオブジェクト指

向データベースの形式化の過程で提案されたモデルであるが，データベース論理の観点からも考察できる．
F論理では，F項とは $T_1 : T_2$ の形か $T_1 : T_2[L_1\to V_1..L_n\to V_n, S_1\to U_1..S_k\to U_k]$ の形を言う．ここで T_1, T_2 は（第一階）項，$L_1..L_n$, $S_1..S_k$ は相異なる項でラベルと呼ばれ関数対応を意味し，$V_1..V_n$ は項，$U_1..U_k$ は項集合である．T_1 は省略できる．F論理式とは，F項の結合（⇒∨∧¬∃∀による）を言う．F項の第1式は型 T_1 の項 T_2 を，第2式はさらにそれが満たすべき条件を意味する．特に第2項のラベル項の働きは，その項のもつ意味と独立に与えられた対応関係を取り出すものである．ラベルは各順序組構成要素に対応し，その値が単純値または集合値となっている．F論理は，レコードの項目記述レベルで情報を扱い，多レベルにわたる情報は識別項（オブジェクト ID）で操作される．例4.5は，従業員情報を記述したものである．

【例 4.5】

Emp : yamada[Name→山田, Role→部長,
info→{yama_sys[Ext→100, Dept→システム]
yama_pla[Ext→150, Dept→プラント]},
Sals→{yama_sal 1[Year→1985, Amnt→10]
yama_sal 2[Year→1989, Amnt→15]}
] ■

データベース論理との最大の違いは型の扱いである．データベース論理は，多類論理であるため，型を対象とする質問はありえずどの項も特定の型をもつ．一方，F論理では，型構造の解釈に順序化を与えるためサイクルは生じず，順序概念を利用することで型と項を区別しないでよい．このことは，継承概念のモデル化に利用される．

多類論理としての解釈のうち，型解釈を限定し「問題のない」実現値（関係）だけを対象とする（一般構造という[47)]）．このアイデアに基づいたオブジェクトモデルの形式化[48)]と同じ視点から，データ論理が提案されている[24)]．ここで想定している多類論理は，定数は複数の型を有してよく，型の制約条件を記述でき，複合オブジェクトの構造化記法（順序組化 [..] や集合化 {..}）を有する．（複合）型は，基本型，順序組型 $[T_1..T_n]$，集合型 $\{T\}$ と型のブール結合（∨∧による）を言う．このとき同時に，型構造が与えられ，$T=W$ の形の有限集合，T は基本型，W は複合型で記述されている．集合化は有限集合の集まりとする．原子式は，$p(t1..tn)$, $t\in s$, $t\equiv s$, $t=s$ のどれかである．$t\in s$ はメンバシップを，$t\equiv s$ は型構造等値（型 T の定数 t の関数値が s），$t=s$ は等値を表す．型構造 $T=W$ は定数と構造値の明示的な結び付きを意味し，非正規関係で述べた「値指向」のモデル化に対応している．変数には複合型が（暗黙または明示的に）指定されるため，データベースに格納されていない構造値が動的に得られる．例4.6は，例4.5と同じ内容を表している．

【例 4.6】

型構造：Emp=[Name, Role, {Dept}, {Sal}]
Dept=[Ext, D_Name]
Sal=[Year, Amnt]

従業員記述

yamada $\equiv_{\mathrm{Emp}}$[山田, 部長, {yama_sys, yama_pla}, {yama_sal1, yama_sal2}]
∧yama_pla $\equiv_{\mathrm{Dept}}$[150, プラント]
∧yama_sys $\equiv_{\mathrm{Dept}}$[100, システム]
∧yama_sal1 $\equiv_{\mathrm{Sal}}$[1985, 10]
∧yama_sal2 $\equiv_{\mathrm{Sal}}$[1986, 15] ■

5. 拡張ホーン節論理

5.1 複合オブジェクトの表現：順序組型

確定ホーン節データベース（またはプログラム）は確定ホーン節の有限集合，等号公理，単一名仮定，領域閉包公理からなる．このデータベースは，宣言的意味，演算的意味および関数的意味がすべて一致するという性質を有する[5),35)]．確定ホーン節データベースの拡張は否定などで試みられている．ホーン節を拡張して複合オブジェクトを表現する試みは大きく二つの流れが存在する．順序組（レコード）型を中心とするものと，集合型を中心とするものである．前者は順序組型の各属性が再び順序組型となることをモデル化したもので，論理プログラム上の述語表現と関数表現の類似性に着目している．後者は，集合型を陽に捉えるものと，集合構成規則（グループ化規則）を用いて内包的に定義できるとする立場がある．

束を用いた情報の階層表現が考えられている[26)]．これまでの第一階述語論理では，述語式では引数の数や並び順が厳格に定義され，関数形式を用いて構造表現を行うことは可能であっても，かなりの煩わしさをともなう．このため，緩い記法を目的とした構文規則から構築されねばならない．そこで，項表現の方法を拡張し，従来の変数・定数・関数に，順序組・集合記法を加える：W_i は項で，A_i は名前（属性名）とする

と，$[A1:W1,..An:Wn]$，$\{W1,..,Wn\}$.

各項を式またはオブジェクト，変数のないオブジェクトを基礎オブジェクトと言う．オブジェクトに部分順序を導入し束と見なすことができる．ただしオブジェクト O_1 が O_2 の部分オブジェクト $(O_1 \leqq O_2)$ とは，ともに集合式のときは O_1 のどの要素も O_2 の要素の部分オブジェクト，ともに順序組のときは O_1 の属性 A の式も O_2 の同じ属性の式の部分オブジェクトになるときとする．これに，O_1 は自分自身の部分オブジェクトであることと，“TOP” オブジェクト ▽ と “BOTTOM” オブジェクト△を加えればよい．

【例 5.1】

$[a:1,b:\{x\}] \leqq [a:1,b:\{x,y\},c:z]$

$\{[a:1],[a:1,b:2]\} \leqq$ かつ $\geqq \{[a:1,b:2]\}$

■

例 5.1 のように，≧は部分順序とならない．2項関係 $p \sim q$ を $p \leqq q$ かつ $p \geqq q$ とすると同値関係となり，オブジェクト集合をこれで分類すれば部分順序となる．よってオブジェクト集合は最小上界，最大下界が定義できる．文献 26)は，これらを計算過程と見なし項書き替えのモデル化を試みている．

オブジェクトの操作については，束になる性質を用いて確定ホーン節形式の論理プログラムが提案され，その形式的意味が考察されている[27]．式 E の基礎オブジェクト O による解釈を次とする：$E(O)=\{E'|E'$ は O の基礎部分オブジェクト，E' は E の代入結果$\}$．すなわち，$E(O)$ とは，E にマッチするすべての O の部分オブジェクトの集合であり，束になる性質からこの結果は再び O の部分オブジェクトとなる．規則 r は次の形である：$E_1 \leftarrow E_2$，ここで E_1, E_2 はともに式とする．規則 r のオブジェクト O による効果とは次を言う：$r(O)=\{E_1'|E_2'$ は O の部分オブジェクト，E_1' はこのときの代入による E_1 の結果$\}$ $r(O) \leqq O$ のとき，O は規則 r で閉じていると呼ぶ．この拡張ホーン節論理では，否定や関数を許さない．

この定義では，プログラム(規則集合) R と解釈（オブジェクト）O が与えられたとき，次の性質を満たす最小モデル M が一意に定まることが知られている：(1) M は O を含む，(2) M は R の各元 r で閉じている．M を R の形式的意味と言う．また，さらに拡張し限量記号を許したO論理が提案されている[23,29]．

この操作の解釈は，等号によるマッチではなく，部分オブジェクトに基づいている．このため，例 5.2 のように自然結合が表現できない[28]．

【例 5.2】

$R\{[A:x,C:z]\} \leftarrow S\{[A:x,B:y]\} \wedge T\{[B:y,C:z]\}$.

従来のホーン節論理では自然結合に対応しているが，束によるO論理ではそうではない：y としてボトム（束の最小元）を代入すれば x, z は B 上の値と独立に設定できる．これは，部分オブジェクトであれば式右辺に代入が可能なことに起因する．■

5.2 集合型を許すホーン節データベース

確定ホーン節データベースでは個々の事実を問うことは可能だが部分または全体を一つのものとして扱うことはできない．変数は個体を表し，集合を値とする場合や集合表現は許されないのである．

もっとも直接的な拡張は，集合型の類を導入した多類論理である．多層論理 (MLL) は，多類論理の類を拡張してべき集合化オペレータを加え，この上の変数を許すものでトートロジ判定に関して部分的可解性を有する[30]．ただ集合は各基本型からべき集合対応によって生成されるものだけを扱う．

言語 LPS は集合変数を導入し，次の形の確定ホーン節だけを許す[31]：

$A \leftarrow (\forall x_1/X_1)..(\forall x_n/X_n)B_1 \wedge .. \wedge B_n$

ここで A，$B_1..B_n$ は多類原子式，X_i は集合型変数，x_i は個体変数を表す．x_i は A に生じてはいけない．次の例は二つの集合の関連（性質）を表すのに十分なクラスであることを示している：

【例 5.3】

include$(X,Y) \leftarrow (\forall x/X) x \in Y$

disjoint$(X,Y) \leftarrow (\forall x/X)(\forall y/Y) x \neq y$ ■

LPS ではこれまで知られている確定ホーン節論理プログラムの主だった性質，たとえば極小モデルの一意存在などを有し，SLD 導出に似た完全な計算過程が存在する．現在 LPS は任意の集合が取れるように拡張されて ELPS として提案され，次に述べる LDL との関連が論じられている[32]．

内包的な集合の定義を許すため，グループ化規則を有する言語がいくつか提案されている．このうち LDL と COL は特に著名である．LDLでは，変数は個体または集合を表し，集合の深さも任意である（述語は型付けされない）[33]．集合は外延的に与えてもよい．LDL はこれらから LPS よりも大きい表現力を有する．グループ化規則は例 5.4 のように，集合を規定する規約を前提部に含む．

【例 5.4】

$p(x, \langle y \rangle) \leftarrow r(x, y)$

$r(x, y)$ となるすべての y に対し p の第二項の集合の元とする．■

LDL では一般に極小モデルが存在しても一意とは限らない．このため，集合項の包含関係まで極小性の意味を拡張し，また結論部にしか集合変数を有さず関数もないとする．否定がなく，集合の構成規則が層状になっていれば，極小モデルはただ一つ存在し，否定に関しても層状であれば，標準モデルが存在して，否定を含まないように変換できる．グループ化規則は極小モデルの計算過程で使用され，有限であることがあらかじめ仮定されている．この前提を除くと，否定が許されなくても標準モデルの存在は保証されず，また存在しても極小モデルは一意だが標準モデルとは限らない[34]．

LDL とは異なり，すべての述語に型付けをした言語 COL がある[35]．COL では，関数を論理プログラム（規則）で定義する．この方法は集約関数の定義と類似しているが意味の定義方法が異なる[46]．次は例 5.4 と同じ処理を表す．

【例 5.5】

$y \in f(x) \leftarrow r(x, y)$

$p(x, f(x)) \leftarrow r(x, y)$ ■

最初の式は，関数 f の定義を行っておりグループ化規則に相当する．COL の優れた点は，集合値を変数や関数に導入して，論理プログラムで定義可能としたことである．ただ，この方式でも極小モデルの存在の一意性は保証されず宣言的意味の導入が難しい，またグループ化規則にともなう有限性の保証ができない．

否定および定義関数に関して層状プログラムを考えることができる．$x \in f(y)$ の形で定義される関数 f は使用する前に，完全に解釈が定まるように制限する．この方法で標準モデルが計算でき，層分解の方法に依存しない．このとき，複合オブジェクト論理を関数を含むように拡張した安全なものと表現力が等価である[17]．

構造化をプログラムの変換技法と考え，複合オブジェクトを含まない確定ホーン節データベースを「折り畳む」研究がある[36]．LDL と同様にグループ化規則を用いてプログラムを変換したとき，二つのプログラムのモデルが対応関係をもつのである．実際，確定ホーン節プログラムならば，最小モデルと変換されたプログラムの最小モデルとが 1 対 1 の関係があり，一般プログラムの場合でも変換されたプログラムの極小モデルが元の極小モデルになる．

6. 結　び

本稿では，複合オブジェクトに基づく演繹データベースを，非正規関係，データベース論理，拡張ホーン節論理の三つの立場から考察した．この研究の動機は非正規関係から開始されているが，複合オブジェクトのための論理[17]はデータ操作に関して一定の答を示しているとみてよい．データベース論理の立場から形式化と証明手続きについて述べた．一般には完全性は得られないが，型解釈を制約した完全なモデルについて紹介した．拡張ホーン論理は，グループ化による集合の意味付けでネスト操作の拡張と考えられる．さまざまな構文と，宣言的意味の導入について触れた．紙面の関係から触れることができなかったが，データベースに対する問い合せ（データベース質問）の表現能力についても研究が進んでいる．一般的な複合オブジェクト操作は高階論理の表現力を有し，相当大きなクラスの質問能力をもつことが明らかになっている[28),37)-44)]．

謝辞　日頃ご指導いただいている小林功武教授（産能大学），有澤博助教授（横浜国立大学）に感謝します．また横田一正氏 (ICOT) の丁寧なコメントにも感謝します．ICOT・DOO 作業グループ（主査・田中譲北海道大学助教授）での討論は本稿に強い影響を与えました．

参 考 文 献

1) Gallaire, H., Minker, J. and Nicolas, J-M.: Logic and Databases: A Deductive Approach, ACM Comput. Surv., Vol. 16, pp. 153-185 (1984).
2) 勝野：演繹データベースの動向，アドバンストデータベースシンポジウム，pp. 81-90 (1987).
3) Gallaire, H. and Minker, J. (eds.): Logic and Data Bases, Plenum (1978).
4) Reiter, R.: Towards a Logical Reconstruction of Relational Database Theory, in On Conceptual Modeling, M. L. Brodie. et al. (eds.), Springer-Verlag, pp. 191-233 (1984).
5) Lloyd, J. W.: Foundations of Logic Programming (2nd), Springer-Verlag (1987).
6) Ullman, J. D.: Principles of Database and Knowledge-base Systems, Vol. 1 and 2, Computer Science Press (1988).
7) 三浦：非正規関係データベース論理の動向，ア

ドバンストデータベースシンポジウム (1987).

8) Hull, R. and King, R.: Semantic Database Modeling, ACM Comput. Surv., Vol. 19-3, pp. 201-260 (1986).

9) Bancilhon, F.: Object Oriented Databases, PODS, pp. 152-162 (1988).

10) Smith, S. and Smith, D.: Data Base Abstractions—Aggregation and Generalization, ACM TODS, Vol. 2-2, pp. 105-133 (1977).

11) Hull, R.: A Survey of Theoretical Research on Typed Complex Database Objects, in Databases (ed. J. Paredaens), pp. 193-256 (1987).

12) Hull, R. and Yap, C. K.: The Format Model, J. ACM, Vol. 31-3, pp. 513-518 (1984).

13) Zaniolo, C.: Represention and Deductive Retrieval of Complex Objects, VLDB, pp. 458-469 (1985).

14) Jacobs, B. E.: On Database Logic, J. ACM, Vol. 29-2, pp. 310-332 (1982).

15) Van Gucht, D. and Fischer, P.: Some Classes of Multilevel Relational Structures, pp. 60-69, PODS (1986).

16) Roth, M., Korth, H. and Silberschatz, A.: Extending Algebra and Calculus for 1NF Relational Databases, ACM TODS, Vol. 13-4, pp. 389-417 (1988).

17) Abiteboul, S. and Beeri, C.: On the Power of Languages for the Manipulation of Complex Objects, INRIA Internal Report (1988).

18) Jaeschke, B. and Schek, H. J.: Remarks on the Algebra of Non First Normal Form Relations, PODS (1982).

19) Aho, A. and Ullman, J.: Universality of Data Retrieval Languages, POPL, pp. 110-117 (1979).

20) Bancilhon, F. and Ramakrishnan, F.: An Amateur's Introduction to Recursive Query Processing Strategies, SIGMOD, pp. 16-52 (1987).

21) Abiteboul, S. and Hull, R.: Restructuring Hierarchical Database Objects, INRIA Internal Report (1986).

22) Kifer, M. and Lausen, G.: F-Logic, SIGMOD (1989).

23) Kifer, M. and Wu, J.: A Logic for Object Oriented Logic Programming, PODS, pp. 379-392 (1989).

24) 三浦, 有澤: 意味データモデルの論理アプローチ, 電子情報通信学会データ工学研究会, DE 89-9 (1989).

25) Jaffar, J., Lassez, J. L. and Lloyd, J. W.: Completeness of the Negation as Failure Rule, IJCAI-83, pp. 500-506 (1983).

26) Ait-Kaci, H.:Type Subsumption as a Model of Computation, Expert Database Systems, pp. 115-139 (1986).

27) Bancilhon, F. and Koshafian, S.: A Calculus for Complex Objects, PODS, pp. 53-59 (1986).

28) Hull, R. and Su, J.: Untyped Sets, Invention and Computable Queries, PODS, pp. 347-359 (1989).

29) Maier, D.: A Logic for Objects, pre-print for Found. of Deductive Databases and Logic Programming (1986).

30) 宇田川, 大須賀: 多層論理式のデータベース質問言語への応用, 情報処理学会, データベース管理システム研究会 25-1 (1981).

31) Kuper, G. M.: Logic Programming with Sets, PODS, pp. 11-20 (1987).

32) Kuper, G. M.: On the Expressive Power of Logic Programming Languages with Sets, PODS, pp. 10-14 (1988).

33) Beeri, C., Naqvi, S. et al.: Sets and Negation in a Logic Database Language, PODS, pp. 21-37 (1987).

34) Shmueli, O. and Naqvi, S.: Set Grouping and Layering in Horn Clause Programs, ICLP, pp. 152-177 (1987).

35) Abiteboul, S. and Grumbach, S.: COL, A Logic Based Language for Complex Objects, INRIA Report 714 (1987).

36) 三浦: 演繹情報の非正規表現, 情報処理学会論文誌, Vol. 30, No. 9, pp. 1135-1143 (1989).

37) Chandra, A. and Merlin, P.: Optimal Implementation of Conjunctive Queries in Relational Databases, ACM Symp. Theory of Computing (1977).

38) Aho, A., Sagiv, Y. and Ullman, J.: Equivalences among Relational Expessions, SIAM J. Comp., Vol. 8, pp. 218-246 (1979).

39) Chandra, A. and Harel, D.: Structure and Complexity of Relational Queries, JCSS, Vol. 25-1, pp. 99-128 (1982).

40) Chandra, A. and Harel, D.: Horn Clause Queries and Generalization, J. Logic Programming, Vol. 1-1, pp. 1-15 (1985).

41) Immerman, N.: Relational Queries Computable in Polynomial Time, Information and Control, Vol. 68, pp. 86-104 (1986).

42) Chandra, A.: Theory of Database Queries, PODS, pp. 1-9 (1988).

43) Chandra, A. and Harel, D.: Computable Queries for Relational Databases, JCSS, Vol. 21-2, pp. 156-178 (1980).

44) Hull, R. and Su, J.: On the Expressive Power of Database Queries with Intermediate Types, SIGMOD, pp. 39-51 (1988).

45) Apt, K. R., Blair, H. and Walker, A.: Towards a Theory of Declarative Knowledge, Found. of Logic Programming and Deductive

Databases, Morgan Kauffman (1988).
46) Miura, T. and Shioya, I.: Aggregate Horn Logic and its Semantics, アドバンストデータベースシンポジウム, pp. 21-32 (1988).
47) Enderton, H.: A Mathematical Introduction to Logic, Academic Press (1972).
48) Beeri, C.: Data Models and Languages for Data Bases, ICDT, pp. 19-40 (1988).
49) Abiteboul, S., Beeri, C. et al.: An Introduction to the Completeness of Languages for Complex Objects and Nested Relations, in Nested Relations and Complex Objects in Databases (ed. Abiteboul. et al.), Springer-Verlag (1989).

(平成元年10月2日受付)