Vol. 37 No. 7　　情　報　処　理　　July 1996

解　説

データベース関連技術の標準化

2. SQL-92の追加機能とSQL3†

芝　野　耕　司††

1. はじめに

データベース言語SQLは，1970年代にIBM社の基礎研究所で開発された言語仕様をもとに，1987年に日本工業規格[1]および国際規格[2]として制定し，その後，2回の改訂を行ってきた．そして現在では，主要なデータベース管理システムのすべてがこの言語を採用している．

1987年に制定した最初の国際規格では，関係データベース管理システムを用いる一般のアプリケーションが必要とする機能だけを中心として，言語仕様の開発を行った．1989年に出版した国際規格第2版では，関係モデルのより良いサポートを目指し，参照整合性を含む整合性拡張機能の追加を行った．JISでは，この版で，日本語機能の追加を行った．1992年に出版した国際規格第3版では，一般のアプリケーション開発だけではなく，データベース管理システムを用いるすべてのアプリケーションで必要とするすべての機能の追加，すなわち，スキーマ操作言語，動的SQL，情報スキーマの提供などを行い，関係データベース管理システム用のデータベース言語としての完成を目指した．

SQL-92となったSQL 2は，SQL-87に引き続き開発を行い，SQL 3は，SQL 2，すなわち，現在のSQL-92の開発で持ち越した課題から，開発を始めた．SQL 3の最初の仕様は，SQL-92 (SQL 2) を分離した1987年にさかのぼる．

また，SQL 3の仕様は，これまでの仕様と100%上位互換性を保持するように開発している．

その後，コンピュータ環境の変化にともない，SQL-92の機能拡張として，Microsoft社のODBC (Open Database Connectivity) をもとにしたSQL/CLIおよびSQL言語に計算完備でサーバに格納可能なモジュール機能を与えるSQL/PSMの2つの機能の追加を行った．SQL/CLI[3]は，1995年に出版し，SQL/PSM[4]は，1996年に出版することを予定している．

初版と第2版のSQL言語は，百数十ページの言語仕様であったが，SQL-92では，700ページ近い仕様となった．最近のプログラム言語仕様では，言語の基本仕様自体は，50ページくらいまでで抑え，残りは，ライブラリ仕様として分離することによって，言語仕様自体を小さくすることが望ましいとされている．しかし，データベース言語がプログラム言語，通信，オペレーティングシステム制御など基本ソフトウェアの主要要素全体にわたる機能を含むことから仕様の増大は，避けられないことではあろうが，この仕様の増大によって，SQL言語を理解することが難しくなることを避けるため，SQL言語仕様を複数パートの規格とすることとした．

現在のSQL 3は，次の9つのパートからなる．

Part 1: Framework (SQL/Framework)
Part 2: Foundation (SQL/Foundation)
Part 3: Call-Level Interface (SQL/CLI)
Part 4: Persistent Stored Modules (SQL/PSM)
Part 5: Host Language Bindings (SQL/Bindings)
Part 6: XA Specialization (SQL/Transaction)
Part 7: Temporal (SQL/Temporal)
Part 8: Extended Objects (SQL/Object)
Part 9: Virtual Tables (SQL/Virtual Table)

これらのパートのうち，Part 2およびPart 5

† The SQL 92 Follow on and the SQL 3 by Kohji SHIBANO (Tokyo International University).
†† 東京国際大学

によって，SQL-92を置き換えることを予定している．

以下，SQL-92の拡張として，制定または近いうちに制定予定のSQL/CLIおよびSQL/PSMについて述べる．ついで，SQL 3の機能について述べる．

2. SQL-92の拡張

従来の大型計算機を中心としたコンピュータ利用は，1980年代になって大きく変わってきている．1990年代には，ネットワークを利用したクライアントサーバ型のコンピュータ利用が主流となってきた．

また，アプリケーション開発においても，従来のプログラム言語を用いて，直接個別アプリケーションを開発する形態から，表計算ソフトなどクライアントでのパッケージアプリケーションとデータベース管理システムとの統合的な利用形態が必要とされるようになった．

一方，サーバの中心的なソフトとしてのデータベースの利用も，大きく拡大し，トランザクション処理に対する性能要求も大幅に増大した．

こうした新しい利用環境に対応するために，SQL 3に先立ち，SQL-92に対する追加機能として，SQL/CLIおよびSQL/PSMの開発を先行させることとした．

2.1 SQL/CLI (Call Level Interface)

SQL/CLIは，従来のSQL言語がSQLアプリケーションのソースコードレベルでの可搬性を保証するための仕様であったのに対して，SQLの機能を呼び出すためのインタフェースを標準化することによって，SQLアプリケーションの実行コードの可搬性を提供することを目指したものである．

この機能によって，従来不可能であった表計算ソフトなどのように，実行コードだけが販売されるようなパソコンパッケージソフトからデータベース機能を呼び出すことを可能にする．

また，この機能は，Excelの装置独立のprinter driver一般化から生まれたMicrosoftによるODBCをもとに開発した仕様で，SAG (SQL Access Group) およびX/Openを通じてISOへ提案された．

当初日本は，動的SQLと似て非なるものであるので，目的は賛成するが仕様に関しては，全面書き直しを主張し，大幅にSQL-92に沿うように書き直し，1995年末に国際規格を出版した．

2.1.1 SQL/CLIの仕様概要

SQL/CLIは，基本的な機能としては，これまでの動的SQLとほぼ同等の機能を与える．しかし，動的SQLがソースレベルでの可搬性を与えているのに対して，SQL/CLIは，実行コードからの呼び出しで，SQL文を実行する新しい仕組みを提供し，次の機能を提供する40以上のルーチンからなる．

(1) 計算資源の割当てと開放
(2) SQLサーバとのコネクションの制御
(3) 動的SQLと似た仕組みでのSQL文の実行
(4) 診断情報の獲得
(5) トランザクションの制御
(6) 実装に関する情報の取得

CLIでは，資源の管理にハンドルを用いる．このハンドルによって，SQL文の処理に必要な資源の管理を行う．

また，CLIでは，アプリケーションパラメタ記述子 (APD)，アプリケーション行記述子 (ARD)，実装パラメタ記述子 (IPD)，実装行記述子 (IRD) の4つの記述子領域を用いて，情報の管理を行う．これらの記述子領域は，動的SQLにおける記述子領域とほぼ同様な機能を持つ．

2.1.2 SQL/CLIの仕様詳細

AllocHandleは，SQL環境，SQLコネクション，CLI記述子領域またはSQL文の処理に必要な資源を割り当て，ExecDirectは，SQL文を1回実行するために用い，PrepareはExecuteを用いてのそれ以降のSQL文の実行のために準備を行う．SQL文の実行では，動的パラメタを用いることができる．

CLI記述子領域は，動的パラメタおよびその値，結果列ならびに結果列に対する相手指定の記述インタフェースを与え，SetStmtAttrによって設定し，GetStmtAttrによって情報を取得する．GetDescFieldおよびGetDescRecによってCLIk記述子領域から情報を得る．Copy Descによって1つの記述子領域の情報を別の記述子領域に複写することができる．DescribeCol,

ColAttribute, NumResultCols によっても記述子領域の情報を得ることができる. SetDescField, SetDescRec によって明示的に記述子領域に情報を設定できる. また, BindCol によって暗に設定もできる. CLI 記述子領域は, SQL 文の準備または即時実行の際に, そのときの SQL コネクションで可能であれば, 自動的にも設定される.

CLI では, <動的 select 文>または<動的単一行 select 文>の実行の際, カーソルが暗に宣言され, 開かれる. カーソル名は, SetCursorName によってアプリケーションから与えることができる. システムが設定するカーソル名は, GetCursorName によって得ることができる.

通常の SQL の Fetch は, CLI では Fetch と ScrollFetch の 2 つに分離. データのやりとりは, PutData および GetData を用い, 結合していない列の値のやりとりを行う. 長い文字列についても, 一部ずつこれらのルーチンによってやりとりすることができる.

<準備可能動的 delete 文: 位置づけ>, <準備可能動的 update 文: 位置づけ>によって, 行を削除または更新する. CloseCursor によってカーソルを閉じる.

SQL の実行状態についての詳細な情報は, 診断領域によって, クライアント側のアプリケーションが取得することができる. すなわち, GetDiagField および GetDiagRec によって診断情報を取得する.

また, Error によって直前に実行したルーチンの診断情報を得る. RowCount および GetDiagField によって直前に実行した SQL 文によって影響を受けた行の数に関する情報を得る.

トランザクションについては, 最初の SQL 文の実行によって, トランザクションを開始し, EndTran によってトランザクションを終了する. Cancel によって並行 CLI ルーチンを取り消す.

GetFunctions, GetInfo および GetTypeInfo によって実装に関する情報を得る.

2.1.3 SQL/CLI の JIS

SQL/CLI の JIS 原案は, 1995 年度末に開発を終了しており, 1996 年後半にも JIS を制定する予定である.

2.2 SQL/PSM (Persistent Stored Module)

SQL/PSM は, SQL 言語の計算完備な拡張(プログラム言語機能) および永続データベース言語手続きの宣言, すなわち, サーバへの SQL モジュールの格納を可能にする. この SQL/PSM には, 手続き中でのフロー制御文, 局所カーソル, 局所変数, 局所 1 次表などの機能が含まれる.

SQL/PSM の開発の背景には, 次のいくつかの問題が存在する. すなわち, (1)多くの製品での 4 GL またはプログラム言語拡張の提供, (2)クライアントサーバ環境への対応, (3)クライアントサーバ間の通信量を減らす, (4)オンライントランザクションアプリケーションの高性能化, (5)ホスト言語とのミスマッチの解消, (6)ホスト言語での各国文字集合機能の開発の遅れなどである.

また, すべてを SQL 言語でプログラム可能にすることによって, データの安全性を確保する防火壁などの安全性の強化も可能となる.

そして, SQL だけでのプログラミングを可能にすることによって, SQL 自身によるクラスライブラリの構築, すなわち, オブジェクト指向拡張の基盤を与えることになる.

2.2.1 SQL/PSM の仕様概要

SQL/PSM では, 従来のアプリケーションクライアント側でのモジュール以外に, サーバに格納されるモジュールおよびルーチンの作成, 変更および削除が可能となる. 同時にそれぞれのモジュール中で複数の SQL 文を含めることが可能になるほか, 複合文, 一般の制御文, 手続きおよび関数呼び出しならびに代入文など, 実行フロー制御が可能となる.

また, 例外処理の宣言と signal/resignal 文の機能が提供されるとともに, 局所宣言としても, 通常のプログラム言語でよく見られる変数, 関数, 手続きに加えて, カーソル, 1 次表の宣言が可能となる.

2.2.2 SQL/PSM の主要機能

SQL/PSM のプログラム例[5] を次にあげる. この例では, 一連のトランザクションをサーバに格納する 1 つのモジュールとして定義している.

```
CREATE MODULE transaction
  LANGUAGE SQL
```

```
  SCHEMA catalog1. schema2.
DECLARE C1 CURSOR FOR...
PROCEDURE get_balance (SQLCODE,
  account_# INTEGER,
  balance DECIMAL (15, 2));
  SELECT balance INTO balance
  FROM accounts
  WHERE account_#=account_#;
PROCEDURE withdraw (SQLCODE,
  account_# INT,
  amount DECIMAL (15, 2);
  UPDATE accounts
  SET balance=balance-amount
  WHERE account_#=account_#;
PROCEDURE deposit (SQLCODE,
  account_# INTEGER,
  amount DECIMAL (15, 2));
  UPDATE accounts
  SET balance=balance+amount
  WHERE account_#=account_#;
END MODULE;
```

SQL/PSMでは，例にあげたサーバモジュールの作成，変更，削除の機能だけではなく，前述のように一般のプログラム言語に相当する言語機能を提供する．この言語機能は，オラクル社のPL/SQLやサイベース社のTransact SQLに相当する機能である．

主な機能は，次のとおり．

-複合文	BEGIN...END
-SQL変数	DECLARE var
-IF文	IF subject (var)<>'urgent' THEN...ELSE...
-CASE文	CASE subject (var) WHEN 'SQL' THEN... WHEN...;
-WHILE文	WHILE i<100 LOOP...END LOOP;
-FOR文	FOR result AS...DO...END FOR;
-LEAVE文	LEAVE...;
-RETURN文	RETURN 'urgent';
-CALL文	CALL procedure_x (1, 3, 5);
-代入文	SET x='abc';
-SIGNAL文	SIGNAL division_by_zero

また，SQL/PSMでの複合文に省略可能なキーワードATOMICを指定することによって，複合文に含まれる複数のSQL文を1つの原子トランザクションとして取り扱うことを指定することができる．

この複合文でのATOMIC指定は，System-Rでの最初のSQLの実装以来，SQLトランザクションが1つのSQL文からなる原子トランザクションのグループとして取り扱われてきたトランザクションの取扱いの大きな拡張にあたる．

3. SQL3の概要

SQL 3の機能拡張にともなって，SQL 3を複数のパートからなる仕様に変更した．前述のSQL/CLIおよびSQL/PSMは，この新しい構成で出す，最初の一連の仕様である．

Part 1: Framework (SQL/Framework) は，全体のパートの構成を記述する．Part 2: Foundation (SQL/Foundation)[6] は，SQL 3の核となる仕様であり，SQLの主要な機能を含む．Part 3およびPart 4は，それぞれ前述のSQL/CLIおよびSQL/PSMに対応する．Part 5: Host Language Bindings (SQL/Bindings) は，SQLとホストプログラム言語との結合についての仕様であり，Part 2とこれの2つで現行のSQL-92を置き換える．

残りの3つのパートは，最近になって開発を始めた仕様である．Part 6: XA Specialization (SQL/Transaction) は，X/Openで開発しているトランザクションマネジャと資源マネジャとの間のインタフェースを規定するXAインタフェースのSQLへの適用を規定する．Part 7: Temporal (SQL/Temporal) は，時制データベース機能を指定する．Part 8: Extended Objects (SQL/Object) は，オブジェクト識別性に関する仕様をまとめたものであるが，これに関しては，Part 2から分離したものであり，また再統合する可能性が高い．Part 9: Virtual Tables (SQL/VirtualTable) は，SQLの表以外のISAMファイルなどの外部ファイルに対してのSQLのインタフェースを規定する仕様である．

ついで，SQL/Foundationの主要な内容について述べよう．

3.1 SQL/Foundationの概要

SQL/Foundationでは，次の機能の拡張を行った．

-データ型
-文字集合サポート
-抽象データ型
-ユーザ定義演算子
-問合せ式の拡張
-述語の拡張
-表の拡張
-参照整合性の拡張
-トリガ
-カーソル機能の拡張
-トランザクション管理機能の拡張

SQL 3では，データ型の拡張として，真偽値をとるブール型，行を値とする一般のプログラム言語でのレコードにあたる行型，バイナリおよび文字巨大オブジェクト型（BLOBおよびCLOB）の機能が追加された．同時に構造型として，集合，マルチ集合およびリスト型の追加を行った．これらの追加，特に，BLOB，CLOBおよび構造型の導入によって，マルチメディアデータや複合オブジェクトのサポートが可能となった．

SQL-92で導入した文字集合サポートでは，事前に準備された文字集合を参照することだけが可能であったが，SQL 3では，文字集合，照合順番，文字変換を最初から自由に定義することが可能となる．

オブジェクト指向拡張にあたる抽象データ型およびユーザ定義演算子に関しては，後述する．

問合せ式に関する拡張としては，結合表の更新可能性に関する拡張，再起的問合せなどの拡張がある．

述語についても，文字列の正規表現を用いた検索を可能とするSIMILAR述語，FOR SOME，FOR ALLなどからなる限定述語，集合の重複を検査するDISTINCT述語，そしてブール述語の拡張を行った．

表定義に関しては，表定義時に既存の表定義を参照するLIKE句，意味データモデルでの一般化/特定化に対応する副表およびそれに関連するデータ操作規則などの拡張を行った．

参照整合性では，SQL-89での参照動作をともなわない整合性規則を明示的に指定するRESCRICTを追加した．

また，新たにトリガ機能を導入した．トリガでは，トリガを指定する表に対するデータ操作のイベント種類，データ操作の前後とトリガを発生する時期を指定し，複数のトリガ動作を指定することができる．

カーソルについては，カーソルを経由してのデータ更新が他から見えるかどうかを指定するSENSITIVEおよびCOMMITを超えて，カーソルの指す位置を保持するWITH HOLDオプションを追加した．

データ安全性に関しては，役割（role）の概念を導入し，拡張を行った．

トランザクションに関しても，明示的にトランザクションの開始を指定するSTART TRANSACTION文，SAVEPOINTの設定，連鎖トランザクションを指定するCHAINオプションを追加した．

3.2 SQL3の今後の予定

SQL 3は，1996年8月にCD（Committee Draft）として，郵便投票を開始する．1997年春には，この郵便投票結果を審議する編集会議を開き，1997年にDIS（Draft International Standard）とすることを予定している．そして，1998年には，新しい国際規格として，これまでのSQL-92を改正することを予定している．

4. SQL3でのオブジェクト指向拡張の概要

SQL 3での大きな開発項目には，オブジェクト指向拡張があった．このオブジェクト指向拡張では，ユーザ定義型，抽象データ型から始め，名前付きのユーザ定義データ型に，その操作，振舞いの記述および内部の情報隠蔽の機能を入れ，複合オブジェクトをサポートするとともに，オブジェクト識別性を取り入れたオブジェクト指向拡張へと開発を進めてきた．

ここでは，このSQL 3でのオブジェクト指向拡張の主要な話題に関して，概要を述べる．

4.1 関係モデルとオブジェクトモデル—情報隠蔽問題

データベース管理システムは，データ構造とアクセス構造とが同一である構造型のデータベースから始まった．これらのデータ間をリンクポインタでつないだ構造型のデータモデルでは，データ

独立性が確保できないことから，この問題を解決するために，関係モデルが提案され，この関係モデルのための言語として，SQL言語が設計された．

すなわち，関係モデルでは，データ構造として，単純な表形式を採用し，アクセス構造とデータ構造とを分離し，データを値からだけ検索し，同時に，データベース中のデータは，その値によってだけ，識別することを可能とした．すなわち，関係モデルは，値指向のモデルである．

一方，オブジェクトモデルは，一種先祖返り的な特徴を持つ．すなわち，オブジェクトモデルでは，データベース中のデータは，それ自身識別可能であり，このオブジェクト識別性をもとに，データ操作を行う．

2次資料を中心とする客観データの管理に関係モデルの値指向モデルが適しているが，マルチメディアデータなどの新しいアプリケーションでは，オブジェクト識別性が重要な意味を持つ．しかし，実際の応用面では，従来のデータと新しいタイプのデータとの統合的な管理が要請される．

こうしたことから，SQL3では，従来の値指向に加えて，オブジェクト識別性を持つオブジェクト指向を統合することを目指した．

現仕様では，これを行型（ROW TYPE）および参照型（REF TYPE）によって実現している[7]．この機能を使った行型および表定義の例を次に示す．

```
CREATE ROW TYPE address_t(
  street VARCHAR (120),
  city VARCHAR (68),
  state CHAR (2),
  zip CHAR (10)
  );
CREATE ROW TYPE customer_t(
  customer_id REF (customer_t),1
  ssno INT,
  name VARCHAR (50),
  address address_t,
  );
CREATE TABLE customer OF customer_t (
  (PRIMARY KEY ssno,
  VALUES FOR customer_id ARE
  SYSTEM
  GENERATED 2);
```

4.2 抽象データ型

一般のプログラム言語でのオブジェクト指向に相当する機能は，抽象データ型として実現した．抽象データ型では，完全に情報は隠蔽され，抽象データ型の属性へのアクセスは，関数だけを通して実現する．

抽象データ型の実現値は，生成関数（constructor function）によって生成される．属性へのアクセスは，observer functionsによって行い，変更は，mutator functionsによって行う．属性には，PUBLIC，PRIVATEまたはPROTECTEDを指定することができる．こうした関数の定義には，SQL自身を用いることができる．

また，この抽象データ型は，データ型を用いるところではどこでも指定することができ，権限によって，データの安全性が保証される．

4.3 クラス階層と呼び出しルーチン決定問題

抽象データ型には，クラス階層を指定することができ，1つの抽象データ型は，1つ以上の上位型（supertype）の副型（subtype）になることができる．すなわち，多重継承がサポートされる．

継承としては，属性からなる構造および関数からなる振舞いを継承する．

また，関数は，置き換え（overload）可能である．

1つの抽象データ型の実現値は，最も特定的なデータ型を持ち，生成されるときに，この型が与えられる．副型は，上位型を用いることのできるところではどこでも用いることができる．すなわち，引数，変数，列，関数の結果などで用いることができる．

多様関数（polymorphism）もサポートされる．

呼び出しルーチンの決定は，引数の型による．

抽象データ型に定義された関数は，演算子としても，定義することができる．

個別型（Distinct Types）によって，強い型付けをサポートする．次に例をあげる．

```
CREATE VALUE TYPE US_dollar
UNDER money
  CAST (us_dollar as cdn_dollar)
```

```
  WITH us_dollar
  (cdn_dollar)
FUNCTION us_dollar (c cdn_dollar)
  RETURNS us_dollar
BEGIN
  DECLARE u us_dollar;
  /*create us_dollar value*/
  SET u=us_dollar();
  SET u..amount=cd..amount*1.3;
  /*exchange rate*/
  RETURN u;
  END
```

5. おわりに

次世代のデータベース言語仕様の開発を目指したSQL 3の開発には，多くの年月が必要であった．そして，今年中には，規格仕様案がまとまり，来年には，国際規格案（DIS），再来年には，国際規格制定の見込みである．

SQL-92では，データベース管理システムのメーカは，その規格制定および規格仕様への追随に関して，消極的であった．しかし，今回のSQL 3仕様に関しては，多くのデータベース管理システムの開発者達が，早期の実装を目標に，早期の規格制定を要望している．

こうした背景には，SQL言語の大きな成功があることは疑いない．同時に，SQL製品の開発社がそれぞれきわめて有力な企業として，ソフトウェア産業界で，中心的な地位を占め，企業規模，開発力とも大きな力を持つようになってきたことも，この変化の大きな要因となっているであろう．

同時に，コンピュータ利用の大きな広がりによって，従来のアプリケーションを超えた場面でのデータベース管理システムの利用の広がりおよび新しい要求へ対応する仕様として，SQL 3に期待が集まっていることも，無視できない．

また，SQL 3の機能仕様は，マルチメディアデータベース言語仕様であるSQL/MMの開発で実際に用いることによって，新しいアプリケーションへの対応を証明しつつ開発を行っていることも，SQL 3への期待が強まっている大きな原因であろう．

一方，日本国内でのデータベース管理システムの開発は，大型計算機システムの衰退にともなって，衰退しているように思われる．しかし，次世代のSQL 3仕様の規格化および計算機環境の大きな変化は，国内でのデータベース管理システムの研究開発面での遅れを取り戻す絶好の機会である，と思われる．

こうした点で，SQL 3にどう取り組むのかが大きな問題として突きつけられている．

参 考 文 献

1) JIS X 3005, データベース言語SQL (1995).
2) ISO/IEC 9075, Information Technology Database Languages-SQL (1992).
3) ISO/IEC 9075-3, Information Technology Database Languages-SQL-SQL/CLI (1995).
4) DIS 9075-4, Information Technology Database Languages-SQL-SQL/PSM (1995).
5) Mattos, N.: An Overview of the SQL 3 Standard, Personal Communication (1995).
6) ISO/IEC JTC 1/SC 21/WG 3 DBL-MCI-004, ISO Working Draft Database Language SQL-Part 2: SQL/Foundation (1996).
7) Kulkarni, K. et al.: Introducing Reference Types and Cleaning up SQL 3 s Object Model, DBL LHR-77 (1996).

(平成8年4月16日受付)

芝野 耕司（正会員）

1979年京都大学大学院工学研究科数理工学専攻博士課程前期課程修了，三井情報開発(株)，日本アイ・ビー・エム(株)東京基礎研究所を経て，現在，東京国際大学商学部経営情報学科教授．ISO/IEC SC 21/WG 3 SQL/MM RG主査，情報規格調査会SQL SG主査，SQL/MM SG主査，JIS X 3005 SQL原案委員会主査，JIS X 4051日本語組版原案委員会主査，JIS X 4061日本語照合順番原案委員会主査，情報規格調査会SC 2専門委員会委員長，JISコード委員会委員長など．工業標準調査会，産業構造審議会臨時委員．主な研究テーマ：データベース管理システム，電子文書処理，フルテキストマルチメディアデータベース，日本語処理，文字コード．