数理解析研究所講究録
第731巻 1990年 250-261

250

オブジェクト指向データベースにおける参照によるオブジェクトの関連

九州大学大型計算機センター 古川哲也 (Tetsuya Furukawa)
九州大学工学部 上林彌彦 (Yahiko Kambayashi)

1. まえがき

計算機の利用分野の広がりにともない，従来のデータベースの機能の拡張が求められている．オブジェクト指向モデルは，利用者インタフェースとしての使い易さやデータの表現能力の豊かさから，データベースと応用プログラムを統合したオフィスシステムなどに適していると考えられている．本稿では，参照によってオブジェクト間の関連を表すことによるオブジェクト指向データベースの設計に関する問題ついて議論する．

オブジェクト指向データモデルは，実世界のもの（オブジェクト）をデータベース内のオブジェクトと1対1に対応させてデータを記憶するものである．各オブジェクトには適用可能なメソッドが定義されており，

(1) 実世界とシステム内のデータの直接対応であるオブジェクト同一性
(2) 複雑な構造を持つオブジェクト（複合オブジェクト）の表現
(3) メソッドによるデータと手続きの一体化
(4) 汎化（IS-A）階層

などの特徴がある．しかし，データベースの基本的な機能は，大量のデータを正しく蓄えることであり，いかにデータ間の関連を表現するかが重要となる．

オブジェクト指向データベースに関する研究では，関係データベースのオブジェクト指向への拡張[1)]や，Smalltalk などのオブジェクト指向プログラミング言語によるデータベースシステム[2)]，ＣＡＤなどの応用のためのシステム[3)]など，システム開発が中心[4)]であった．モデルに対する共通の認識がなく，そのため，オブジェクト指向モデルに対する形式的な議論が望まれ[5)]，モデルの形式的な定義[6)]やメソッドに対する議論[7)]なども行われている．

本稿では，参照を用いたオブジェクト間の関連の表現に注目し，オブジェクト

指向モデルの特徴とそれらの参照との関係から，データベースの設計時に考慮すべき問題点について議論する．データベースの設計基準として，冗長性の削減，意味制約の保持，質問処理の効率化がある．オブジェクトに値を持たせるより，その値については他のオブジェクトを参照する方が，冗長性や意味制約の保持に有効である．そのため，参照によるオブジェクトの値の表現について形式的な定義を行い，意味制約として与えられた関数従属性を保持する方法を示す．

2．基本的事項

オブジェクトの値とオブジェクト間の関連を，参照を用いて次のように定義する．

n個の定義域を$D_1, D_2, \cdots, D_n (n \geq 1)$，識別子と呼ばれるシンボルの集合を**ＩＤ**とする．D_iの要素，**ＩＤ**の要素の組 $\langle id_1, id_2, \cdots, id_m \rangle$，**ＩＤ**の要素の集合$\{id_1, id_2, \cdots, id_m\}$を値という．オブジェクトoは，識別子id，値v，メソッドの集合mの組o(id,v,m)である．(id,v,m)は，識別子idのオブジェクトは値vを持ち，メソッドの集合mを適用できることを表す．id(o)をオブジェクトoの識別子，vl(o)をoの値，mt(o)をoのメソッドとする．vl(o)が定義域の要素dのとき，id(o)をdで表すこともある．また，ob(id) で識別子がidであるオブジェクトを表す．本稿では，オブジェクトの参照関係に注目するため，o(id,v,m)を単に識別子idと値vで表す．

オブジェクトのクラスRは，構造Sとメソッド集合Mからなり，R(S,M) で表す．Sは，基本構造D，組構造$\langle A_1:R_1, A_2:R_2, \cdots, A_n:R_n \rangle$，集合構造A:{Q} のいずれかである．Dは定義域名，A_i,A は属性名，R_i,Q はクラス名である．このとき，属性A_i(A)はクラスR_i(Q)を参照するといい，Rが参照するクラスの集合をRef(R) で，参照の閉包を$Ref^+(R) = \{Q \mid Q = R$または$Q \in Ref^+(P), P \in Ref(R)\}$で表す．$\{A_1, A_2, \cdots, A_n\}$，Aをクラスの属性集合といいat(R)で，Rのメソッド集合Mをmt(R)で表す．また，クラスの集合**R**に対して，$at(\mathbf{R}) = \{A \mid A \in at(R), R \in \mathbf{R}\}$，$mt(\mathbf{R}) = \{m \mid m \in mt(R), R \in \mathbf{R}\}$とする．

クラス間の汎化階層を，ISA(Q,R)（RはクラスQを汎化したもの）で表す．汎化には次の性質がある．

(1) メソッドの継承：$mt(Q) \supseteq mt(R)$

QはRの特殊なものであり，Rのオブジェクトに適用可能なメソッドは，Qのオブジェクトにも適用できる．

(2) 属性の継承：$at(Q) \supseteq at(R)$

Rのオブジェクトが属性Aの値を持てば，Qにも属性Aが定義されQのオブジェクトもAの値を持つ．

データベースのスキーマ**S**は，構造が組または集合であるクラスの集合**C**とクラス間の汎化階層の集合**ＩＳＡ**であり，**S(C，ＩＳＡ)**で表す．属性Aがデータベーススキーマ中ですべて同じクラスRに対応づけられているときは，AをクラスRで表すこともある．

スキーマ**S(C，ＩＳＡ)**で，クラスR(S,M)（$\in$**C**）の状態は，Rのオブジェクトの集合である．各定義域D_iに対し，Sが基本構造D_iであるクラスが存在する．このクラスのオブジェクトの値は，定義域D_iの要素である．Sが組構造 $\langle A_1:R_1, A_2:R_2, \cdots, A_n:R_n\rangle$ であることは，このクラスのオブジェクトの値は組$\langle id_1, id_2, \cdots, id_n\rangle$（$id_i$はクラス$R_i$のオブジェクトの識別子）であることを意味する．Sが集合構造{Q}のとき，このクラスのオブジェクトの値は集合値$\{id_1, id_2, \cdots, id_m\}$（$id_i$はクラスQのオブジェクトの識別子）である．

スキーマが**S(C，ＩＳＡ)**であるデータベースの状態ρは，クラスR（$\in$**C**）の状態の和集合，即ちデータベース中のオブジェクトの集合でる．

スキーマ**S**と状態ρで，オブジェクトoの属性Aの値をA(o)で，属性集合Xの値を$X(o) = \{A(o) \mid A \in X\}$で表す．また，オブジェクトoのクラスの属性集合をat(o)で，オブジェクト集合**o**の属性集合を$at(\mathbf{o}) = \{at(o) \mid o \in \mathbf{o}\}$で表す．

クラスRのオブジェクトoに適用可能なメソッドの集合は，$mt(o) \cup mt(R)$である．mt(o)とmt(R)に同じ名前のメソッドが存在するときには，名前によるメソッドの実行はmt(o)のメソッドが優先する．

オブジェクトの参照関係を，参照グラフG(V，E）で表現する．Vはクラスと1対1に対応する節点の集合，Eは節点間の有向枝の集合であり，R_1がR_2を参照する属性を含むとき，R_1からR_2へ向かう枝が存在する．簡単のため，参照グラフは閉路を持たないと仮定する．

節点R_1, R_2で、枝(R_1, R_2)が存在するかまたはR_1からR_3，R_3からR_2へ到達可能な節点R_3が存在するとき，R_1はR_2へ到達可能であるという．$Ref^+(R)$はRおよびRか

ら到達可能な節点のクラスの集合である．$Ref^-(R)$ をRへ到達可能な節点の集合 $\{Q \mid R \in Ref^+(Q)\}$ とする．クラスRのオブジェクトは，$Ref^+(R)$ に含まれるの基本構造クラスのオブジェクトの値で特徴づけられ，それらの値は $Ref^-(R)$ のオブジェクトを特徴づける値の一部となっている．

クラスRの基本構造のクラスを参照する属性集合を $at\_b(R)$，クラス集合 $\mathbf{R}$ に対し，$at\_b(\mathbf{R}) = \{A \mid A \in at\_b(R), R \in \mathbf{R}\}$ とする．

［例１］　図１(a),(b) はそれぞれスキーマ，状態の例である．id_{ij} は識別子である．属性が基本構造を参照している場合は，識別子ではなく，その値（野球，南野，工など）で表している．例えば，識別子 id_{31} のオブジェクトは学生名が“南野”で，識別子が id_{21} のオブジェクトのクラブに所属しており，学部名は“工”，学科名は“情報工”である．□

部室<部屋番号>	id_{11}<A103>, id_{12}<A105>
↑	id_{21}<野球, id_{11}>, id_{22}<バレー, id_{12}>
クラブ<クラブ名,部室>	id_{31}<南野, id_{21}, 工, 情報工>
↑	id_{32}<浅香, id_{22}, 工, 情報工>
学生<学生名,クラブ,学部名,学科名>	id_{33}<山瀬, id_{21}, 理, 物理>
(a)	(b)

図１　スキーマと状態

3．参照によるオブジェクトの等価性

データベース内でのオブジェクトの値の表現法として，すべて基本オブジェクトを参照する方法と，構造オブジェクトの参照を許す方法がある．値a,b,c,dを持つオブジェクトoは，

(1) id_1<a,b,c,d>

(2) id_1<a,b,id_2>, id_2<c,d>

などのように表現できる．値の構造（入れ子など）が本質的でないとき，(1), (2) の id_1 はオブジェクトoの表現については同等である．(2)で id_2 のオブジェクトの存在は，データベースのビューとして id_2 を認識する必要があるかどうかの問題である．

値の構造が本質的な違いとなるとき，どの属性値を構造オブジェクトの参照にするかによってオブジェクトが正しく表現されないことがある．オブジェクトoが，(2)のように<c,d>を入れ子とする構造のとき，

(3) id_1<a,id_3,d>, id_3<b,c>

はそのようなオブジェクトを表現できておらず，(1) も表現できていないとすべきである．

(2)と(3)は，(1)のオブジェクトの内部表現とすることができる．次に，参照を用いたオブジェクトの表現を定義し，参照に関するスキーマの性質を示す．

オブジェクトの表現法はスキーマにより記述される．スキーマS_1のオブジェクトをスキーマS_2のオブジェクトが表現していることを次のように定義する．

［定義１］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$とその状態ρ_1, ρ_2で，ρ_1, ρ_2でのオブジェクトo_1, o_2について，以下の条件を満たすとき，o_2はo_1を表現するといい，$o_1 < o_2$で表す．

(1) o_1, o_2はともに基本構造オブジェクトであり，その値は等しい

または，

(2) $at(o_1) = \{A_i(1 \leq i \leq n),\ B_{jk}(1 \leq j \leq m, 1 \leq k \leq n_j)\}$,

$at(o_2) = \{A_i(1 \leq i \leq n),\ B_j(1 \leq j \leq m),\ C_k(1 \leq k \leq l)\}$で，

(2-1) $ob(A_1(o_i)) < ob(A_2(o_2))$ $(1 \leq i \leq n)$

(2-2) o_1のB_{jk} の値から新たにオブジェクトを作ると，

$$<B_{j1}:B_{j1}(o_1), B_{j2}:B_{j2}(o_1), \cdots, B_{jnj}:B_{jnj}(o_1)> < ob(B_j(o_2))\ (1 \leq j \leq m)$$ □

(2-1) は，o_1が参照するオブジェクトを表現しているオブジェクトをo_2が参照していることを，(2-2)は，o_1の属性集合$\{B_{jk}(1 \leq k \leq n_j)\}$の値は，$o_2$では$B_j$の属性値で参照しているオブジェクトで表現されていることを示している．

オブジェクトの表現には次の性質がある．

［補題１］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$とその状態ρ_1, ρ_2で，ρ_1, ρ_2のオブジェクトo_1, o_2とo_1, o_2のクラスR_1, R_2について$o_1 < o_2$ならば，$at\_b(Ref^+(R_1)) \subseteq at\_b(Ref^+(R_2))$. □

（証明） $o_1 < o_2$ の判定は定義より， $Ref^+(R_2)$以外のクラスには無関係であり， $at(Ref^+(R_2))$以外の属性を用いない． 属性 $A \subseteq at_b(Ref+(R_1))$で $A \not\subseteq at_b(Ref+(R_2))$となるものがあれば， $at(Ref^+(R_2)) \subseteq at_b(Ref+(R_2))$より， Aは再帰的な定義のどのステップでも (1) には適用できない． $o_1 < o_2$ であれば $at_b(Ref+(R_1))$のすべての属性は(1)で用いられるので， $A \subseteq at\_b(Ref+(R_1))$ ならば $A \subseteq at\_b(Ref+(R_2))$， 即ち， $at\_b(Ref^+(R_1)) \subseteq at\_b(Ref^+(R_2))$． （証明終り）

スキーマS_1のいかなる状態もスキーマS_2のある状態で表現できるかどうかを判定する．即ち，S_2はS_1の他の表現法であることの判定である．そのために，まずS_1のクラスに対応するクラスがS_2に存在することを定義し，S_1の状態がS_2の状態で表現されていることを定義する．

［定義２］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$で， C_1からC_2への写像fは，１対１の全写像，即ちS_1のすべてのクラスR（$\in C_1$）についてS_2のクラスf(R)（$\in C_2$）が存在し，$R_1 \neq R_2$ならば$f(R_1) \neq f(R_2)$のとき， S_1からS_2へのクラス写像であるという．□

［定義３］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$の状態ρ_1, ρ_2で，クラス写像fが存在し， すべてのクラスR（$\in C_1$）についてRのオブジェクトとf(R)のオブジェクトは１対１に対応し， 対応するオブジェクトについて，$o_1 < o_2$であれば，ρ_2はρ_1を表現するといい，$\rho_1 < \rho_2$で表す．□

スキーマS_2のオブジェクトの表現法で，S_1の任意の状態についてそれを表現するS_2の状態が存在しなければならない． また，S_2の状態で，S_1のいかなる状態も表現していないものがあれば， そのような状態はS_1では存在しえないものである．次にスキーマの表現性を定義し，その性質を示す．

［定義４］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$とあるクラス写像fについて， S_1の任意の状態ρ_1に対して$\rho_1 < \rho_2$ となるS_2の状態ρ_2が存在し，S_2の任意の状態ρ_2に対して$\rho_1 < \rho_2$となるS_1の状態ρ_1が存在するとき，S_2はS_1を表現するといい，$S_1 < S_2$で表す．□

［定理１］ スキーマ$S_1(C_1, ISA_1)$, $S_2(C_2, ISA_2)$で， $S_1 < S_2$であ

るための必要十分条件は，C_1のすべてのクラスRについて

$at\_b(Ref^+(R)) \subseteq at\_b(Ref^+(f(R))$

である．□

（証明）　$S_1 < S_2$であれば，C_1のすべてのクラスRについて，Rのオブジェクトとf(R)のオブジェクトは1対1に対応し，対応するオブジェクトo_1, o_2は，$o_1 < o_2$である．補題1より，$o_1 < o_2$であれば，
$at\_b(Ref^+(R)) \subseteq at\_b(Ref^+(f(R))$.

C_1のすべてのクラスRについて$at_b(Ref^+(R)) \subseteq at_b(Ref^+(f(R))$とする．$S_1$の状態$\rho_1$に対し，$S_2$の状態$\rho_2$を次のように定める．Rとf(R)のオブジェクトを1対1に対応させる．基本構造オブジェクトのみを参照するクラスRについて，Rのオブジェクトo_1に対応するS_2のオブジェクトo_2の値を共通属性は同じ値にする．f(R)に含まれない属性は，参照するクラスに含まれるので，o_1と同じ値を持つオブジェクトを参照するようにする．これにより，$o_1 < o_2$である．組構造オブジェクトを参照する属性を含むクラスのオブジェクトo_1で，参照するオブジェクトがすべてS_2のオブジェクトで表現されていれば，対応するオブジェクトo_2もo_1の参照するオブジェクトを参照することにより，$o_1 < o_2$とすることができる．従って，$\rho_1 < \rho_2$となる．S_2の状態ρ_2に対しても同様に$\rho_1 < \rho_2$となるS_1の状態ρ_1を定めることができる．よってこのとき，$S_1 < S_2$である．（証明終り）

4．参照による冗長性の削減

データの冗長性や更新処理を考えると，オブジェクトの値をすべて基本構造オブジェクトの参照にせず，他の組構造オブジェクトを参照する方がよい場合がある．値が基本構造オブジェクトの参照のみからなるときの問題点には次のものがある．

(1) データの冗長性：属性集合Xの値が等しいオブジェクトが複数あれば，Xのデータの関連が複数のオブジェクトで記憶されていることになり，冗長である．

(2) 更新時の検査：Xの値を変更する際，Xを含むすべてのクラスのオブジェクトを調べなければならない．

(3) 削除による関連の喪失：オブジェクトoを削除する際，X(o)が等しいオブジェクトが他に無ければ，X(o)の関連がデータベースから失われてしまう．

属性集合Xからなるクラスを作り，Xの値はそのクラスのオブジェクトを参照するようにすれば，(1)～(3)の問題に対処できる．

［例２］　図２(a)では，オブジェクトはABの値を重複して持つ．図２(b)のようにABをクラスR_1として独立させ，もとのクラスRに対応するクラスR_2のオブジェクトは，ABの値についてはR_1のオブジェクトを参照するようにすれば，ABについての冗長性を削減できる．□

$R\langle A,B,C\rangle$ $id_1\langle a_1,b_1,c_1\rangle$
$id_2\langle a_1,b_2,c_2\rangle$
$id_3\langle a_1,b_1,c_2\rangle$
$id_4\langle a_1,b_2,c_3\rangle$

(a)

$R_1\langle A,B\rangle$ $id_{11}\langle a_1,b_1\rangle$
$id_{12}\langle a_1,b_2\rangle$
$R_2\langle R_1,C\rangle$ $id_{21}\langle id_{11},c_1\rangle$
$id_{22}\langle id_{12},c_2\rangle$
$id_{23}\langle id_{11},c_2\rangle$
$id_{24}\langle id_{12},c_3\rangle$

(b)

図２　参照による冗長性の削減

複数のクラスにXがある場合も，Xをクラスとすることにより，１ヶ所で記憶できる．クラスRの属性集合Xによる分解，および分解後のオブジェクトの対応の決定は次のように行う．

［手続き１］　クラスRの属性集合X（$\subseteq at(R)$）による分解

1．R_1を$at(R_1)=X$であるクラスとする．
2．R_2を$at(R_2)=(at(R)-X)\cup A$（AはクラスR_1を参照する属性）とする．
3．R_1のオブジェクト集合を{o | ∃Rのオブジェクトo'，$vl(o)\subseteq vl(o')$，id(o)は任意の値}とする．
4．R_2のオブジェクト集合を{o | ∃Rのオブジェクトo'，$at(R_2)-A$ の値はo'の値に含まれる，Aの値は$vl(o'')\subseteq vl(o')$ となるR_1のオブジェクトo''の識別子，$id(o)=id(o')$}とする．
5．Rを参照しているクラスをR_2を参照するようにする．

6．スキーマ{C，ＩＳＡ}を{C'，ＩＳＡ'}とする．

C' = (C-{R}) ∪ {R_1, R_2}

ＩＳＡ' = (ＩＳＡ-{ISA(R', R") | R' = RまたはR" = R})

∪ {ISA(R_2, R') | ISA(R, R') ∈ ＩＳＡ}

∪ {ISA(R', R_2) | ISA(R', R) ∈ ＩＳＡ} □

RのオブジェクトはR_2のオブジェクトと１対１に対応し，その値は対応するR_2のオブジェクトとその参照するR_1のオブジェクトの値として得ることができる．従ってR_2のメソッドは，オブジェクトoでR_2から除かれた属性Xの値X(o)を用いる部分をoの参照するオブジェクトA(o)のXの値X(A(o)) を用いるように変えればよい．

［定理２］　スキーマS_1とS_1のクラスを手続き１を用いて分解したスキーマS_2では，$S_1 < S_2$．□

（証明）　S_1の状態ρ_1に対し，ステップ３，４のようにS_2の状態ρ_2を決めると$\rho_1 < \rho_2$．　また，S_2の状態ρ_2に対してR_2の参照するXの値をS_2のRのオブジェクトの値とすることで，$\rho_1 < \rho_2$となるS_1状態ρ_1が作れる．（証明終り）

5．関数従属性を用いた分解

データベースでは，データの持つ制約を保持することが１つの重要な要素である．本節では，基本的な制約である関数従属性のオブジェクト指向データベースにおける保持について議論する．

関数従属性は，データ間の対応関係が多対１であることを示す基本的な制約である．例えば，学生とその所属する講座の対応は多対１であり，各学生に対して所属講座は一意に決まり複数の講座には所属していない場合などである．

オブジェクト指向モデルにおける関数従属性を次のように定義する．

［定義５］　クラスRと属性集合 X, Y (⊆at(R)) で，Rの任意のオブジェクトo_1, o_2に対し，$X(o_1) = X(o_2)$ならば$Y(o_1) = Y(o_2)$のとき，Rのオブジェクト集合は関数従属性：X→Yを満足するという．　X→Yは，Xの参照するクラスの識別子集合を定めるとYの参照するクラスの識別子集合が一意に定まることを表す．　このよ

うな関数従属性をクラス内の関数従属性という．□

［定義６］　クラスRのオブジェクトを１つ定めるとそのオブジェクトの値が定まる．即ち，Rのオブジェクトの識別子から参照するクラスのオブジェクトの識別子が定まる．この性質をクラス間の関数従属性といい，R→at(R)で表す．□

データベースで満足すべき関数従属性集合が与えられているとき，関数従属性：X→Yについても４節での議論と同様の問題が生じる．

(1) データの冗長性：Xの値が等しいオブジェクトが複数あれば，XとYの同じデータの関連が複数のオブジェクトで記憶されていることになり，冗長である．

(2) 更新時の検査：新しくオブジェクトを追加するときや，オブジェクトの値の変更をする際に，X→Yを満たしているかどうかの検査が必要である．

(3) 削除による関連の喪失：オブジェクトoを削除する際，X(o)が等しいオブジェクトが他に無ければ，X(o)とY(o)の関連がデータベースから失われてしまう．

属性集合XYとクラスR（at(R)⊇XY）について，手続き１を用いてクラスを分解し，XYからなるクラスを新たに作り，RからXYを除いて，その値はXYのクラスのオブジェクトを参照するようにすることにより，RではX→Yによる問題を考える必要がなくなる．

［例３］　図１のスキーマで、クラス“学生”では，学科は１つの学部にのみ所属するので，関数従属性：学科名→学部名を満足する．“学生”を手続き１で分解すると，スキーマ，オブジェクトはそれぞれ図３(a),(b)となる．学科名→学部名はクラス“学科”で保持され，“学生”のオブジェクトは“学科”のオブジェクトを参照する．□

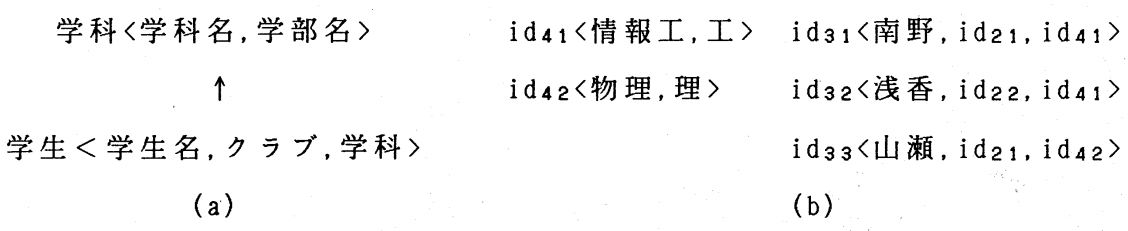

図３　関数従属性による分解

関数従属性集合**F**が与えられているとき，手続き１を用いて**F**中の関数従属性でクラスを分解した結果のスキーマ$\mathbf{S}_2$では，　**F**の満足の検査が複雑になる場合がある．

［例４］　属性集合ABCDからなるクラスRで，関数従属性集合{A→B, BC→D}が与えられているとする．RをA→Bで分解した結果のスキーマでは，BC→Dの満足の検査は，２つのクラスにまたがって行わなければならない．　図４は，BC→Dを満足していない状態の例であるが，参照した値を調べる必要がある．id_{21}はid_{11}を参照しているのでBCの値はb_1c_1であり，id_{22}もid_{12}を参照しているのでb_1c_1である．しかし，Dの値はd_1, d_2であり，BC→Dを満足していない．□

$R_1\langle A, B\rangle$	$R_2\langle C, D, R_1\rangle$
$id_{11}\langle a_1, b_1\rangle$	$id_{21}\langle c_1, d_1, id_{11}\rangle$
$id_{12}\langle a_2, b_1\rangle$	$id_{22}\langle c_1, d_2, id_{12}\rangle$

図４　関数従属性の満足の検査

分解後も他の関数従属性の検査を容易にするため，他の関数従属性の左辺の属性をR_2に残す．右辺の属性がR_2に含まれない場合，それはクラス間の関数従属性として保持される．手続き１のステップ２は以下のようになる．

2'．R_2を$at(R_2)=(at(R)-(W-XY))\cup A$　（Wは他の関数従属性の左辺，　AはクラスR_1を参照する属性）とする．

このとき，R_2に残った属性とR_1の属性に共通のものがあれば，R_2の各オブジェクトoでのその値と，oが参照するR_1のオブジェクトのその属性の値は同じでなければならない．例４の場合，R_2に属性Bを残すので，$B(o)=B(R_1(o))$である．

［制約１］　各オブジェクトについて，参照しているオブジェクトが同じ属性を持てば，その値は等しい．□

６．むすび

オブジェクト指向データベースについて，参照の様々な性質について議論し，データベース設計において考慮すべき問題を示した．データベース設計は，初期

値（データベースのビュー）として与えられたクラスの集合Sと，データの意味制約である関数従属性集合から，Sを表現する範囲で冗長性の少ないスキーマを求めることになる．一般に，関数従属性による分解と，複数のクラスに共通属性があればそれらによる分解で冗長性の少ないスキーマが設計できる．

一方，質問処理効率を考えたとき，値を参照せずオブジェクト内に保持している方がよい場合もあり，これらの関係を明らかにする必要がある．

参考文献

1) Stonebraker, M., "Object Management in POSTGRES Using Procedures," Proc. Int. Workshop on Object-Oriented Database Systems, pp. 66-72, Sept. 1986.
2) Maier, D. and Stein, J., "Development and Implementation of Object-Oriented DBMS," in Research Directions in Object-Oriented Programming, The MIT Press, 355-392, 1987.
3) Banerjee, J., Chou, H., Graza, F., Kim, W., Woelk, D., Ballou, N., and Kim, H., "Data Model Issues for Object-Oriented Applications," ACM Trans. on Database Syst., Vol. 5., No. 1, pp. 3-26, Jan. 1987.
4) Andrews, T. and Harris, C., "Combining Language and Database Advances in an Object-Oriented Development Environment," Proc. OOPSLA '87, pp. 430-440, 1987.
5) Bancilhon, F., "Object-Oriented Database Systems," Proc. ACM Symp. on Principles of Database Syst., pp. 152-162, March 1988.
6) Lecluse, C., Richard, P., and Velez, F., "O_2, an Object-Oriented Data Model," Proc. ACM SIGMOD Int. Conf. on Management of Data, pp. 424-433, June 1988.
7) Hull, R., Tanaka, K., and Yoshikawa, M., "Behavior Analysis of Object-Oriented Databases: Method Structure, Execution Trees and Reachability," Proc. Int. Conf. on FODO, Lecture Notes in Computer Science 367, Springer-Verlag, pp. 372-388, June 1989.
8) Kim, W., Nicolas, J.-M., and Nishio, S. (eds.), Proc. Int. Conf. on Deductive and Object-Oriented Databases, Dec. 1989.
9) Ullman, J. D., Principles of Database Systems, 2nd ed., Computer Science Press, 1982.