解　説

データベースの関係形式*

穂鷹良介** 渋谷政昭***

1. 概　論

関係形式データベース (relational database) とはIBM San Jose 基礎研究所の E. F. Codd が構築した理論的なデータベース・モデルである．彼がこれについてはじめて発表したのは 1969 年の IBM Research Report の論文であるが，彼の発表が一般に知れわたったのは 1970 年の CACM の論文[1]である．それから約5年が経過し，関係形式データベースは NCC のデータベース分科会の話題を半ば独占する程にもなり，内外に多くの支持者を得ている．このモデルの成果は完全な実用段階にはいまだ達していないが，後述のようにそこへの試みも始められており着実な発展を成している．

その特徴とする所は

（1） 高度のデータ独立性
（2） 簡単で理解しやすい汎用的なデータ・モデル
（3） 述語論理に基づく理論的検討とそれらが容易に行われるモデル構成
（4） 照会処理の集合演算化
（5） データの意味について中立的立場にたったデータベース設計の可能性
（6） 概念モデルと外面的モデルの対応の容易さ
（7） 種々の高水準のデータ準言語設計の可能性

などである．特に Codd が最も強調する特徴は（1）のデータ独立性で，これは従来のデータベースがとかくデータ処理プログラムにデータの物理的表現を意識させる結果，データの物理的表現方法が変化した場合にデータが表現している意味が変わらなくてもプログラムの手直しが必要となることをなくそうとするものである．

データベースの関係形式モデルでは情報を表の形で表現し，物理的な表現手段の考慮を一切持ち込まない．また利用者は属性（従来のファイルのフィールドに相当するもの）のみを意識するインタフェースでデータベースとやりとりをするだけなのでプログラムのデータ独立性が高まる仕組みとなっている．

以下関係形式モデルの基本概念を定義し主な成果を説明する．

適当な集合列 $D_1, D_2, \cdots, D_n$ が与えられたものと仮定する．この集合の直積 $\prod_{i=1}^{n} D_i$ の部分集合 R を関係 (relation) と呼び，関係を適当に集めてできる集合族 $\boldsymbol{R}$ を関係の型 (relation type) という．このとき n をその関係あるいは関係の型の次数(degree)といい，各 D_i を定義域 (domain) という．$D_1, D_2, \cdots, D_n$ の中には同じ定義域が出現しても構わない．そこで，これらを区別するために改めて定義域に $A_1, A_2, \cdots, A_n$ という互いに異なる名前を与えることにする．これを属性 (attribute) という．A_i は関係 R あるいは関係の型 $\boldsymbol{R}$ の第 i 属性である．一つの関係の型に属性を併記して

$$\boldsymbol{R}(A_1, A_2, \cdots, A_n)$$

のように表わしたものを関係あるいは関係の型の図式 (schema) と呼ぶ．記号 $\Omega(\boldsymbol{R})$ は属性全体の集合 $\{A_1, A_2, \cdots, A_n\}$ を示す．

関係としては有限集合のみを考える．したがってこれを行列あるいは表の形で表現することができる．

いま

$$R=\{r_i \mid i=1, \cdots, m\} \quad (r_i=(r_{i1}, \cdots, r_{in}))$$

と書き表わすことができたとき第 ij 要素が r_{ij} である $m \times n$ 行列がそれである．r_i を組(tuple) と呼ぶ．

関係を表の形に表現するときには一般性を失わず $r_i \neq r_j$ $(i \neq j)$ とできる．表に同じ組がなく，どの属性もモデルとしてはそれ以上に分割して考えないときにこの表現は第1正規形であるという．以後関係は特に断らぬ限りすべて第1正規形にあるものとする．データベースは関係の型の集りであり，一時点でのデータベースは関係の集りである．関係の型の数，種類は

---

* Relational Database by Ryosuke HOTAKA (Social Science Laboratory Ltd.) and Masaaki SIBUYA (IBM Japan Ltd., Scientific Center).

** ソーシアル・サイエンス・ラボラトリ（株）
*** 日本アイ・ビー・エム（株）サイエンティフィック・センター

急激には変化しないものとする.

2. 集合演算

関係は集合と同一視して良いから，これに各種の集合演算を定義することができる.

R_1, $R_2 \in \boldsymbol{R}$ のとき $R_1 \cup R_2 = \{r \mid r \in R_1$ または $r \in R_2\}$ を R_1 と R_2 の和という.

R_1, $R_2 \in \boldsymbol{R}$ のとき $R_1 \cap R_2 = \{r \mid r \in R_1$ かつ $r_2 \in R_2\}$ を R_1 と R_2 の積という.

R_1, $R_2 \in \boldsymbol{R}$ のとき $R_1 - R_2 = \{r \mid r \in R_1$ かつ $r \notin R_2\}$ を R_1 と R_2 の差という.

$\boldsymbol{R}$, $\boldsymbol{S}$ を2つの関係の型とし，それぞれの次数を n, $n+l$ とするとき $R \in \boldsymbol{R}$, $S \in \boldsymbol{S}$ なる関係に対し，$S \div R = \{q \mid (r, q) \in S, \forall r \in R\}$ を S を R で除した商という．たとえば $S = Q \times R$ (Q と R との直積) のとき $S \div R = Q$ である.

$A \subset \Omega(\boldsymbol{R})$ を空でない属性の集合とする．たとえば $A = \{A_{i_1}, A_{i_2}, \cdots, A_{i_k}\}$ のとき $r \in R$ に対して

$$r.A = (r_{i_1}, r_{i_2}, \cdots, r_{i_k})$$

と定義し

$$P_A(R) = \{r.A \mid r \in R\}$$

と定義する．$P_A(R)$ は R の A の上への射影といわれる．なお $r.A_i$ を $r[i]$ とも書く.

$R \in \boldsymbol{R}$, $S \in \boldsymbol{S}$, $A \subset \Omega(\boldsymbol{R})$, $B \subset \Omega(\boldsymbol{S})$ かつ A と B は同じ定義域を持つとする．このとき

$$R[A\theta B]S = \{(r, s) \mid r.A \theta r.B, r \in R, s \in S\}$$

を R と S の θ 結合 (join) という．ここで θ は2項関係のオペレータでたとえば θ が等号 $=$ のときの結合は R, S を2つのファイルとみなしたとき，共通のキーによって対応するレコード同志を結合したものの全体となる．この場合を，自然な結合 (natural join) と呼ぶ．θ を上と同様の2項関係のオペレータとするとき

$$R[A\theta B] = \{r \mid r.A \theta r.B, r \in R\}$$

を制約 (restriction) という.

データベースの照会回答というものは，以上の諸演算の合成であるとみなすことができる．Codd は関係形式モデルにおける照合回答を Alpha 表現と称する述語論理式で書き表わす方法も示している．これは，上の外延的な表現にたいする内包的な表現であるが，Alpha 表現で与えられる組の集合は，上で述べた集合演算を逐次適用することによって求め得ることが証明されている．ここでは Alpha 表現の文法の詳細を述べることはしないで例示のみに止める (**図-1** 参照).

$\{r_1 \mid r_1 \in R_1 \wedge r_1[3] > a_1\}$
$\{r_2[3] \mid r_2 \in R_2 \wedge \exists r_1 (r_1 \in R_1)(r_1[3] = r_2[5])\}$
$\{(r_1, r_2) \mid (r_1[3] = r_2[5]) \wedge \exists r_3 (r_3 \in R_3)(r_3[1] = a_1)\}$
$\{r_2[1] \mid (r_2 \in R_2) \wedge (r_2[2] \neq 15)\}$

図-1 Alpha 表現の例

$\{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge \forall (r_2 \in R_2) \exists (r_3 \in R_3)((r_1[1] = r_3[1]) \wedge (r_3[3] = r_2[1]))\} = \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge \forall (r_2 \in R_2) \exists (r_3 \in R_3)(\Pi_{i=1}^{2} R_i(r_1[1] = r_3[1]) \cap \Pi_{i=1}^{2} R_i(r_3[3] = r_2[1]))\}$

ここで $T_3 = (\Pi_{i=1}^{2} R_i(r_1[1] = r_3[1]) \cap \Pi_{i=1}^{2} R_i(r_3[3] = r_2[1]))$ とおくと

$= \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge \forall (r_2 \in R_2) \exists (r_3 \in R_3)((r_1, r_3, r_3) \in T_3)\}$
$= \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge \forall (r_2 \in R_2)((r_1, r_3) \in P_{R_3}(T_3))\}$

ここで $T_2 = P_{R_3}(T_3)$ とおくと

$= \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge \forall (r_2 \in R_2)((r_1, r_2) \in T_2)\}$
$= \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge (r_1 \in T_2 \div R_2)\}$

ここで $T_1 = T_2 \div R_2$ とおくと

$= \{(r_1[2], r_1[3]) \mid (r_1 \in R_1) \wedge (r_1 \in T_1)\}$
$= \{(r_1[2], r_1[3]) \mid r_1 \in T_1\}$
$= P_{\{2,3\}}(T_1)$

図-2 Alpha 表現の計算過程の例

また Alpha 表現を集合演算によって計算する過程も例によって示す(**図-2** 参照)．Codd が定義した Alpha 表現はここでの例示と実は多少異なった記号法を用いているが，本質的には同じである.

3. 正規形

A, $B \subset \Omega(\boldsymbol{R})$ のとき $\boldsymbol{R}$ の任意の関係 R, R の任意の2元 r_1, r_2 に対して

$r_1.A = r_2.A$ ならば常に $r_1.B = r_2.B$

であるとき，B は A に関数従属 (functionally dependent) であるといい $A \rightarrow B$, と書く．B が A に関数従属でないときには $A \nrightarrow B$ と書く.

図式 $\boldsymbol{R}(A, B, C)$ を持つ関係の型 $\boldsymbol{R}$ において仮に $A \rightarrow B$, $A \nrightarrow C$ とする．そのときには**図-3** に示すように A, B の組 (a, b) としての同じ値が一般には複数個一つの関係 $R(\in \boldsymbol{R})$ に存在する．B は A に関数従属であるから，もしもこの関数関係が a_1 に対して $b_2 (\neq b_1)$ を対応させるように変化したとすると，(a_1, b_1) の対が図-3 のように3個あれば3個の R の要素について変更しなくてはならない．また上記の関係で a_1, b_1 の対を値としてもつ組が3個とも削除されたとき，a_1 に対応して b_1 が定まるという関数関係そのものがデータベースから失われてしまって不合理である.

この場合には**図-4** (次頁参照) に示すようにもとの

R	$(A$	B	$C)$
	a_1	b_1	c_1
	a_1	b_1	c_2
	a_1	b_1	c_3

図-3 非正規形の例

S (A B)		T (A C)	
a_1	b_1	a_1	c_1
		a_1	c_2
		a_1	c_3

図-4 非正規形の分解

関係の型を $S(A, B)$, $T(A, C)$ の2つの関係の型に分解して，一つの関係の型の中で上述のような関係がなくなるようにするのが望ましい．

このように条件を表現したのがBoyce-Coddの条件と言われるものである[2]．

[Boyce-Codd の条件]

A, $B \subset \Omega(R)$, $A \neq \phi$, $B \neq \phi$, $A \cap B = \phi$, $A \to B$ ならば $A \to \Omega(R)$

正規性の条件としては Codd が文献3）で導入した第3正規形 (Third Normal Form 略して TNF あるいは 3NF などという）が有名であるが，3NF と Boyce-Codd の条件とは実は同じではない．Boyce-Codd の条件が成立すれば常に 3NF であるが逆は必ずしも成り立たない．また3NF の定義はたとえば全依存 (full dependence)，推移依存 (transitive dependence) などやや面倒な概念を必要とするため，ある関係の型が3NF であるかどうかを検証するのに複雑な判断を必要とするなど欠点を有している．また 3NF であって Boyce-Codd でない場合はいわゆるキー破壊的な場合となっているが，3NF は Boyce-Codd 条件にキー破壊的な場合を付け加えた条件にも一致しておらず，それよりも狭い概念となっている（参考文献 4）参照）．実用上は Boyce-Codd 条件を正規条件として考えるだけで十分であろう．

4. データ準言語

データベースについて照会し回答を得る照会言語は汎用計算言語ほどの強い機能をもっていないので，データ準言語(data sublanguage) と呼ぶのが適当なようである．関係形式モデルの特徴の一つは，その枠組の中で種々のデータ準言語を設計できることである．データ準言語も，端末からの直接利用のため，汎用の親言語から参照するため，また，照会回答だけのため，データの更新も含めた使用のため，によって設計方針が異なるが，新しい傾向としては，端末からの照会回答のための準言語に関心が払われている．

（1） 組操作型： 親言語からデータベースを参照するとき，検索条件が複雑であったり，データ更新の計算が複雑であれば，集合にたいする演算は困難となり，個々の組について処理を行うことになる．このような場合には，準言語としては，ある条件を満たす組を1個ずつ取り出してくる機能を満たせばよい．また，このような機能は，多かれ少なかれ集合を扱う準言語の基礎となっている．本誌別稿の BARTH[5] はこの種の言語である．

（2） 述語論理型： **2.** で述べた Alpha がこの型である．論理式を用いれば正確な記述が可能であり，また論理式の性質は詳しく調べられている，という利点があるものの，広汎な利用者は期待できないし，十分に実用的とは言えない．データ準言語は当然 Alpha により記述できる照会を表現できなければならないという要請 (relational sufficiency) の基準としては重要である．

（3） 代数型： 関係形式モデルにおいては照会回答をすべて関係についての集合演算とみなせることを述べたが，演算過程の代数的な記述は明確なデータ準言語であり，計算機処理もし易い．PRTV[6] はこのような準言語を用いている．

（4） 写像型： A, B を共に属性の集合とし，図式が $R(A, B)$ の関係を考える．これは一般に，定義域 A から，B のすべての部分集合の集合族 2^B への写像 f_{AB} とみなすことができる．A の値についての条件を満たす組を求めることは，A の部分集合 A_0 の f_{AB} による像 $f_{AB}(A_0)$ を求めることに他ならない．結合演算も本質的には写像である．本章末で述べる SEQUEL は，なじみ易い表現に変えているものの写像型の準言語である．

（5） 実例型： 対話型のデータ処理系で，特に表示端末装置を備えている場合には，より非手続的な照会回答の方式が考えられる．Query by Example[7] では，関係の名前を入力すると属性名を見出しとし空欄をもつ枠が表示される．その空欄の中に，表示すべきか否か，検索条件，他の関係との結合，などを記入する．表示すべき欄や結合のための欄には，何でもよいのであるが，たとえばこういう値が求まるであろう，という数や文字列を記入するので実例型と呼ぶ．

（6） 自然言語型： データベースは，現実世界の小局面についての情報を表示しているのであるから，それについての照会は日常的な言葉を用いるとしても，明確な深層構造をもつし，それほど多様な表現とはならない．自然言語は曖昧さをもつし，複雑な検索条件を表現しにくいこともあるが，他方において，最小の予備知識により照会を行えるという可能性をもっている．REQUEST[8], REDEZVOUS[9] などが報告

され，渋谷なども日本語もどきによる実験を行っている．

SEQUEL (structured English query language)[10],[11] は前述の写像型の準言語の一つであるが，5. で述べるシステム R の主要言語であるため，やや詳しく説明する．SEQUEL における照会の主要な形式は次の形をしている．

```
SELECT 〈attribute list〉
FROM 〈relation〉
WHERE 〈condition〉;
```

第2行で指定する関係の中の組で，第3行の指定する条件を満たすものの，第1行の指定する属性を取り出す．第3行はなくともよいし，第1行のパラメータを＊印とするとすべての属性を取り出す．〈condition〉は諸属性についての等式，不等式を含む論理式である．〈attribute list〉により取り出す属性が数値であれば，この前に SUM，COUNT，MAX などの演算子を書くことができる．

〈condition〉として

```
〈attribute〉 IN 〈set〉
```

という形を許す．〈set〉は〈attribute〉の取る値を羅列したものであるが，ここに SELECT 文を書くことも許す．これによって何段もの SELECT の階層が可能となる．ただし，これは結合演算でありながら結合する一方だけを上位に置き，その属性だけを表示する形となり対称性を欠いている．結合演算をそのまま表示するには，

```
SELECT X.A, Y.B
FROM 〈relation 1〉 X, 〈relation 2〉 Y
WHERE X.C=Y.C AND 〈condition〉;
```

のように，結合する関係にラベルを付け，それを用いて結合条件，条件，表示の属性を書き表わす．

次の2例は少し特殊な演算として，関係をある属性の値によって分割し (GROUP BY)，また分割の中からある条件を満たすものを選出する (HAVING) ものである．ただし第1例の F は $R2$ のキーとする．

```
SELECT X.A, Y.G, AVG (X.B)
FROM R1 X, R2 Y
WHERE X.A=Y.F GROUP BY Y.F;
SELECT A FROM R WHERE S>100
GROUP BY A HAVING COUNT(*)>10;
```

第2例で WHERE の条件は GROUP BY 以下より先行し，S>100 の組の数が 10 以上ある，A の値による分割，の結果を求めることになる．

SEQUEL は照会言語として出発したものであるが，データの操作にもその形式が利用できる．データの更新，削除，追加，新しい関係の定義などは次のようにできる．

```
UPDATE 〈relation〉
SET 〈attribute〉=〈expression〉
WHERE 〈condition〉;
DELETE 〈relation〉
WHERE 〈condition〉;
INSERT INTO 〈relation〉
SELECT ・・・ ;
〈relation〉 (〈attribute list〉) ←
SELECT ・・・ ;
```

さらに次節で，他の SEQUEL 文について述べる．

SEQUEL のような関係形式モデルのデータ準言語では，データのたどり方 (access path, navigation) については何も指定せず，その選択をシステムに委ねている．これはシステムの負担の増加であり，日常的な少数個の大規模な適用業務における効率を最高に保つことを第一目的としている場合には無駄な配慮かもしれない．しかし，最適化のアルゴリズムがうまく働いておれば損失は少ないであろうし，非日常的な問題解決のためには大きな力を発するし，多種の適用業務の処理を一様に最適化することができる．

5. システムR

関係形式モデルの実働化 (implementation) は既にいくつかの試みがなされているが，System R は実用的な規模で実用的な環境の下での性能の検討を目的として，IBM San Jose 基礎研究所で研究試作中のシステムである．

System R の主要な特徴は，前章で述べた SEQUEL を主要な言語とし，末端利用者および適用業務プログラマー両者の便宜を計り，データベース利用の全般的性能向上を目指すと共に，次の配慮がされていることである．適用業務のためにデータベースの一部をある見地から眺めることを許すため視点 (view) という，いわば副図式 (subschema) を導入することにより，データの高度の独立性を保っている．これはまた，データベース保全のための利用認可のためにも役立つ．データの統合性 (integrity) を保つために，データの意味から生じる，値についての不等式制約条件を指定しておけば，データ操作の際に検査が自動的に行われる．多数の利用者が同時に同一データに触れようとする場

合に生じ得る矛盾を回避し，並流性 (concurrency) を保つための施錠が自動的に行われる．また，階層形式モデル，網目形式モデルのシミュレーターを備えていて，これらのモデルに基づく応用プログラムも使用可能にしている．

System R は開発の便宜上仮想計算機 VM/370 (文献 13)を改造したものに基づいている．各使用者は，使用開始に当たって，応用プログラムを動かすための仮想計算機を稼働すると，全システムを管理するための仮想計算機がこれにたいしてそれぞれのデータベース仮想機を創設し，これが各使用者のためのデータベース管理を行う．これは処理装置が複数個のシステムでの効率を向上し，またデータベース仮想機がページ不在に出合ったときの影響を局限するためである．

各使用者から眺めると，システムは記憶の運用を中心とする下半分と，SEQUEL 文を処理する上半分とから成っている．本稿では上半分だけを概観する．

データの定義. データベースの概念的モデルを作ることは，System R では基底となる関係 (base relations) を創設する (CREATE TABLE) ことに他ならない．それは。関係の型をすべて，それぞれの属性の定義域，物理的尺度，単位，出力表現，を指定することにより定義することになる．

外面的モデルを作ることは，基底の諸関係についての SEQUEL 照会文の形により，新しい関係を定義することになる．これが視点の定義 (DEFINE VIEW) である．視点は，いわば論理的な関係である．応用プログラムは視点を通じた照会として記述することにより簡単になる．また基底の関係そのものを変更したとしても，実質的データが失われていなければ，DEFINE VIEW の中の SEQUEL 照会文を修正すれば，応用プログラムは全然変更せずにすむことになる．

内面的モデルを作ることは，多くのデータベース管理システムでは困難な仕事であるが，System R では次の 3 つの機構にまとめている．第 1 に，ある関係において 1 個または複数個の属性の値を指定する検索を早めるための，いわゆる 2 次索引を作り上げる，索引像の構成 (CREATE IMAGE) である．物理的には属性値の釣り合い樹木 (balanced-tree) 型の索引が作られる．第 2 に，照会回答演算の中でもっとも手間のかかる結合演算を早くするため，共通の属性間の対応を示す物理的指示子 (physical pointer) の 2 項関係である連結環を構成する (CREATE LINK)．共通属性は，少なくとも一方の関係ではキーでなければならず，連結環の構成では，それを指定しなければならない．第 3 に，データベース本来のデータおよび管理上のすべてのデータは，外部記憶を大きく，いくつかの区域 (segment) に分けた中に入れられる．この区域は物理的に接近しているため，密接な関係を同じ区域に入れることにより効率を上げられる．システムがデータのたどり方を最適化するときには，以上の 3 つの内部構造と，各関係内の組の数を考慮する．

概念モデル自体の変更には，一般にはデータをすべてダンプし，再編成するが，それは手間のかかる仕事である．System R で，関係に新しい属性を追加する (EXPAND TABLE) には，ただそのことを記録するに止め，新しい属性の値は空としておき，実際に値を入れる段階で徐々に物理的な再編成を行う．

データの操作. 主要な操作については前節で，SEQUEL 文の解説として述べた．さらに，汎用な親言語との接触面として，個々の組を参照するための滑子 (カーソル，cursor) が用意されている．滑子は各関係の中を滑り動くのであるが，論理的には，関係の部分集合と，「先頭の」組の属性値そのもの，を表わしている．使用者は名前をつけた複数個の滑子を関係の中で定義し (OPEN)，ある条件を満たす組に動かす (FETCH) ことができ，滑子の指す組を除去し，更新し，また追加することができる．滑子がもっている値を，SEQUEL 照会文で参照することができる一方，照会文の結果の値を滑子に付与することができる．

滑子のこのような柔軟な機能を利用することにより，階層形式モデルをシミュレートすることも可能となる．

データ利用認可. 各関係のデータの利用，追加，除去，更新(属性ごと)，定義取消し，属性追加，連結環や索引像の構成と取消し，および利用認可・取消しの権限，の認可を，ある人に，またすべての人に，与え，あるいは取り消す，ことによりデータの保全を行う．このとき，各関係を単位とするだけでなく，視点を単位とすることができる．そのため，各部門の社員の給与は見られないが部門の平均給与は見られる，とか，課長が自分の課の社員のデータだけを更新することができるなどの利用認可が可能となる．

データの統合化 データベースの統合性 (integrity) を保つために，属性の取るべき値についての制約条件を制定 (ASSERT) しておくと，システムは適当な時点で，その条件が満足されているかどうかを検査する．たとえば，ある関係が更新されたとき，更新後に関係

全体として満たすべき条件が破られていれば，システムはそれを指摘し，更新の処置全体を無効とする．

データベースの中に，他のデータから演算によって求まる量をあえて保存し，統合性の検査に用いることがある．このようなときには，関連データが操作されたときにいちいち更新しなければならない．更新のアルゴリズムは，どのようなデータを貯えているかに依存しているために自動化することができず，利用者が指定することになるが，関連データが操作されたときに更新アルゴリズムを駆動するために，PL/I における on 条件と同様の「引き金 (trigger)」が用意されている．

システム型録 (system catalogs) System R に貯えられているのは，以上で述べた次の「もの」である：基底関係，視点，索引像，連結環，制定，引き金．これらの内容はシステム型録に貯えられており，利用認可をもつ使用者は SEQUEL 照会文によって見ることができる．

本章の初めでも触れたように，多数使用者が同時にデータベースを利用するときには，相互の衝突による矛盾を防ぐために，区分など適当な単位で記憶に施錠し，同時利用を制限する．むやみに制限すれば特別の困難はないのであるが，効率を落とすことになり，したがって，矛盾を生じない範囲でできるだけゆるやかに，あるいは少々の矛盾は無視してまでゆるやかに施錠することが問題になる．System R での処理法は，システムの記憶割当てを主に扱う下半分の仕事であるので，これ以上は触れない．

6. あとがき

関係形式データベースとはデータベース・モデルの一つであると最初に述べた．データベースのモデル化についてもいろいろな見方がありえようが，広く認められているものでは，概念的 (conceptual)，外面的 (external)，内面的 (internal) の3面からとらえようとする．概念的とは，現実世界における情報システムの抽象としての把握，外面的とは各応用プログラムおよび末端使用者から見たもの，内面的とは計算機の記憶割当て，データのたどり方，から見た把握である．

関係形式データベースは何よりもまず外面的なモデルであり，利用者の見地よりデータベースを使用し易くしようとするものである．その意味で，関係形式モデルを実働化の方法とは無関係に論ずることもできる．外面モデルとして，階層形式や網目形式と比較すると，データ構造の表現力という点で，3者に実用上の差は少ないものの，関係形式モデルは共通の属性に関する結合により，諸関係間を結び付けられる点で他の2モデルよりも一般的であり，また諸属性に関して対称的であるために扱い易い．

しかしながら関係形式データベースの特徴はそこだけに限られず，データベースにまつわる他の諸問題を探求するにも有効な枠組みのようである．単に外面モデルの一つとしてではなく，概念的モデル，内面的モデルをも含めて統括的な設計を行い得る接近法として有用である．表題を曖昧な言葉にした意図もこの点にある．

現実世界での情報システムの要求は多様であり，そこで使用される道具も多様である．関係形式データベースといっても諸種の変形が生じ得る．たとえばBARTH（参考文献 5)）は比較的小規模の関係をなるべく高速に処理することを念頭においている．

EXIR（参考文献 14)）は単一の次数の高い関係についての照会を目的としているが，組ごとの操作ではなく逆に属性ごとの操作を基準としている．

データベースの発展と普及は，もちろん磁気ディスクの性能価格比の向上に負っている．新しい大規模記憶装置の出現はそれを促進するとともに，ある段階では大きな変革を迫るであろう．(文献 15))ではそのさきがけが紹介されている．関係形式データベースは，このような変革にも処し易い，有力な接近法と予想される．

なお関係形式モデルに関する詳しい解説は（文献 16))を，諸文献については 17) を見られたい．

参考文献

1) E.F. Codd: A relational model of data for large shared data banks, Comm. ACM, Vol. 13, No. 6. pp. 377～387. June 1970.
2) E.F. Codd: Recent investigations in relational data base systems, Information Processing '74 (Proc. IFIP Congress, Stockholm, 1974), North-Holland.
3) E.F. Codd: Further normalization of the data base relational model, Courant Computer Science Symposia **6**, "Data Base Systems", pp. 33～64, Prentice-Hall New York City, 1971.
4) 穂鷹良介，第3正規形について，情報処理，投稿中.
5) 宇野栄，宇土正浩：対話型アプリケーションにおけるリレーショナル・モデルの応用例，情報処理，本特集号. pp. 978～985.

6) S. J. P. Todd: Peterlee Relational Test Vehicle PRTV, A technical over-view, IBM Scientific Centre Report UKSC 0075, July 1975.
7) M. M. Zloof: Query by Example, Proc. National Comp. Conf., Anaheim, 1975.
8) S.R. Petrick: Semantic interpretation in the REQUEST System, IBM Research Report RC 4457, IBM Research Center, Yorktown Heights, 1973.
9) E.F. Codd: Seven Steps to Rendezvous with the casual user, Proc. IFIP TC-2 Working Conf. on Data Base Management Systems, Cargese, Corsica, North-Holland April 1974.
10) D. D. Chamberlin, R. F. Boyce: SEQUEL: A structured English query language, Proc. ACM-SIGMOD Workshop on Data Description, Access, and Control, Ann. Arbur, Mich., 1974,
11) 渋谷政昭, 鷹尾洋一, SEQUEL について, 情報処理学会, データベース研究会資料 22, 1975年6月.
12) M. M. Astrahan et al.: System R: A relational approach to data base management, to appear in TODS.
13) IBM Corp., Introduction to VM/370, IBM Publication, No. GC 20-1800.
14) L. Abbott (ed.): EXIR, A computer system for information storage and retrieval, User's Manual, Taximetric Laboratory, University of Colorado, Boulder, Colorado.
15) 植村俊亮: CODASYL 方式のデータベース・システム, 情報処理, 本特集号. pp. 892～903.
16) C.J. Date: An Introduction to Database Systems, The Systems Programming Series, Addison-Wesley, 1975.
17) E. F. Codd: Relational Data Base Management, A Bibliography, IBM Research Lab., San Jose, Calif., Aug., 1975.

(昭和51年6月14日受付)