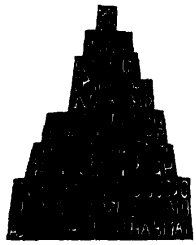

25. データベース用言語†

植　村　俊　亮††

1. はじめに

データベースシステム研究開発の隆盛の中で，データベース用言語という分野はいがいに活気にとぼしい．「だれにでも使いやすい」データベースシステムの完成が重要であることは衆目の一致するところであり，データベースシステムと利用者とのインタフェースがデータベース用言語にほかならないことを考えると，これは奇妙な現象といえる．データモデルやデータベース設計の研究はさかんでも，それを具体的に利用者の手に引き渡すプログラム言語にまでは，なかなか手がまわらないのが実情であろうか．

本稿では，まずこの分野の古典ともいうべき CODASYL のデータベース用共通言語体系の現状を報告し，さらに関係モデルにもとづく代表的なデータベース用言語を簡単に取り上げる．最後に今後の方向を考察する．

なお本稿でいうデータベース用言語とは，データベース管理者がデータベースを記述したり，利用者がデータベースを利用したりするときに使う諸言語をいう．このほかに，データベースシステムそのものを作成するためのシステム記述言語が考えられるが，このための新しい言語はとくにみられないので，本稿では扱わない．

2. CODASYL 方式のデータベース用言語

2.1 概　要

CODASYL が長年にわたって開発し保守してきたデータベース用共通言語体系は，さかんなデータモデル論争の渦中で毀誉褒貶相半ばしつつ着実に普及しており，わが国ではミニコン，オフコン上にも処理系が作成されつつある．アメリカでは標準化作業が進行しており，これも近い将来わが国に影響を及ぼすと考えられる．

この方式は COBOL や FORTRAN を親言語としているので，原則として利用者はプログラマである．実動化にあたっては，各システムで，プログラマでない（末端）利用者のための問合せ機能を付加することがおおい．残念なことにこの部分は方言にとどまってしまう．CODASYL 自身でも末端利用者のための問合せ機能開発を行っているが，具体的な言語仕様になって現われていない．

2.2 古典的な体系の変遷

CODASYL 方式の言語仕様開発は十数年以前に開始されたので（現在でも，十年前に言語仕様案をまとめたデータベース作業委員会の名前をそのまま踏襲して，DBTG 仕様といった用語を使う資料を散見する），いわばデータベース用言語の古典をここに見ることができる．

実世界のデータを計算機によるデータベースに写像するためには，まずそのデータの名前，特性，相互の関連を整理して，計算機処理可能な形式で記述しておかなければならない（4. 参照）．このようなデータベースの記述をスキーマといい，スキーマを記述するプログラム言語をスキーマ DDL (Schema Data Description Language) あるいはたんに DDL という．DDL はデータを記述する記述項だけからなり，通常のプログラム言語における文に相当するものはない．DDL で記述したスキーマは，いわば COBOL のデータ部に近いが，データ部はその COBOL プログラムの一部であるのに対して，スキーマは共用されるデータベース全体の記述であって，応用，応用プログラム，プログラム言語から独立している．個々の利用者ではなくて，データベース管理者が DDL を使う．

CODASYL の DDL 委員会が DDL の言語仕様を保守している．最初に公刊された DDL 仕様書[1]以後の，言語仕様のおもな変更を次にあげる[2]．

（1） 記憶構造に関する記述項はすべて DDL から削除し，DSDL (Data Storage Description Language) という別の言語にまとめた．DSDL による記憶構造の

† Database Languages by Syunsuke UEMURA (Computer Science Division, Electrotechnical Laboratory).

†† 電子技術総合研究所ソフトウェア部

記述を記憶スキーマという．

（2） 呼出し制御（機密保護）機能は DDL とは独立した機能とみなして，言語仕様から削除した．すなわち ACCESS-CONTROL 句がなくなった．

（3） 言語仕様を全体にわたって整理した．AREA 句も削除した．

こうした仕様変更は，ここ十年ほどのデータベースシステム研究の成果を反映したものである．とくにアメリカ規格協会のデータベース制御システム研究班報告書[3]の影響を受けている．この研究班の用語でいいかえると，スキーマは概念スキーマ，記憶スキーマは内部スキーマである．

データベース全体の記述であるスキーマに対して，個々の応用の立場からみたデータベースの一部分の記述を副スキーマ (Subschema)，応用プログラム中でデータベースを操作する文をデータ操作文という．この二つは応用プログラムを書く親言語 (COBOL か FORTRAN) の言語仕様中に埋め込まれている[2),4)]．

副スキーマの考え方は，関係モデル研究者のいう視野 (view) の概念に引き継がれていった．CODASYL 方式の副スキーマは，スキーマの一部分の記述であって，スキーマ中のデータ構造の一部をほぼそのまま記述することが原則である．これはきわめて現実的な仕様といえる．

データ操作文も従来の1時1レコード型（レコードを一つずつ操作するやり方．集合演算でない）の仕様から基本的にかわっていない．ただ関係モデル的な考え方の影響はここにも現われていて，たとえばデータベース中のレコードを，そのレコードの任意のデータ項目の値を指定して，呼び出すことができるようになった．1971 年にこの言語仕様案が最初にまとまった時点では，この種の機能はなかった．のちにレコードキーと指定した項目についてのみ呼び出せることになっていたが，現在では副スキーマからレコードキーの概念も消えてしまって，任意のデータ項目の値でレコードを呼び出すことができる．利用者にとっては使いやすくなったが，システムの負担がふえた．

CODASYL の COBOL 委員会が，COBOL 文法の一部として，副スキーマ機能およびデータ操作文の言語仕様を担当している．副スキーマ機能をよく検討してみると，むしろ DDL に近いので，これは COBOL ではなくて DDL 委員会で扱うべきだとの議論が続いている．このほか，さきに DDL から削除された AREA の概念にしても，COBOL 委員会の側では，対応する REALM の概念を残す（AREA と REALM との写像関係はなくする）といった処置をとっており，両委員会の間にくいちがいがみられる．

さらに，アメリカ規格協会は CODASYL とは別の，アメリカ国内の標準化を担当する組織であるが，ここでも COBOL の標準化と DDL の標準化とを別の委員会が担当しており，合計四つの委員会が互いの連絡調整に苦労している．

3. 関係モデルとデータベース用言語

関係モデルによるデータベースシステムも長い研究期間をへて，ついに IBM 社の QBE[5),6)]，SQL/DS[8)]，DEC 社の INGRES[9)] といった商品の段階に達しつつある．いずれも現段階では比較的小規模なシステムを指向しているが，今後利用者の反応もみつつ，計算機の性能向上と歩調を合わせて，徐々に大規模データベースに適用範囲を広げていくと考えられる．

商品化された三つの言語を直観的に比較する．

QBE (Query-by-Example) は関係モデルによるデータベース用言語の中でも，きわ立って斬新である[5),6)]．事務処理の分野で古典的な存在である RPG (Report Program Generator) の伝統と，データの関係モデルとを結び合わせて，コーディング用紙上ではなくて画像端末上に巧妙に問合せ言語を実現した．通常の問合せでは，利用者が打鍵入力しなければならない字数はごくすくなく，使いやすく，わかりやすい．自然言語によらずに使いやすい言語をめざす方向の一つの頂点といっても過言ではない．

次の二つの関係を例として考える．

HYOODAI (BANGOO, DAIMEI)

KEY (BANGOO, KEYWORD)

「"DATABASE" をキーワードとする文献を求めよ」という問合せは，画像端末上に次のように打鍵する．

KEY	BANGOO	KEYWORD
	9	DATABASE

HYOODAI	BANGOO	DAIMEI
	9	P.

利用者は，表示された表のしかるべき位置に 9, DATABASE, 9, P. を打鍵するだけである．9 (実際のシステムでは "_9" と打つ) のように下線のある要素を例要素という．これだけで，「キーワードが

"DATABASE" である文献の番号をたとえば 9 とするとき，その文献の題名を印字せよ (P.)」という意味になる．

APAD[7] では，正規化されてない表に対する QBE 型の問合せ言語が提案されている．

SQL/DS (Structured Query Language/Data System) は，かつての SEQUEL, SEQUEL II の商用版であり，SEQUEL II の実動システムが System R であったことから，System R そのものあるいはその部分仕様にもとづくと考えられる[8]．SQL は関係モデルによる言語族の中では，写像型という奇妙な型に分類されていることからもわかるように，利用者の使い勝手に現実的な配慮をした言語である．さきの問合せを SQL 流に表現すると，次のようになる．

```
SELECT  DAIMEI
  FROM  HYOODAI
  WHERE  BANGOO=
  (SELECT  BANGOO
     FROM   KEY
     WHERE KEYWORD='DATABASE')
```

QUEL (QUEry Language) あるいはそれをC言語に埋め込んだ形式の EQUEL (Extended QUEL) は，カリフォルニア大学バークレイ校で，関係モデルデータベースシステム INGRES (Interactive Graphics and Retrieval System) のために開発された[9]．この言語はコッドの Alpha にかなり忠実で，SQL よりも関係論理に近い．利用者はどんな単純な問合せでも，変域を指定した組変数を宣言して使う．

```
RANGE  OF  K  IS  KEY
RANGE  OF  H  IS  HYOODAI
RETRIEVE  INTO  W
(DAIMEI)
WHERE  K. BANGOO=H. BANGOO
   AND  K. KEYWORD="DATABASE"
```

どの言語にしても，関係モデルの集合演算を基礎としており，さらに各言語は，単なるデータの検索，更新にとどまらず，外部スキーマ（視野，view），データ制御（呼出し制御，同時実行制御）まで言語仕様に含めてしまう傾向にある．実動化がどこまで進むか興味深い．

その一方で，SQL は PL/I, EQUEL はCという親言語を想定しており，また QBE, SQL は IMS の DL/I を補完する機能を果たす．これらの関係言語は，現在すくなくとも次の三つの問題点を残している．

（1） 親言語の1時1レコード処理と，関係言語の集合演算との間に違和感がある．

（2） 報告書作成といった典型的な事務データ処理を統一的に議論できない．

（3） データベース管理者機能が明確でない．

4. 今後の方向について

今後の方向について，いくつかの項目を列挙する．

（1） 統合の試み

当分の間は，各種のデータベース用言語が共存していくものと考えられる．UDL (Unified Database Language)[10),11)] は，CODASYL 方式や IMS の1時1レコード処理から，関係モデルの集合単位のデータ処理まで，一つのデータ準言語で統一的に扱おうとする試みである（UDL 自身はデータベース機能だけをもち，完結した言語ではない．その意味でデータ準言語という）．かつて FORTRAN と COBOL とから PL/I が産まれたことを想起させる．

（2） データモデルと概念スキーマ

実世界のデータを概念スキーマに写像するためのデータモデル研究はさかんであるが，具体的に概念スキーマ記述言語を提示するものはほとんどない．むしろデータモデルの世界でいう概念スキーマは，計算機にそのままのせるプログラムのようなものではなくて，いわば実世界のデータを抽象整理したものにとどまるとする方向もある．概念スキーマに記述したデータをすべて計算機処理しなければ意味がないということではないが，すくなくとも概念スキーマそのものは，計算機でも処理可能なていどの厳密さをもって記述されているべきであると，筆者は考える．概念スキーマにも，隔層があるのかもしれない．いずれにせよ読む人によっていろいろに解釈できる概念スキーマでは，作った意味がない．一方で，概念スキーマを担当するのは，その組織体のデータの専門家であるが，計算機の専門家とはかぎらない．したがって概念スキーマ記述用言語は使いやすいものでなければならない．

（3） データ辞書とデータ記述言語

ANSI/X 3/SPARC 報告書の内容のうちで，CODASYL 方式のデータベース用言語が吸収しきれなかった概念は，データ辞書 (data dictionary) である．データの名前，属性，相互の関連など，データのデータ（メタデータともいう）をまとめたものをデータ辞書という．計算機側のディレクトリに対応した人間側の記述がデータ辞書である．ANSI/X 3/SPARC 報告

書では，3層のスキーマがこのデータ辞書を介して結合している．このような立場から，データ辞書に情報を書き込むことが，すなわちスキーマの記述であるとする考え方がある[12]．こう考えると，データ記述言語もデータ記憶記述言語も必要でなくなり，データ辞書に対するデータ操作文がデータ記述を行うことになる．実際データは人間の行為の所産にほかならないのであって，データを記述することもまたデータ操作とみなしうる．問合せ言語で外部スキーマを記述しようとする SQL のような方向もある．データベースにデータを書き出すデータ操作文と，データ辞書にメタデータを書き出すデータ操作文とは同じであるべきか，それぞれに別に用意すべきか．さらに断片的に提示されたメタデータをもとに，データベースを構築する手順はどうあるべきか．興味深い研究課題である．

（4） 要求仕様とデータベース

データがデータベースに含まれているということは，これから役に立つことを前提としている．役に立つとは，使用したいという要求があることである．したがってデータの要求仕様をまとめると，概念スキーマができあがると考えることもできる[13]．要求は個々の利用者が出すものであるから，要求仕様からは外部スキーマができて，それを集約すると概念スキーマになる．もちろん現在時点での要求仕様を近視眼的に集めただけでは，実際に存在する有益なデータを見失なう恐れがある．さらにデータベースが寿命を長く保つためには，将来変化していくであろう利用者要求を柔軟に吸収できるものでなければならない．

ファイル設計の古典的な手法の一つは，そのファイルから必要とされる出力をまず整理して，目的の出力をえられるファイルを設計するというものであった．要求仕様によるスキーマ設計はこれに近い．利用者のデータベースに対する問合せも，高水準になるほど要求仕様そのものに近づくであろう[13),14]．ソフトウェア工学における要求仕様技術をデータベース用言語に応用する研究は，まだ端緒についたばかりである．

本稿の執筆にあたって，有益な御示唆をいただいた椿正明氏に深く感謝いたします．

参 考 文 献

1) CODASYL データベース用データ記述言語1973年6月版，p. 206，情報処理学会 (1977).

2) CODASYL 方式のデータベースシステムに関する基礎資料は，すべてカナダ政府から出版されており，下記に送金すると手軽に入手できる．
Supply and Service Canada,
Materiel Data Management Branch,
4 B 1 Place du Portage, Phase 3,
11 Laurier Street,
Hull, Quebec, CANADA KIA0S5
価格などをまとめて次に示す．
CODASYL Data Description Language Committee Journal of Development, 1978 ($6.50).
CODASYL FORTRAN Data Base Facility Journal of Development, 1980 ($5).
CODASYL COBOL Committee Journal of Development, 1978 ($10. Page change update service は年間 $15).
なお DDL 委員会 JOD, COBOL 委員会 JOD ともに，1981年版が刊行される予定である．

3) The ANSI/X 3/SPARC DBMS Framework, Report of the Study Group on Database Management Systems, Information Systems, Vol. 3, No. 1, pp. 173-191 (1978).

4) 植村俊亮：データベースシステムの基礎，p. 237，オーム社 (1979).

5) Zloof, M. M.: Query by Example, Proc. AFIPS NCC, Vol. 44, pp. 431-438 (1975).

6) Zloof, M. M.: Query-by-Example: a Data Base Language, IBM Syst. J., No. 4, pp. 324-343 (1977).

7) 北川，国井：APAD-オフィス自動化のためのデータベースシステム，ソフトウェア・プロダクト工学，bit 臨時増刊，pp. 224-237 (1981).

8) Astrahan, M. M., et al.: System R: Relational Approach to Database Management, ACM TODS, Vol. 1, No. 2, pp. 97-137 (1976).

9) Stonebraker, M., et al.: The Design and Implementation of INGRES, ACM TODS, Vol. 1, No. 3, pp. 189-222 (1976).

10) Date, C. J.: An Introduction to the Unified Database Language (UDL), Proc. 6th Int. Conf. on VLDB, pp. 15-32 (1980).

11) Date, C. J.: An Architecture for High Level Language Database Extensions. (Unified Database Language-UDL) PL/I Version, IBM TR 03.099 (1980).

12) 椿 正明氏の私信による．

13) 穂鷹良介：データベースの論理設計，p. 108，情報処理叢書 6，情報処理学会 (1981).

14) 米澤明憲：プログラム合成の一手法，同上 3 F-4 (1981).

(昭和56年4月22日受付)