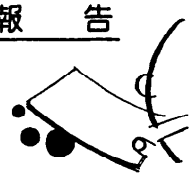
報　告



パネル討論会 (2)

データベース・システム理論の研究開発動向と今後の課題

昭和54年度第20回全国大会† 報告

パネリスト

上園　忠弘[1]，植村　俊亮[2]，上林　弥彦[3]

高平　　叡[4]，古川　康一[5]，司会　大須賀節雄[6]

パネル討論会にあたって

大須賀節雄

データベース・システム技術は大量のデータを処理するという現実的な要求に応える技術として 1960 年代に生まれた．1960 年代は計算機技術全般にわたる発展期であり，各種の応用システムが開発されたが，データベース技術は当時はこれら応用システムのサポート・システムかあるいはたかだか応用技術の一つという程度の位置づけであったと考えられる．その後十数年を経て，これら各種の応用はあるものは完成された技術として定着し，またあるものは忘れられていったが，いずれにしろ当時の応用技術の多くは理論面の研究という点では下火になっている．

これにたいし，データベースに関しては理論研究が一層増大しているように見える．これはデータベースが一応用技術というより，情報処理の基本に関わる何かを含んでいるからであろう．

理論研究を促がす最初のきっかけを作ったのは CODASYL DBTG が新しいモデル（ネットワーク・モデル）を提案したこと，続いて E. F. Codd が関係モデルを提案したことであろう，これ以後，多種類のデータ・モデルが提案されてきた．この中で上記二つに加え，IBM 社の IMS で代表される（というより事実上唯一の）階層モデルの三種類が，代表的なものとして論じられている．

データ・モデルの概念はデータベース技術の基本であるデータ独立の概念と不可分である．従来，情報処理において主たるものはプログラムであり，データはプログラムごとに定義される，いわば従の位置におかれていた，このような形態はデータの集中管理というデータベース本来の目的と一致しない．そこでプログラムからのデータの分離あるいはデータからの応用プログラムの独立が必要とされた．これを達成するべくシステム側でデータの構造，蓄積およびアクセスの方式を応用プログラムとは独立に定め，各応用プログラムとデータベースの間にデータベース管理システムが介入してこの両者を結び付けるというデータベース・システムの基本的なアーキテクチャが確立されていった．実際のデータベースへのアクセスはデータベース管理システムが行うにしても，各応用プログラムは何らかの方法で欲するデータを指定しなくてはならない．データベースの記憶は記憶装置の物理的構造に依存しているから，これをそのまま応用プログラムに見せるのは，データ独立の思想に反し，工合が悪い，応用プログラムが必要なのはデータそのもの，あるいは他のデータとの論理的な関係であって，データが物理的にどのように配置されているかには無関係だからである．そこでデータの論理的な関係のみを表わす構造表現を実際の物理的データベースと応用プログラムの間に置き，応用プログラムはこの論理的な構造に基づいてデータを要求する．データベース管理システムはこの要求を受け取って，それが指定する論理構造部分に対応する実在の（物理的な）構造に変換して，記憶装置にアクセス要求を出す．このように，データの論理構造に相当する部分はいわば応用プログラムに見えるデータベースであり，データ・モデルとかスキーマ

† 日時　昭和54年7月25日，12：30～14：15
場所　日本大学工学部

1) 日本アイ・ビー・エム(株)
2) 電子技術総合研究所
3) 京都大学
4) 日本電信電話公社横須賀電気通信研究所
5) 電子技術総合研究所
6) 東京大学宇宙航空研究所

本パネル討論は，大阪大学　田中幸吉教授の企画になり，同教授の司会のもとに運営される予定であったが，田中教授御病気の為，大須賀節雄助教授が代りに司会をつとめた．

と呼ばれる．

データベースはこのようにそれが用いられる現実の世界の一つのモデル表現であると考えることができるが，この解釈のもとではスキーマはその世界内の論理構造を表わしている．データの論理的関係は当然データベースが用いられる世界により異なる．たとえば企業 A と企業 B では組織もデータ処理の方式も異なるからデータベースの論理構造は異なるのが普通である．しかし論理関係の表現法は一定の範囲内で，異なった世界たとえば企業，官庁，学校，銀行，病院などにわたって共通のものが多い．したがって，基本の情報構造の型を定め，この型の範囲内で任意の具体的構造を定義する手段を用意しておけば，これを用いて多様な世界の論理構造すなわちスキーマを記述し，定義することができる．汎用データベース・システムはこのようなシステムであるが，基本の情報構造の型をどのように定めるかがこのような汎用データベース・システム設計の最も重要な決定であり，前に挙げた代表的なデータ・モデルはこの型として網状構造（ネットワーク・モデル），表形式（関係モデル）および木構造（階層モデル）を選んだものといえる．

これらの基本型には当然個有の表現能力がある．というより表現能力の限界がある．表現能力は大であることが論理的には望ましいが，実用上のシステムとしては処理の効率も考慮せねばならない．実用面から見たデータ・モデルの評価は，データベース・システムが利用される対象に応じて異なるはずである．このような評価の問題は実用データベース・システムにとって重要な課題であり，この方面の研究も盛んになってきている．またデータを集中管理するというその性質から，特に，データベースの保守の問題が重視され，データの完全性や安全性の問題，多数ユーザのもとでの相互干渉の問題も生ずる．さらに今後の問題としてデータベース・マシンや分散型データベースなどに関心が寄せられている．

一方，これら実用化に則した研究課題とは別にデータベースの理論研究を通して，情報処理の基本問題である情報のセマンティクスを解明しようとする傾向も顕著である．データがプログラムの付属物である時は，データにはプログラムの目的に応じた意味が与えられたが，データとプログラムが独立した時，もはやプログラムとの関連においてデータを意味づけることはできず，データそのもののもつ意味を把握することが必要になる．データベースを実世界のモデルとする考え方は実世界に存在する諸関係との対応によってデータの意味を定義しようとすることに他ならない．ここで前述の構造の型の表現能力が問われて来る．実世界に存在する関係を十分に表現できないなら，データベースは実世界の忠実なモデルと言えないからである．

現在実用化されているデータ・モデルは必ずしも十分な表現能力を備えているとは言い難い．処理対象が今後共，前述のような組織内で要求されるデータに限定されるならばデータのセマンティクスに関するこれ以上深い考察は不必要かも知れない．しかしデータベースが実用化されたことにより，データベースを当初予想された範囲を越えて応用しようとする試みがなされ，これがより一層表現力の豊かなデータ・モデルや新しいデータベース・アーキテクチャへの要求となって表われている．データベース理論研究のかなりの部分がこの方向を目ざしているといってもよいだろう．

研究対象としてのデータベースを論ずる時，この二つの立場，すなわち現状の程度のデータ・モデルという枠組を設定し，この範囲内でより優れた実用システムを追求する立場と，この枠組自体を拡張し，そのためには現在の技術にこだわらない立場とがあることを認識しておくことが必要であり，このどちらにも偏ることなく，現在から将来への流れとしてこの両者を位置づけることが出来れば実りある議論が行えるものと思う．

データベースとシステム監査

上園 忠弘

はじめに

データベースの有用性を支える要件として CARP なる要素をシステムに与える必要がある．(図-1)

これらの要素のうち C, R, P については比較的よく論じられているが，Auditable という要素については逆に比較的論じられることが少ない．

しかしながら，近時システム監査の重要性が叫ばれている折柄，データベース・システムとそのオーディ

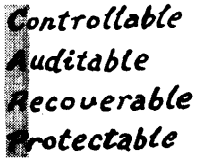


図-1 システムの質を保つ要件

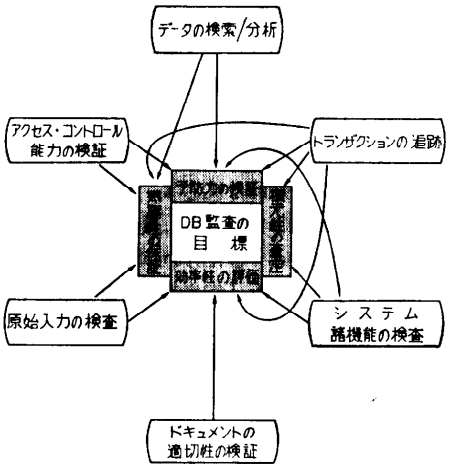


図-2　データベース監査の視点

ットの問題は，必然的に対（つい）にして論じられなければならない．この場合の幾つかの問題点とその解決法について考える．

データベース監査の視点と問題点

システム監査人がデータベースをどの角度から見ようとするかは，システムの環境によって多少の変化はあろうが，その目標とする所は，図-2 の中央におかれた四つの項目にあると考えることができる．一方，これらの目標を達成するために，システム監査人がおこなう活動は，図-2 の外側におかれた6項目に要約することができる．逆に監査人は上記のような6項目の活動をおこなうための手段または道具を，彼等自身の手中に収めていたいと望んでいると考えて良い．

では，現在データベース・システムは上記のような道具をシステム監査人に与えているか，すなわち Auditable であるか，必ずしもそうではない．問題点は次の通りである．

① データベースの複雑性

データベースの内部構造の複雑さは，データ処理の専門家とは限らないシステム監査人にとって，時に近より難い壁と映る．克服には時間が必要となる．

② データベースの統合的性格

データベース・システムにおいては，一つのトランザクションが他の多くのトランザクションをシステム内で派生させ，芋ずる式の更新が行われ，しかも中間結果が残らない．そうなると監査人が拠り所とすべき証跡が消失するという問題が生じる．

③ データベースの唯一独自性

データベースが整備されて行くほど，企業体におけるデータはデータベース以外からは得られなくなる．

それだけにデータベースの信頼性の保証が必須のこととなる．システム監査人はそれを確認しようとして，データベースにアプローチするのであるが，只一つしかないものを傍証で確認することはできない．つまりデータベースの信頼性を外側から検証する方法がなくなって，データベースそのものを見る方法しか残されなくなる．

④ データベースへの直接的アクセスの困難さ

データベースにおけるデータの独立性は，プログラムとデータとの間に，いわゆる DBMS が介在することによって得られることは周知の通りである．しかし，監査人にとってはそれが独立にまた直接にデータに近づくことをさまたげるという難点となる．データベースの監査を行うために DBMS の厄介になるというのは，自己矛盾を生じさせることになるのである．

問題点の解決

さて，今までに述べた監査人の要求事項と幾つかの問題点との間にバランス点を見つけるための方法を考えてみることにする．

① 監査用ソフトウェアと DBMS のインタフェース

すでに多数発表されている汎用の監査用ソフトウェアは，大部分がデータベース時代以前に開発されたものであるため，ファイルを順次的に扱うものでしかない．そこで，これと DBMS との間にインタフェースを作る案が出てくる．しかし，これもデータベースの静態的なテストに止まり，データベース内での動的な変化をトレースするには難点がある．

② 監査用ソフトウェアの再開発

DBMS に依存しないでもデータベースに直接アクセスできるような監査用ソフトウェアの開発が考えられる．これは監査人の独立性を保つためには望ましいことであるが，開発のコストと労力およびどの DBMS に対しても汎用性を保つことの難しさが指摘される．

③ データベースの再編成用プログラムの開発

データベースを一旦順次ファイルにダンプし，次いで在来の監査用ソフトウェアの処理に適した形式に再編成する方法は良くとられる所である．しかし，この方法も ① のケースと同じ欠点を持たざるを得ない．

④ DBMS ユーティリティの利用

汎用監査用ソフトウェアに依存しない方法として，

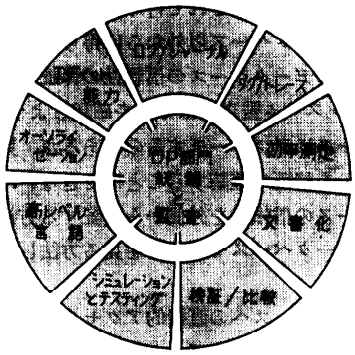


図-3 監査人が関心を持つ各種機能／能力

DBMS が提供するユーティリティの活用が考えられる．例えば，テストのためのユーティリティを監査に流用したり，ログの機能を活用するなどである．これは DBMS に依存するという難点はあるが利用価値はかなりある．

⑤ DBMS にビルトインされた監査機能

もし設計時点で監査の能力(Auditability) が考慮された DBMS が存在するなら，問題の大部分は解決する．これは，例えば DBMS 内に組込まれた機能によって，監査人が指定するデータないしトランザクションを，その処理の中で自動的に監査用ファイルに記録し，監査人は自分が望む時に自由にそのファイルにアクセスできるような能力の存在を意味する．次代の DBMS 設計においては，システムのクオリティを保つためにも，Auditability は忘れてはならない要素の一つと言うことができる．

その際，**図-3** のような各種機能を監査人が独立に活用できるように設計されることが望ましい．

データベース・システム理論が CODASYL 方式に及ぼした影響

植村 俊亮

70 年代のデータベース・システム理論が現場にどのような影響を与えたかを，CODASYL 方式を例にとって考えてみたい．CODASYL プログラム言語委員会のデータベース作業班 (DBTG) がいわゆる DBTG '71 報告書を最終的にまとめつつあった時期に，作業班のある委員 (R. Engles 氏) が次のような提案をしたことがあった．

「プログラム作成の単純化，データの一貫性確保，プログラムとデータとの間の独立性を高度に保証するデータ操作言語をもつことは，計算機社会の最大の関心事である．よって DBTG 仕様を次の方向に変更すべきである．『領域，データベースキー，配置モード，現在指示子，親子集合選択となんらかかわりなしに，応用プログラムを書くことができる．』」

わたくしは，たまたまこのときの DBTG に出席していたが，委員の一般的な反応は，提案の趣旨，精神は正しいものと認識しながらも，具体的に言語仕様にこれを取り込むにはどうしたらいいか見当がつかないといったところであった．「報告書をまとめるまでに時間がかかりすぎている．このへんで一区切りとすべきである．」という現実論が強く，また「データベースの世界では，システムの性能が重要な課題である．さきの項目はいずれも性能向上のための道具立てであって，なしですむにこしたことはないが，抽象論ではなくて，どうすればよいかという技術を示せ．」という反論もあった．結局この提案は圧倒的大差で否決された．

CODASYL は DBTG '71 報告を基本線としてデータベース用共通言語体系の仕様を定めてからも，COBOL の場合と同様に，その改訂作業を熱心に継続してきた．アメリカ規格協会(ANSI) でも，この標準化作業が開始されたので，1978 年に CODASYL は，それまでの改訂をまとめた新しい言語仕様書を公刊した．これが，

COBOL 委員会開発報告書 1978 年

DDL 委員会開発報告書 1978 年

で，いずれもカナダ政府から出版されていて自由に購入できる．両書を検討すると，1970 年代のデータベースシステム研究の進展を反映して，言語仕様が格段に進歩し，さきの Engles 氏提案をかなり取り入れた形になっていることがわかる．おもな改訂内容を以下に列挙する．

① スキーマを記述する DDL から，配置モード句などいくつかの句を削除し，新たに記憶スキーマ記述用の言語 DSDL を設定した．これは CODASYL 方式のスキーマ構成を ANSI/X3/SPARC データベース研究班の提唱する 3 層のスキーマ構成に近づけようとする努力であり，CODASYL 方式でいうスキーマがかなり抽象されて，いわゆる概念スキーマに近くなってきたことを意味している．なお DSDL (Data Storage Description Language) の仕様は，DDL 委員会開発報告書の付録として案が示されている段階である．

② 再帰的親子集合を認めた．かつては，親子集合

型の親レコード型と子レコード型とが同じであってはならないという制約があり，議論のまとになっていたが，この制限は徹廃された．また従来の CODASYL 方式の親子集合では，親と子とに共通する情報はすべて親に1回だけ代表して記録し，情報を共有する子をつなぎでつなぐという伝統的な考え方があった．こうした情報搬送構造 information bearing structure は関係モデルの立場から激しい批難をあびたが，CODASYL 方式でも，親と子とに共通する情報は親にも子にも繰り返して記録するという考え方を次第に認めるようになって，この線にそった言語仕様改訂が行われた．STRUCTURAL CONSTRAINT（構造制約）句はその例である．

③ 従来からのデータベースキーの概念を変更し，これをシステムのための道具と規定して，利用者の目にふれさせないことにした．新たに，レコード中の実際のデータ値にもとづくレコードキーの概念を導入した．

④ 同時実行の制御機能を全面的に改訂した．従来の監視モード，拡張監視モードといった考え方は削除され，これまでの領域の専有，共有宣言のほかに，レコードの施錠を暗に陽に行うことになった．解錠のための命令まで用意された．これらを利用者にゆだねることは，近年のデータベース・システム研究における風潮であり，CODASYL 方式もまたその影響を受けているわけであるが，長い目でみて正しい方向かどうか疑問である．

こうして改訂を検討してみると，DBTG '71 報告書当時にくらべて，データベース・システム研究がずい分進歩し，これを積極的に取り入れてきた CODASYL 方式の共通言語仕様も格段によくなったという印象を受ける．

なお，関係モデル実験システム System R について，一言ふれたい．1979年5月の IEEE COMPUTER 誌に，System R の総まとめ的な論文が掲載されている．その中でわたくしがとくに印象に残ったことは，SQL の性能が他のデータベース・システムのそれに匹敵できる水準に達したということと，会計や在庫管理などの事務応用では，たいていの利用者は親言語を介して System R を使うということであった．関係モデルが事務応用全体をカバーしていないという意味で，後者はとくに興味深い観察であった．

データベース理論の発展とその問題

上林　弥彦

ここ数年間データベースの理論面での発展にはめざましいものがある．それに伴って，1975～76 年にかけて，データベース関係の論文のみを扱う雑誌(ACM Transaction on Database Systems, Information Systems) や国際会議 (ACM SIGMOD International Conference, Very Large Data Bases) もスタートした．これらの国際会議には，実際家と理論家が共に参加して議論しており，計算機科学の他の分野より両者の隔たりは少ないように感じられる．

このような，データベース関係の最近の発展の理由として，次のようなものが考えられる．

① データベース・システム自体は，1960 年代より実用化している．この実際面での蓄積が理論の発展への動機の一つとなっている．また，データベース・システムをさらに発展させるためには，種々の問題を解決しなければならず，理論による扱いが不可欠となってきた．

② ハードウェアの発展により，データベースが経済的に実現できるようになってきた．このため，巨大データベースや個人用データベース等に対する潜在的な需要を換起した．

③ データ自体が財産であるという認識が広がり，実際に有用なデータが種々の場所で蓄積されてきた．ニューズウィーク誌には，日本企業に NTIS をはじめとするアメリカのデータを自由に使わせるのは利敵行為である等という意見 (1979 年5月ごろ) も出されており，データの蓄積自体が国の経済や安全といった問題におよぶ可能性まで出てきている．

④ 1970 年に Codd によって提案されたデータベースの関係モデルは，データベースの諸問題を数学的に扱うための有力な道具となっており，このモデルを用いて理論面が大きく発展した．

⑤ 計算機科学の他の分野が大きく発展してきており，それらの分野で開発された手法をデータベースの分野で活用したり，逆にデータベースの側から他の分野への問題提起が行われる等，他分野との関係で発展した．

⑥ データベースの統一的発展をさまたげていた一つの原因に，各グループごとに別々の概念や用語が使用されていた面もあり，これらは，Date の教科書をはじめとする最近の努力で解消されつつある．

従来，データベースは，ソフトウェアの一部とみなされてきたが，最近は計算機科学の諸分野と大きく関係し合ってきているので，主なものを以下にまとめる．

① ソフトウェア：本来のソフトウェアは計算を対象としており，データベースのように記憶を対象としたものよりかなり複雑である．したがって，計算を対象としていると困難な問題でも，データベースでは簡単に解ける問題もある．たとえば，非手続き的言語やBackusの提案した関数型言語等は容易に定義できる．また，ソフトウェア工学，同期理論，プログラム理論の成果の一部はデータベース理論に利用されている．

② 計算機システム：現在のノイマン型計算機の持つソフトウェア作成や効率上の問題を解決するために種々の新しい試みが提案されている．この中に，データ処理の効率化を目的とした．連想記憶やデータベースマシンの研究がある．データフロー，高度分散処理，可変構造等の試みと合さって新しい型の計算機システムを生み出す可能性がある．1979年秋には1メガビットバブルメモリが市販される等，最近のハードウェアの大幅な進歩は，このようなシステム研究を活発化している．

③ 計算機基礎理論：アルゴリズム解析や効率の良いソート等の研究成果がデータベース理論にも生かされている．とくに，この分野で著名なUllmanやAhoらが論文をかなり発表しており，この分野の人人のデータベース研究はさかんになりつつある．

④ 人工知能：推論機能とデータベースを結合した高度なデータベース，自然言語によるデータベース，学習型データベース等，人工知能の手法を取り入れたデータベース研究もさかんである．

⑤ 情報検索：データベースと情報検索はともに情報の蓄積と検索を対象としているにもかかわらず，独立して研究されることが多かった．最近では両者の長所を生かしたシステムの研究もさかんになりつつある．

⑥ 符号理論：符号理論の中でデータベースに特に有用なものとして，データ圧縮と暗号がある．1976年に提案された公開キー暗号方式は，安全なシステム構成のための方法として大きく注目されている．

⑦ 計算機応用：計算機の応用分野では，データベースの比重が増しつつある．また，これらの分野から，新しいデータベース研究への動機（2次元図形の扱い等）がもたらされている．

このように，データベース研究の重要性は増加しつつある．現在の主要な研究テーマとしては次のようなものがある．

データモデル論，データベース言語，データベース設計，意味論，制約の処理，質問処理，並行処理，安全性，分散処理，同期問題，種々のデータの処理（文献，画像），高次データベース，各種応用．

データベースが従来のファイルシステムと異なる二つの大きな点は，データの統合とデータ独立の実現にある．データを統合することによって，記憶容量を減少させ，冗長度を除いたり，データ内の矛盾を避けたりすることができるが，そのため，安全性や並行処理等の面で問題が生じてきた．また，データ独立は，プログラミング言語の世界において，高級言語が機械と独立であることに類似した概念であるが，その達成のための諸問題や効率の問題等，解決すべき問題が多い．言葉を変えれば，データベース本来の機能を実現するための研究が研究の主流となっているといえる．現在のデータベースの主目的が，「個人用のデータ管理システムと全く同じように使え，かつ統合化による利点が加わり，さらに各種のデータ独立性を保障すること」であると考えると，個人用データ管理システムを中央の大型データベース・システムに結合したようなシステムが，現状における一つの解となるであろう．

データベース理論の今後の見通しとして，つぎのような要素が考えられる．

① 理論と実際の間のフィードバックによって，データベース理論はより発展してゆくであろう．とくにデータベース・システムの機能の向上，能率の向上ならびに各種機能の理論的限界の明確化等に対して，理論的立場からの検討は不可欠である．

② ハードウェアの発展とその低価格化，データベース・システム自体の高機能化等により，データベースの応用面での多様化が促進され，その結果理論面もさらに多様化してゆくであろう．

③ データベース理論の発展に伴って，計算機科学の諸分野の再構成を考える必要がある．従来は，ハードウェア，ソフトウェア，基礎理論，応用という分け方が一般的であったが，計算や記憶（データベース）に対応する研究はこれらの分野に分散しており，とくにデータの扱いに対しては，データ科学としてまとまってゆく方向にあると考えられる．

データベース・システム理論の研究開発動向と今後の課題

—分散データベース技術に関して—

高平　叡

ここで述べる分散データベースとは，地理的に分散しているデータベースを，概念的に一つのデータベースとして構成するものである．分散の形態には種々あり，また計算機の機種やデータベース管理システムの種類など多様な要因が，分散データベース技術に次のような多くの問題を提示している．

概念的に一つのデータベースとして見せるための共通インタフェースの問題として，共通のデータモデル，共通の問合せ言語，問合せの最適化など．

地理的に分散することに関わる問題として，データおよびディレクトリの配置法(重復配置/非重復配置)，同時更新制御，データの完全性・整合性の保障法，救済制御など．

計算機やデータベースの種類が異なる場合の問題として，データベース・アクセスやデータの変換，データベース制御の変換など．

次に，これ等の問題に対するシステム理論の現状を概観する．ひとまとめにして論ずることは危険を含むが，あえて，それを歴史的な動向と完成度に写して見たのが図-4 である．横軸は年代を，縦軸は関連論文をパロメータに完成度を示している．

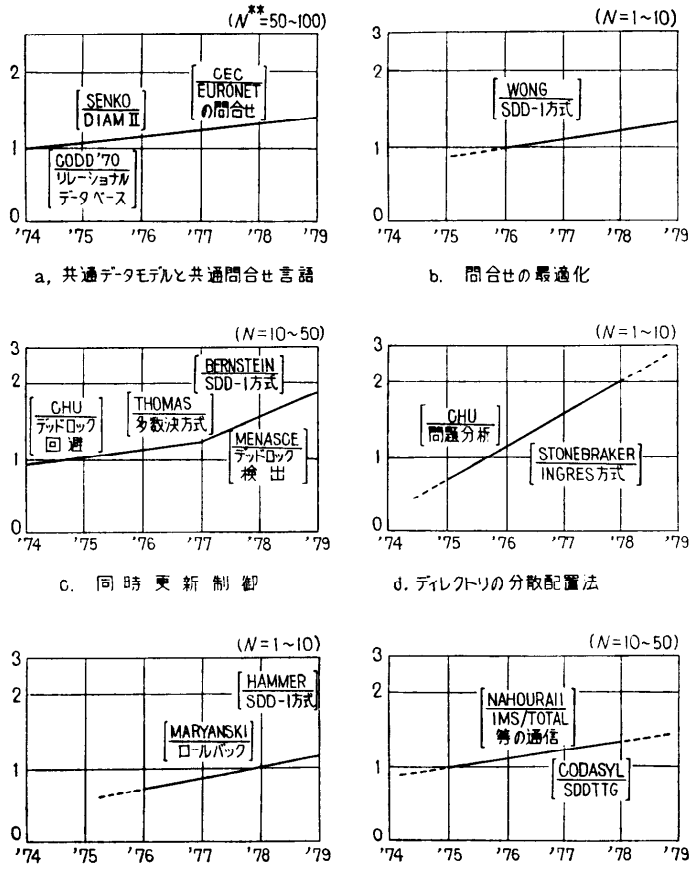
① 共通データモデル（図-4（a））

個々のデータベースを統合し概念的に一つに見せるために，すなわちグローバルな概念スキーマをつくるために，共通的なデータモデルを設定する必要がある．そのデータモデルは，データの意味が完全に記述できることは無論，個々のデータベースからのマッピングが容易でなければならない．データ・モデルについては Codd の n 項関係モデルをはじめ多くの研究がある．しかし理想的な形で，すなわち，大多数にとって共通であるようなモデルは得られていないし，一つのデータ・モデルで合意を得ることは先ず困難であろう．既存の代表的なデータ・モデルに共通なインタフェースを確立しようとすると，記述力からは低水準のモデルに，通信効率の面からは高水準なものを指向していくことになるが，当面は，同一モデル間の結合に制限するか，専用的なものが現実である．

② 問合せの最適化（図-4（b））

DB に対する問合せを最適にする問題は，DB の基本形態や配置の仕方で違ってくる．問合せの内容を分割し，いかに並列処理を行わせるかであり，Wong の関係モデルにおける最適分割の研究などがある．この問題は，処理効率の問題であって，これがシステム構築上決定的な影響をおよぼすことは少ないので除々に解決されていけばよいと言えよう．

③ 同時更新制御（図-4（c））



* 縦軸：完成度，横軸：年代　　** N：主要文献に発表された論文数

0：未着手　1：問題点の明確化　2：対案が出そろった　3：解決

図-4 分散データベース関連技術の動向

複数個のトランザクションが同時に分散したデータベースを更新する場合に，データベースの内容に矛盾を生じさせないようにする問題である．矛盾には，完全性と整合性とがある．完全性とは例えば一つのデータに対し複数の更新要求があるとき，複数の更新要求が正しく重ね合せられることであり，データを読取って処理を加え書込むという一連の流れの中に他の流れが混入しないように防止することである．整合性とは，分散データベースに重復配置したデータが正しく合致していることであり，更新が直列に行われている途中を横から見ると矛盾を生じていることになり，この状態時にアクセスを許すと誤まりを生じるし，障害があると不整合のままになることがある．この制御方式には，ロック方式や非ロック方式で多数の具体的な提案がされており，基本的な方式は，ほぼ出そろったかの感がある．通信コストや遅れを小さくする，各ノードを公平に扱う，などの観点から改良が進められよう．

④ ディレクトリの分散配置法（図-4（d））

データが現実にどこに存在しているかの情報を持っているディレクトリの作り方に関するものである．構造定義情報やファイル情報など，どこまで取込んでおくか，それを各ノードに重復して置くのか，一ヵ所に集中して置くのかなどである．これらは，実際のシステム設計において，問合せの処理効率や記憶容量，保守等の面から決定されることになる．

⑤ 救済制御（図-4（e））

一つのノードの障害が他のノードに影響を与えないようにすることが重要である．障害回復の問題は，同時更新制御における手法とも関連がある．本格的な研究は，実際のシステムが設計される段階でされるようになろう．

⑥ 異種性に関する技術（図-4（f））

各ノードのデータモデルあるいは問合せ言語と，共通インタフェースとの対応問題である．一つはデータベースとデータの変換に関する．もう一つは，データベース制御の変換である．前者は，データモデルの相互変換という汎用的な解決の方向と，問合せ言語レベルでの専用的な方向とがあり，そこでの変換率を上げる問題は，共通インタフェースの問題に帰着する．後者の問題は，更新の同期制御や救済制御等，アクセス対象のノード相互で矛盾が無いようにするため，各ノードの既存の制御法と整合をとることであるが，インプリメント法に依存することから，まだほとんど触れられておらず，またかなり難かしいであろう．

以上，ごく簡単に動向を見たが，研究の歴史も浅く今後に待つところが多い．しかし実用化は分散データベースのニーズに対応して，各技術に制限を加えた範囲の中で進められ，並行して理論研究が深められていくことになろう．

手軽でインテリジェントなデータベース・システム

古川 康一

将来，情報システムが一般家庭をも含めてより広範に利用されることが考えられるが，そのような状況でのデータベースは，

① システムが簡潔で，容易に専用のデータベースを作り上げることができること，

② 各種の変更に対処できるように柔軟であること，

③ 使い易いこと，

が要求される．以下に，この三つの要求を満たすための技術的課題を中心に述べる．

（1） システムの簡潔性の達成

データベース・システムは，現在のところ，OS と並んで巨大なソフトウェア・システムとなっている．データベースの機能を高度化しようと思えば，その傾向はより強まるであろう．すると，新しい計算機システムが現れる度に，その巨大なソフトウェア・システムを開発しなければならなくなり，データベースが非常に高価なものについてしまう．

このようなデータベースの巨大化を防ぐ手だてとしては，

① ハードウェアの高機能化，

② ソフトウェア開発技術の改善

の二つが考えられる．

ハードウェアの高機能化には，具体的には，データベース・マシンの開発がある．これによって，低レベルのデータ・アクセスをハードウェア化することが可能となり，プログラムはずっと簡潔になる．

これに加えて，ここでは，記号処理マシンの開発を，データベース・システムの簡潔化に役立つものとして取り上げたい．このマシンは，データベース・アクセスの，より高いレベルの処理を行う．高いレベルとは，たとえば自然言語による質問応答を考えれば，自然言語理解を行ったり，そのレベルの質問を，より低いレベルの質問，あるいは検索手続きへ変換したりするこ

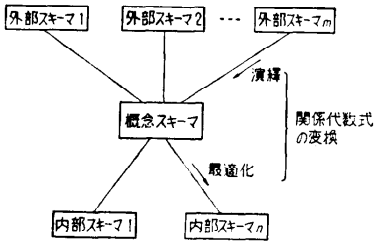


図-5　3層スキーマ構造と質問の変換

とである.

ソフトウェア開発技術の改善は，近年のソフトウェア工学の成果であるデータの抽象化と，それに基づくシステムの階層化が有用であろう．さらには，代数的仕様などの形式的なアプローチも注目に値する.

(2)　柔軟性の達成

柔軟性は，データ独立性と言い換えることもできる．ANSI/X3/SPARC [1975]*によって提唱された3層スキーマ構造は，このデータ独立性を，二つの成分に分解してとらえている．その第1は，従来の，応用プログラムのデータ表現からの独立性で，図-5 での概念スキーマ・内部スキーマ間のインタフェースと考えられる．第2は，概念スキーマのユーザの質問言語からの独立性で，図-5 の外部スキーマと概念スキーマ間のインタフェースと考えられる.

図-5 に示すように，外部スキーマ・概念スキーマ間の変換は，演繹過程によって実現できる．すなわち，これらのスキーマ間の論理的な関係を記述しておき，その記述を用いて，外部スキーマ上の質問文を概念スキーマ上の質問文に，演繹的に変換することが可能である.

一方，概念スキーマ・内部スキーマ間の変換は，最適化の過程と見なすことができる．内部スキーマは，データの物理表現を規定しているが，その物理表現に適した検索手続きを求めることが必要となる．この変換は，最適表現を得るためにデータ表現を動的に変更した場合にも追従できなければならない.

私がここで提案したいのは，これら二つの変換を，関係代数式の変換によって行う方法である．そのためには，外部スキーマ・概念スキーマ間，および概念スキーマ・内部スキーマ間の関係を関係代数方程式で表わし，それらの式と，他の関係代数の諸性質を用いて，外部スキーマ上の質問を，それと等価な内部スキーマ上の検索手続きに変換する機構が必要となる．この機構は，(1)で述べた記号処理マシンの上で効率良く処理されるであろう.

このアプローチは，Backus [1978] の Functional Programming のスタイルに類似している．ここで，Functional Programming で問題となるデータの更新問題を考えてみよう．Backus は，データの更新によって，計算の環境が変更されるので，特別な扱いが必要であることを述べている．しかしながら，計算の環境にコンテキスト機構を導入することによって，前の環境をこわさずに，新たな環境を作り出すことができる．すなわち，データの更新をコンテキスト機構を用いて行うことにより，functionality を保つことが可能となる．さらに，このような更新機構を用いることによって，過去に遡ったり，誤った更新を容易に取り消したりすることが可能となる.

(3)　使い易さの達成

人間とのやりとりに，人間同士が会話をするときに用いる通信手段を使えることが最も望ましい．それは自然言語であり，数式であり，図面であり，イメージである．データベースへのアクセスを考えると，この中でも特に自然言語による質問応答が，現在関心を集めている．自然言語理解は難しい問題であるが，用途を限ればかなり成功することも示されてきている．とくに，構文解析プログラムの進歩，意味処理導入の簡単化，データベースの辞書化などの技術により，各用途に応じた自然言語理解システムを短時間で作れるようになってきた.

使い易い質問応答システムを実現するには，自然言語を単に理解しただけでは不十分である．より本質的なことは，自然言語で表現された質問の意図を理解することである．これは，自然言語理解以上に困難な問題であるが，自然言語理解と同様に，用途を限り，対象領域を十分に制限すれば，扱い得るであろう.

最後に，研究の方向付けについての一般的なコメントをするならば，効率よりも機能を重視して研究目標を設定するのが良いと思われる．莫大な計算時間を要するような処理でも，その機能が有用なものであれば，ハードウェアの進歩に伴って，いつの日か実用的な技術となるであろう.

質疑応答

質問　1

氏名　米田　茂　(日立製作所)

質問　System R のアプリケーションの記述におい

* ANSI/X3/SPARC [1975] "ANSI/X3/SPARC Study Group on Data Base Management Systems, Interim Report", ACM FDT, 7, 2, 1975.

て，SQL 2 を用いるより HOST 言語 (COBOL, PL/I その他) を用いた方法が多かった．これは ① SQL-2 の記述能力が不足していたのか ② 人間の考える思考方法が手続き的であり，SQL を用いた考え方より手続き的な HOST 言語の考え方に近いのか，いずれと考えられるか。

植村 SQL の記述力が不十分であることは確かである．問合せ言語であるから，作表などの出力能力が不足している．また同じような問合せを繰り返す時，SQL では毎回同じことをやる必要があるというわずらわしさもある．もちろんこれらは今思い出した少数例に過ぎず，御質問の理由は推測に頼る他ない．

質問 2

氏名 石田喬也 (三菱電機)

質問 conceptual schema を relational model 等，第1階述語論理の世界で記述することには限界がある (relationship 自体を entity あるいは object とみなすなど)．それゆえ高階述語論理の世界で表現する方法はないか？ そのあたりの基礎理論の研究の見通しはどうか．

上林 問題は可能性のあるものはすべてやるか，まずできることをきちんとやるかの態度の相違である．この問題は人工知能研究にも関連し，この分野からのインパクトとしても，積極的に未知領域の開拓をする行き方とできる範囲のことをやる行き方がある．データベース側としては後者の立場でやる方が良いと思う．

古川 relationship を entity とすること自体は高階論理を使わなくてもできる．relation 名を変数とすることは2階論理とも言えるが，かなりの程度1階論理でもできるはずである．

質問 3

氏名 増永良文 (東北大学 電気通信研究所)

質問 パネラ各自が各自の立場からデータベース研究動向を述べられたが，本年度の全国大会講演をみるに，データベース・モデル理論のみならず，データベース・システムを実際設計・作製しようという動向が顕著であった．この様な動向を認識して，今後データベース・システム開発を行おうとする研究者，グループに対してパネラの方々のアドバイスがあれば伺いたい．

植村 開発しようとするシステムの具体的な目標により異なるので一概に言えない，現在開発されているのは文献検索システムが多いようだが，こういうシステムは外国ですでにいくつか存在しているのでそれをできるだけ参考にするのが良いだろう．

質問 4

氏名 椿 正明 (千代田化工建設)

質問(コメント) データベースは使うためにあるので開発だけでなく使用のことをもっと考えるべきではないか，たとえばどんな組織で運用するか，データベースの内容を保全するためにどんな手続きを設定するかなどを考えれば，view update などはあまり現実的でないことが分るであろう．

またデータベース設計において誰がいかに conceptual schema を設計するかを考えれば一発開発的なデータベース設計アプローチにどんな限界があるか分るであろう．また一つの information processing environment に多くのデータベースがあり，この間の integrity にむずかしい問題があることも分るであろう．

生の問題から問題を見つけて来て研究をすすめるよう希望したい．

司会 貴重な御意見として拝聴するが，本日のパネルはデータベース理論の研究開発動向と今後の課題がテーマであるため，理論面が主たる話題となった．