データベース・システム 59－6
（１９８７．５．１８）

データベース言語 SQL

芝野耕司

日本アイ・ビー・エム株式会社東京基礎研究所
102 東京都千代田区三番町 5-19

現在，国際標準化機構 (ISO) 及び日本規格協会で標準化が進められているデータベース言語 SQL は，関係データベースシステムのための言語として広く普及しており，今後その重要性を一層増すと思われる。本稿では，規格制定の経緯，基本概念及び特徴について述べる。データベース言語 SQL は，また，現在 ISO 参加各国の積極的な貢献のもとで精力的な改良及び拡張の作業が続けられている。本稿では，データベース言語 SQL補遺 1 及びデータベース言語 SQL2 の機能として現在考えられている機能についても述べる。

Database Language SQL

Kohji Shibano

**Tokyo Research Laboratory, IBM Japan Ltd.
5-19, Sanban-cho, Chiyoda-ku, Tokyo 102, JAPAN**

Database Language SQL, standardization of which is being progressed by ISO (International Organization for Standardization) and JSA (Japan Standard Association), has been widely spreaded as a language for relational database systems and has gained growing attentions. In this paper, we describes the history of its standardization, its basic concepts, and its functions. Since ISO member bodies are kept their keen attentions and are continuing to contribute to it, the SQL language is subject to active improvements and enhancements. In this paper, we also describes Proposed Draft Addendum 1 and Database Language SQL2.

はじめに

データベース言語 SQL は，IBM 社 San Jose 研究所の E. F. Codd によって1969年から1970年にかけて提案された関係データベースの理論に基づいて開発された言語である[Codd69]。SQLは，同研究所の D. Chamberlinによって最初に提案された[Chamberlin74]。D. Chamberlin は，同研究所で開発された関係データベース管理システム System-R [Astrahan76] のための言語としてこの言語を提案した。この言語は，彼によって構造化問合せ言語 (Structured Query Language) の頭字語をとってSQL と名付けられた[1]。

SQL 言語は，その後，関係データベース管理システムのための言語として数多くのシステムで用いられ，複数の実用関係データベース管理システムで実装されているという意味で殆ど唯一の言語となった[Date86a]。このような現実を踏まえ，国際標準化機構 (International Organization for Standardization, ISO) では，この SQL 言語を標準の関係データベース言語とすることに決めようとしている。

本稿では，関係データベースのための標準データベース言語である SQL について，規格原案[SQL87][日本規格協会87]をもとに解説する。最初に，規格制定の経緯を述べ，次に基本概念・データ定義言語・モジュールとデータ操作言語について述べ，最後に，現在審議中のデータベース言語 SQL の拡張[PDAD187][SQL287]について述べる。

データベース言語 *SQL* 規格制定の経緯

関係データベースの標準化の作業は，ANSI (American National Standard Institute) によって始められた。ANSIは，ANSI/X3/SPARCに関係データベース言語の標準化の問題・方向についてのタスクグループを設け，関係データベース言語の標準化の検討を行った。このタスクグループは，1981 年に最終報告をまとめ ANSI/X3 に提出した[ANSI81].これを受けて ANSI は，X3内の X3/H2 で，関係データベース言語の標準化を開始した。因みに，ANSI規格の序では，データベース言語 SQL の標準化プロジェクトの目的を次のように述べている。

> 関係データベース管理システムのインターフェイスの機能(必要な機能のセットと個々の機能の意味)の標準化を計る．これらの機能は，関係データベースの定義，問合せ及び変更に用いられる．

ISO では，1982年から，ISO/ TC97 (Information Processing System)/ SC5 (Programming Language)/ WG5 (Database Language) においてデータベース言語の標準化の検討が開始された。1985年に，SC16 (Open Systems Interconnection) の再編成が行われ，これに伴い SC5 と SC15 (Labelling and File Structure) でのデータベース関連標準化が一括して新しい SC21 (Information Retrieval, Transfer and Management for Open Systems Interconnection) 中の WG3 (Database) に統合された。データベース言語 SQL は，1985年 5 月に国際規格案(Draft Proposal, DP) の登録が行われた。

ISO, ANSI でのデータベース言語の標準化作業は，ネットワークデータベース言語 NDL と関係データベース言語 RDL の両言語標準を同じ概念で統一して開発することをめざした。例えば，モジュールや多くの共通する要素については，現在でも SQL と NDLとの間で統一されている。SCが再編されると機を同じく RDL から SQL への名称変更が行われ，これに伴い，NDLとの統一性の議論は，少し薄らいではいるが，この規格でも，多くの概念・構文要素・構文規則・一般規則において統一化が計られている。

ANSI では，1986年 11 月にデータベース言語 SQL をアメリカ国家規格 (American National Standard, ANS) ANSI X3.135-1986 及び連邦情報処理規格 (Federal Information Processing Standard, FIPS) として交付した。ISOでは，11月に規格原案 (Draft International Standard, DIS) DIS 9075 が郵便投票を通過し，1987 年 2 月に中央事務局に送付された[SQL87]。JISでは，1987年 3 月に JIS 原案の作成を完了している[日本規格協会87]。

基本概念

SQL データベースは，表 (table) を基本とするデータの集まりである。表は，行 (row) と列 (column) からなり，スキーマに含まれる。SQL データベースのデータは，一つ以上の ＜スキーマ (schema)＞ によ

1 ISOでは，SQL という名称は，頭字語ではなく，“SQL”という固有のデータベース言語を指すとの見解が提示されている。

り定義される。＜スキーマ＞では，表，ビュー及び権限を定義する。SQL データベース内の表には，実表とビュー表がある。

表は，行の集まりである。行は，値の並びであり，その表のすべての列の値を含む。行のi 番目の値は，i 番目の列の値である。行は，表に挿入又は表から削除できるデータの最小単位である。SQL データベース中の表では，同じ値をもつ複数の行が存在することを許す。

列 (column) は，値の集まりであり，列の値は，同じデータ型の値である。一つの列の一つの値は，表から選択・更新できるデータの最小単位である。

データ型は，表現可能な値の集合である。SQL データベースのデータ型には，文字列，概数 (approximate numeric) 及び真数 (exact numeric) がある。概数は，計算機内で近似的に表現される数であり，真数は，計算機内で正確に表現される数である。例えば，FORTRAN での概数は，REAL で表現され，真数は，INTEGERで表現される。SQLで指定できるデータ型は，文字列型としては，CHARACTER，真数型としては，NUMERIC，DECIMAL，INTEGER 及び SMALLINT，概数型としては，FLOAT，REAL 及び DOUBLE PRECISION である。

表には，実表 (base table)，導出表 (derived table)，ビュー表 (viewed table)，グループ表 (grouped table) 及びグループビュー表 (grouped view)がある。

実表は，＜表定義 (table definition)＞により定義される名前つきの表とする。実表の記述は，それの名前を含む。導出表は，＜問合せ指定 (query specification)＞の結果，データベース内の一つ以上の表から導出される表とする。導出表の値は，もとの表の値から導出される値とする。ビュー表は，＜ビュー定義 (view definition)＞により定義される名前つきの導出表とする。

グループ表は，＜GROUP BY句＞の結果によって導出されるグループの集合とする。グループビューは，＜ビュー定義＞中の＜問合せ指定＞に＜GROUP BY句＞を含むビュー表とする。

SQL 言語は，表を基本的な対象とするSQLデータベースに対し，構造・整合性制約 (integirity constraints) を定義するスキーマ定義言語 (SQL-DDL) と，データ操作を行うモジュール言語・データ操作言語 (SQL-DML) からなる。標準 SQL では，二つの水準を設定している。水準 2 は，完全な SQL であり，水準 1 は，部分 SQL である。

SQL のデータ操作言語によるデータ操作は，基本的には，データベース中の実表とビュー表から＜問合せ指定＞によって導出される導出表をもとに，データベース内の導出されるもととなった表に対し行われる。

```
SELECT    <値式>，<値式>，．．．
FROM      <表参照>，<表参照>，．．．．
WHERE     <探索条件>
```

図 1: ＜問合せ指定＞の基本的な枠組み

表1，表2，表3 を次のようであるとする。

```
表1          表2          表3
  行1          行1          行1
  行2          行2          行2
  行3          行3          行3
   ・           ・           ・
```

FROM 表1，表2，表3 の結果生じる表4は，次のようになる。

```
表4
  行1    = (表1.行1，表2.行1，表3.行1)
  行2    = (表1.行1，表2.行1，表3.行2)
  行3    = (表1.行1，表2.行1，表3.行3)
                    ・
  行n    = (表1.行1，表2.行1，表3.行n)
  行n+1  = (表1.行1，表2.行2，表3.行1)
                    ・
  行n*m+1=(表1.行2，表2.行1，表3.行1)
                    ・
```

表4の列は，次のようになる。

```
表4
  (表1.列1，表1.列2，....表2.列1，
...表3.列1，...)
```

図 2: 拡張直積

＜問合せ指定＞による導出表の導出は，次の順で行われる。最初に，＜FROM句＞で指定されるデータベース中の一つ以上の表に対し，拡張直積 (extended cartesian product)をとり表を得る。次に，この得られた表に対し，＜WHERE句＞で指定される探索条件を適用する。この結果から，SELECTに続く＜選択リスト＞により指定される＜値

式＞ を列とする表を導出する。 ＜値式＞ には，列名，集合関数，定数や変数等を指定する ＜値指定＞ からなる式を書くことができる。 ＜問合せ指定＞ の基本的な枠組みを次図に示す。

＜FROM句＞ では，データベース中の表を指定する。表の指定は，表名と相関名によって行われる。相関名は， ＜問合せ指定＞ 中で指定される表を一意に指定するために用いられる。すなわち，相関名は，同じ表名が一つの ＜問合せ指定＞ 中で指定された場合にどこで指定した表であるかを識別するために用いられる。

＜FROM句＞ で表1, 表2 及び表3 の三つの表が指定された場合，＜FROM句＞ の拡張直積の結果生じる表4の行は．表1の行の右に表2及び表3の順で行を接続することによってできる。表4の列の順序位置は，表1の列，表2の列，表3の列の順になる。

＜FROM句＞ での拡張直積の結果生じる表に ＜探索条件＞ を適用することによって ＜問合せ指定＞ の結果が得られる。＜探索条件＞ には，＜述語＞ を指定する。SQL で指定できる述語には，比較述語，BETWEEN 述語， IN 述語， LIKE 述語， NULL 述語，限定述語及び EXISTS 述語の五つの述語がある。＜問合せ指定＞では，この五つの述語を組み合わせた論理式を指定することができ，各述語の引数には，＜値式＞ を指定することができる。＜問合せ指定＞ の結果は， ＜FROM句＞ の結果に探索条件の論理式の結果が真となる行のみでできる表に SELECT で指定される ＜選択リスト＞ を適用することによって得られる。また，＜GROUP BY句＞ を指定することにより，導出表をグループ化することを指定することができる。グループ化を指定した場合，グループ表が満たすべき条件は，＜HAVING句＞で指定する。

比較述語は，比較演算子により二つの値の大小比較を指定する。BETWEEN 述語は， BETWEEN の左辺の値が右辺で指定される二つの値の範囲に入っているかどうかを範囲比較を指定する。IN 述語は，指定された値のリストに含まれているかのテストを行う。 LIKE 述語は，文字列のパターン照合比較を指定する。LIKE 述語では，ESCAPE キーワードにより，パターンとして用いられる“%”や“_”等を含めた照合比較ができるようになった。NULL 述語では，ナル値のテストを指定する。限定述語では，＜副問合せ＞ によって指定された値の集合に含まれているかのテストを指定する。EXISTS 述語では，＜副問合せ＞ の結果が空集合であるかのテストを指定する。

SQL では，限定された形式の ＜問合せ指定＞ である ＜副問合せ＞ を用いて複雑な探索条件を記述することができる。＜副問合せ＞ は，＜問合せ指定＞ に比べ，SELECT キーワードの後に ＜値式＞ のリストを指定することはできない。＜副問合せ＞ は，比較述語，IN述語，限定述語及び EXISTS 述語で指定できる。＜副問合せ＞ を用いた探索条件の記述は，英語での表現で用いる関係代名詞と似た形式での表現となっている。

スキーマ定義言語 *(SQL-DDL)*

SQL データベースは，スキーマからなり， ＜スキーマ＞ (CREATE SCHEMA) により定義される。スキーマには，実表，ビュー及び権限が含まれる。＜スキーマ＞ の ＜スキーマ認可句＞で，＜スキーマ認可識別子＞ が指定され，この認可識別子を用いて ＜権限定義＞ によりスキーマ内の実表及びビュー表に対するアクセス制御が行われる。

スキーマ中の実表は， ＜表定義＞ (CREATE TABLE) により定義され，表名により識別される。表名は，認可識別子と表識別子からなる。表には，列と一意性制約を含む。従来の多くの SQL の実装では，一意性制約 (UNIQUE) は，インデックスの定義で行われてきた。標準SQLでは，この一意性制約を ＜表定義＞ 又は ＜列定義＞ 中で行うことができる。

表の列は， ＜列定義＞ により定義される。 ＜列定義＞ では， ＜列名＞ と ＜データ型＞ を定義する。＜列定義＞ には，このほかに NOT NULL と UNIQUE を指定することができる。

一意性制約は， ＜一意性制約定義＞ と ＜列定義＞ の UNIQUE により定義することができる。 ＜列定義＞ での UNIQUE の指定は，主に一つの列に関して表中の行が一意であることを要請する一意性制約を定義し， ＜一意性制約定義＞ は，主に複数の列に関して一意性制約を定義する。

ビュー表は， ＜ビュー定義＞ (CREATE VIEW) により定義される。ビュー表は，＜問合せ指定＞ が実行されたならば生じるであろう仮想的な表である。ビュー表が実際にデータベース内の表として実体化されるかどうかは，各実装による。

データベース内の表に対するアクセス権は， ＜権限定義＞ (GRANT) により設定される。アクセス権の制御は，認可識別子により行われる。ある表に対する権限は，始めは，その表を定義する ＜スキーマ＞ の認可識別子がもつ。その権限は，＜権限定義＞ によって他の認可識別子に付与することができる。＜権限定義＞ 中で WITH GRANT OPTION

が指定された場合，ある表に対する権限は，付与を受けたスキーマの認可識別子によって，別の認可識別子に付与することができる。＜権限定義＞ で指定できる ＜権限＞ には，SELECT, INSERT, DELETE, UPDATE 及び ALL PRIVILEGES を指定することができる。データ操作を行う場合，モジュールの認可識別子に対し，モジュール中で指定するデータ操作が許されているかどうかが検査される。

例えば，供給者-部品データベースの SQL-DDL を用いた定義は，次のとうりである。

```
CREATE SCHEMA
AUTHORIZATION SPDB
CREATE  TABLE SUPPLIER
        (S#     CHAR(6)  NOT NULL
UNIQUE,
         SNAME  CHAR(16),
         LOC    CHAR(10))
CREATE  TABLE PARTS
        (P#     CHAR(6)  NOT NULL,
         COLOR  CHAR(8),
         WEIGHT INTEGER,
         UNIQUE (P#))
CREATE  TABLE SP
        (P#     CHAR(6),
         S#     CHAR(6))
CREATE  VIEW VPARTS (P#, WEIGHT)
        AS SELECT P#, WEIGHT
           FROM   PARTS
           WHERE  WEIGHT >  25
        WITH CHECK OPTION
GRANT   ALL PRIVILEGES  ON  SUPPLIER
        TO  APPL
        WITH GRANT OPTION
GRANT   SELECT, UPDATE  ON  PARTS
        TO  PUBLIC
GRANT   SELECT, INSERT, DELETE  ON
VPARTS
        TO  PAPPL
```

図 3: 供給者-部品データベースの例

モジュール言語とデータ操作言語 *(SQL-DML)*

標準データベース言語 SQL におけるデータ操作は，一つ一つの適用業務プログラム毎に定義されるモジュールをもとに行われる。COBOL, PL/I, FORTRAN, Pascal 等の親プログラムからは，モジュール中の ＜手続き＞ の“呼出し”によって SQL データ操作文 (SQL-DML) が起動させられる。従来から用いられてきた埋込み構文は，モジュール中の ＜手続き＞ の“呼出し”に翻訳され実行される。

SQL 言語で提供されるデータ操作には，検索 (SELECT)，挿入 (INSERT)，削除 (DELETE)，更新 (UPDATE) 及び取出し (FETCH) がある。他に，トランザクションの制御に用いられる コミット (COMMIT) と ロールバック (ROLLBACK) の機能やカーソルを開いたり閉じたりするための OPEN文と CLOSE文がある。

典型的なデータ操作の流れは，次の一連の処理として行われる。

1. モジュール中での ＜カーソル宣言＞ (DECLARE CURSOR) によりカーソルを宣言する。
2. 宣言されているカーソルを開き (OPEN)，
3. カーソルで指定される表の行を取り出し(FETCH)，その行に対し，
4. 親言語での処理を行い，必要であれば，削除 (DELETE) や更新 (UPDATE) を行い，
5. カーソルを閉じる (CLOSE)。

＜カーソル宣言＞ では，＜問合せ指定＞ で指定される導出表の UNION を取り，処理の対象となる表を指定することができる。現在の標準 SQL では，明示的な関係演算については，この UNION のみが提供されており，この UNION の取り扱いについても一般の ＜問合せ指定＞ 中で指定することができず，＜カーソル宣言＞ でのみ指定するようになっており，削除・更新等のデータ操作についても制約がある。

削除・更新・検索処理については，カーソルを用いる行単位の処理方式以外に，SQLデータ操作文で直接＜探索条件＞をWHEREで指定することによって，指定された条件に合致するすべてのデータに対しての処理を行うことができる。

```
MODULE  SPPARTS
  LANGUAGE    FORTRAN
  AUTHORIZATION  SPDB
  DECLARE  CURPARTS CURSOR  FOR
    SELECT P#, WEIGHT/1000, COLOR
    FROM    PARTS P, SUPPRIER S, SP
    WHERE   P.P# = SP.P# AND S.S# =
SP.S#
            AND S.S# = 'ABC LTD'
  PROCEDURE  DBOPEN SQLCODE;
    OPEN CURPARTS;
  PROCEDURE  DBCLOSE SQLCDE;
    CLOSE CURPARTS;
  PROCEDURE  RET PNO CHAR(6) WKG IN-
TEGER
      COLOUR CHAR(8) SQLCODE;
    FETCH CURPARTS INTO PNO, WKG,
COLOUR;
  PROCEDURE  UPD SQLCODE;
    UPDATE PARTS
    SET     COLOR = 'RED'
    WHERE CURRENT OF CURPARTS;
```

図 4: モジュール言語の例

```
    .
EXEC SQL BEGIN DECLARE SECTION  END-
EXEC
      ホスト変数定義
EXEC SQL END DECLARE SECTION  END-
EXEC
    .
EXEC SQL
DECLARE ..  CURSOR  FOR
    .
WHENEVER  SQLERROR GOTO 9090
WHENEVER  NOT FOUND GOTO 1000
END-EXEC
    .
EXEC SQL
FETCH ..
END-EXEC
    .
EXEC SQL
UPDATE ..
END-EXEC
    .
```

図 5: 埋込み SQL 文 の枠組み

*SQL*言語の拡張

ISO での審議過程では，より良い標準を早く(時代遅れにならないように)作成することが求められている。良い標準を作成することと早く出すことは，しばしば両立しないことがある。データベース言語 SQL の標準化にあたっても，幾つかの点で現在の標準に対する問題点や拡張提案があった[Date86b]。そのなかでもとくに，より忠実に関係データモデルを記述できる機能は，ISO 参加各国からの強い支持もあった。ISOでは，データベース言語 SQL の標準化を早期に実現させることとこの機能強化の要請に答えることを両立させるために，補遺 1 の制定を急ぐこととした。

SQL 補遺 1 で求められた関係データモデルをより忠実に表現するための機能とは，次のような機能である。関係データモデルでは，C. J. Date によれば，データオブジェクトとして表と行，列をもち，関係演算をもとにした演算ができ，整合性制約 (integrity constraints) として参照整合性 (referential integrity) と主体整合性 (entity integrity) を満たすことが要請される。

これらの関係データモデルから要請される新しい機能を**整合性拡張機能 (integrity enhancement feature)** として一括し，補遺 1 に含めることにした。主な，機能は，次のとおりである。

1. 一意性制約の機能拡張 (PRIMARY KEY) 及び満たさなければならない表間の参照制約 (FOREIGN KEY)

2. 表の行に適用される検査制約 (CHECK条件)

3. 表の中に行が挿入される時，列の値を指定しないときにとられる値 (DEFAULT句)

一意性制約には，PRIMARY KEYの指定が追加された。PRIMARY KEY は，表の各行を識別するキーを指定し，効果としては，NOT NULL UNIQUE と同じ効果を得られる。

参照制約は，表の各行の指定された一つ以上の列中の値が指定された表のある行の指定された一つ以上の列の値と同じであることを要求するもので，＜列定義＞ と ＜表定義＞で指定することができる。

検査制約は，指定された＜探索条件＞が表のすべての行について真であることを要求する。

データを挿入するときに値を指定しないときにとられる値は，＜DEFAULT句＞　で指定される。＜DEFAULT句＞　では，＜DEFAULT句＞を指定する列に値が挿入されるとき，値が指定されなければ，＜DEFAULT句＞　で指定された　＜定数＞，USER 又は NULL の値が挿入される。

```
CREATE SCHEMA
AUTHORIZATION SPDB
CREATE  TABLE SUPPLIER
        (S#     CHAR(6)   PRIMARY
KEY,
         SNAME  CHAR(16),
         LOC    CHAR(10))
CREATE  TABLE PARTS
        (P#     CHAR(6)   PRIMARY
KEY,
         COLOR  CHAR(8)   DEFAULT
'BLUE',
         WEIGHT INTEGER   CHECK
(WEIGHT > 0),
         UNIQUE (P#))
CREATE  TABLE SP
         PRIMARY KEY (P#, S#)
        (P#     CHAR(6),  FOREIGN
KEY REFERENCES PARTS.P#
         S#     CHAR(6)   FOREIGN
KEY REFERENCES SUPPLIER.S#)
```

図 6: 供給者-部品データベース - 整合性拡張機能を含めたスキーマ定義の例

SQL2　では，補遺1として検討されている項目以外に，SQL言語のより大きな改良及び拡張が検討されている。SQL2 により，現在の標準案における SQL 言語仕様に対する批判に答えようとしている。

- 参照制約定義に動作を追加

補遺 1 で導入された参照制約は，指定された参照制約が満たされない場合には，エラーとすることのみを規定している。SQL2 では，これにエラーの戻りコードを返すだけではなく，このときに有効な動作を追加する。この動作としては，被参照表中の被参照列が削除又は更新されたときに同じ変更を参照表に対して行う CASCADE (波及指定)，被参照表の変更を禁止する RESTRICT，被参照表の変更により参照表の値をナル値，USER 及び　DEFAULT句で指定された値に変更する SET 等の動作の指定が検討されている。

- SQL 言語の言語要素の直交性を高めること

SQL の現仕様では，UNIONを指定したときには制約があり，また，　＜FROM句＞　中に問合せが指定できない等の点が指摘されている。SQL2 では，UNION での制約を除くことや現在の＜問合せ指定＞をより一般化した形式の＜問合せ式＞として扱うことが検討されている。

- データ型の追加やデータ型に対応する演算の追加，特に，各国語やDATETIMEデータ型の追加

日付及び時間をデータ型として導入し，日付や時間の演算の定義を追加することや文字列データ型に対する部分文字列の取り出し・接続等の演算の導入が検討されている。また，日本語を含めた各国語データ型の導入が検討されている。

- 言語接続の追加

ADAとCに対する言語接続の追加が検討されている。

- エラーコードの規格化

現在の　SQLCODE　パラメタは，整数データ型で，正常終了時の値(0)とデータなしの時の値(100) のみが規定されている。それ以外のエラーが起きたときには，ただ負の数が返されるとのみ規定されており，可搬な適用業務プログラムの作成には，十分とはいえない。SQL2では，文字データ型のSQLSTATEパラメタを導入し，一般規則で規定されているエラー及び例外条件に対しより詳細なエラーコードの規格化を検討している。また，　SQLCODE　については，将来の規格では削除されると記載することが検討されている。

- トランザクション処理のための機能拡張

トランザクション処理に対し，より効率的に処理を行うための　＜SET TRANSACTION文＞の導入が検討されている。この　＜SET TRANSACTION文＞　では，READ ONLY 又は READ WRITEを指定することができるようになり，これにより処理系でトランザクションの種類に適したより効率的な扱いが可能になる。

- スクロールカーソル(カーソル位置づけ機能の拡張)

現在の ＜FETCH文＞ は，カーソルを表の次の行に位置づけ，その行の値を取り出す。SQL2では，この ＜FETCH文＞ の位置づけ位置を次の行 (NEXT) だけではなく，前 (PRIOR)・最初 (FIRST)・最後 (LAST) 及び絶対 (ABSOLUTE)・相対 (RELATIVE) 行指示を可能にすることが検討されている。

- その他

その他に，スキーマ情報の表についての規定，定義域についての規定，UNION 以外の集合演算の明示的な指定等の導入が検討されている。

おわりに

データベース言語 SQL は，関係データベース管理システムの言語としての重要性を今後ともより一層増すであろうと考えられる。SQL の最初の規格は，これまでの CODASYL 以来のデータベース言語の標準化の遅滞からの反省により，早期標準化を第一の目標に作業を進めてきた。そのため既存の SQL 言語の実装からかけ離れない形での標準化を目指してきた。一方，標準化の仕事の中では，既存の実装をもとにするだけではなく，今後の実装をガイドすることも重要であると考えられる。SQL 言語の改良及び拡張の努力は，今後 SQL 及び関係データベース管理システムをより良いものにガイドすることを第一に考えた努力と言えよう。とくに，SQL2 での改良及び拡張には，この後者の考え方が色濃く見受けられる。SQL2 の開発には，まだ数年かかることから考えて，わが国でもこの意味で今後ともより積極的な国際標準化での貢献が期待されている。

謝辞

本稿をまとめるにあたり，2年半にわたる ISO 原案の検討・寄書の作成を行っている情報処理学会規格委員会 SC21/WG3 データベース小委員会及び同 SQL アドホックグループのメンバーに感謝します。また，本稿でももとにしているデータベース用語の検討及びJIS原案の作成にあたった日本規格協会情報技術標準化研究センターデータベース言語調査研究委員会及び同 SQL 分科会のメンバーに感謝します。データベース言語 SQL の標準化全般に亘ってご指導いただいている両委員会の委員長筑波大学穂鷹良介教授及び幹事として尽力いただいている日本電信電話株式会社鈴木健司氏には，とくに感謝いたします。

文献

[ANSI 81] ANSI X3 SPARC DBS-SG: Final Report on Relational Database Task Group. ANSI SPARC-81-690 (September 1981).

[Astrahan 76] M. M. Astrahan et al: System R: Relational Approach to Database Management. ACM TODS 1, No. 2 (June 1976).

[Chamberlin 74] D. D. Chamberlin and R. F. Boyce: SEQUEL: A Structured English Query Language. Proc. 1974 ACM SIGMOD Workshop on Data Description and, Access and Control (May 1974).

[Codd 69] E. F. Codd: A Relational Model of Data for Large Shared Data Banks. CACM 13, No. 6 (June 1970).

[Date 86a] C. J. Date: Introduction to Database Systems, Vol. 1, 4th edition. Addison-Wesley (1986).

[Date 86b] C. J. Date: Relational Database. Addison-Wesley (1986).

[日本規格協会 87] データベース言語調査研究委員会: 高度ネットワークのためのプロトコル標準化に関する調査研究(データベース言語調査研究). 日本規格協会情報技術標準化研究センター (昭和 56 年 3 月).

[PDAD1 87] ISO SC21 WG3: Proposed Draft Adenndum 1 to DIS 9075 Database Language SQL. (1987).

[SQL 87] ISO SC21 WG3 N291 (ISO SC21 WG3-DBL RENO-2) Final Draft 9075-1987 (E): Database Language SQL. (November 1986).

[SQL2 87] ISO SC21 WG3 N325: Database Language SQL2. (April 1987).