ISO/IEC JTC1/SC21 WG3 DBL SOU ___
X3H2-94-276

June 1, 1994

I S O

International Organization for Standardization
Organisation Internationale de Normalisation

IEC JTC1/SC21
Information Retrieval, Transfer, and Management for OSI

WG3

Source: U.S.A.

Status: SQL3 Working Paper

Title: TSQL2

Authors: Cliff Leung, Nelson Mattos

References:

1. ISO/IEC JTC1/SC21 WG3 DBL SOU-003 and X3H2-94-080, “(ISO/ANSI working draft) Database Language SQL3”, by Jim Melton (ed.), March 1994

2. Snodgrass, R.T., I. Ahn, G. Ariav, D.S. Batory, J. Clifford, C.E. Dyreson, R. Elmasri, F. Grandi, C.S. Jensen, W. Kaefer, N. Kline, K. Kulkarni, T.Y.C. Leung, N. Lorentzos, J.F. Roddick, A. Segev, M.D. Soo, and S.M. Sripada, “TSQL2 Language Specification,” in: SIGMOD Record, Vol. 23, No. 1, March, 1994, pp. 65--86.

1.0 Introduction

Temporal databases have been an active research area for more than fifteen years. While many database applications intrinsically deal with time-varying data, there has been very little temporal support in the SQL standards. Basically, there is only some predefined temporal data types in SQL92.

With the definition of the SQL/MM project, temporal support has become one of the topics being addressed by the SQL committees. Temporal support has been originally introduced in the spatial part of SQL/MM, primarily to support time-varying data such as the development of an oil spill over time. However, several discussions in the SQL committees have pointed to the fact that temporal support is quite orthogonal to the support of spatial data. Temporal support is useful even in the context of "traditional" applications, as we illustrate in the examples given in the following sections of this paper.

2.0 TSQL2

Following the ARPA/NSF International Workshop on an Infrastructure for Temporal Databases that took place in Arlington, Texas, in June 1993, in which 44 researchers from 10 countries participated, a group of individuals from universities, industry, and public institutions volunteered to develop an extension to the SQL language to support the representation and manipulation of time-varying data. It was a major goal of this group to develop a language that would be an upward compatible extension to the SQL92 standard and that would be relatively easy to implement. For this reason, the extended language was called TSQL2.

In addition to the TSQL2 effort, another group of individuals (some of them also worked on TSQL2) started to focus on a language extension to SQL3 (called TSQL3). The work on TSQL3, however, is yet to get underway, partially due to the fact that SQL3 is still a "moving target".

In March 1994, a preliminary specification of TSQL2 was published in the ACM SIGMOD Record [2]. The language extension, which was designed by a committee comprising members from database vendors, industrial research labs, industrial users, and academia, contains many desired features for temporal databases. However, (as you can see in some of the examples given in this paper) the language still has some incompatibilities with SQL92.

3.0 Purpose of this working paper

The purpose of this paper is to bring to the ANSI committee's attention the TSQL2 work.

It is our belief that we could take most (if not all) of the work done in TSQL2 as the basis to extend SQL in this direction. This would provide the features being currently required by the spatial part of SQL/MM.

In a conversation that I had with Rick Snodgrass (the chairman of the TSQL2 committee) at the last SIGMOD conference, he expressed the interest in working with the SQL and SQL/MM committees to see how the TSQL2 work could be integrated as part of a future generation of the SQL standard. The TSQL2 committee is *"committed to bringing TSQL2 into compliance with SQL92, and is actively working on that now"*. It is also willing to provide input to the SQL/MM to help them define the semantics of time.

4.0 Some Examples of TSQL2 features

In this section, we provide a suite of examples to illustrate some of the features of TSQL2. As mentioned above, TSQL2 still has some incompatibilities with SQL92. Therefore, some language constructs and/or syntax used in the following examples may not be adequate from the point of view of the SQL standard.

We will illustrate the semantics of two different notions of time: valid time and transaction time. Valid time concerns the time when a fact is true in the enterprise being modeled in the database, while transaction time concerns the "history" of a fact in the database.

4.1 Valid Time Support

The valid time of a tuple is the time when the fact being represented by that tuple is true in the modelled reality. For example, we may want to store the information regarding an employee Ben in the Sales department with $30K salary during the period from 1/1/89 to 3/31/93, and in the Development department with $100K during 4/1/93 to 6/1/94. This kind of temporal information can be represented in SQL-92, but not all the queries that an application would like to ask can be expressed directly in SQL-92. Typical queries in this type of database include:

- **Who worked in Sales department as of 2/1/94?**
- **When was a person Ben hired in Sales department?**
- **Who was on the payroll for more than 5 years?**

While the first two queries can be easily expressed in SQL-92, the third one cannot. Even with help of SQL3 features, the query would be very difficult to express. In TSQL2, one can create employee and department tables which store the associated valid time information as follows:

```
create table Employee
(Empname char,
Department char,
Salary integer) as valid
```

Here we assume the column "Empname" is the primary key of the employee table. The new reserved word "valid" is an extension to SQL-92 for valid time support. Conceptually, one can think of an implicit timestamp column being associated with the above table to store the valid time information for each tuple.

Example 1.1

To insert a tuple regarding the above employee into the employee table:

```
insert into Employee
values (Ben, Sales, 30000)
valid period'1989-01-01 - 1993-03-31'
```

Period is a new predefined data type.

Example 1.2

To find out who worked in Sales department as of 2/1/93:

```
select snapshot Empname
from Employee e
where Department = 'Sales' and valid(e) contains date'1993-01-01'
```

Here "valid(e)" is a builtin function, in functional style, which extracts the valid time value from the tuple. The predicate "valid(e) contains date'1993-01-01'" is true if the valid time starts earlier than and ends later than February 1, 1993. "snapshot" is a new reserved word indicating that a non-time-varying table is to be returned.

Example 1.3

To find out when Ben was hired in Sales department:

```
select Empname
valid begin (valid (e))
from Employee e
where Department = 'Sales' and Empname = 'Ben'
```

"valid" is a new clause which specifies when the resulting tuple is considered to be valid. "begin" extracts the first datetime of a period.

Example 1.4

To find out who has/had been on the payroll (i.e., in any department) for more than five years, and for how long:

```
select Empname, interval (valid(e))
from Employee(Empname) e
where interval (valid(e)) > interval '5' year
```

"interval" is a function, which returns the length of the argument period, returning (logically) an interval. "Employee(Empname)", an overloaded name, is equivalent to projecting only the specified column ("Empname" in this example) and the implicit timestamp column, and "coalesce" the timestamp column (i.e., two overlapping intervals are "merged" into a single interval). Note that an employee could be in the same department with different salaries and thus there may be more than one tuple for the same person in the table. Note that expressing this kind of queries is not possible in SQL-92 (because the "coalesce" operation.)

4.2 Transaction Time Support

The transaction time of a database fact is the time the fact is current in the database and may be retrieved. Typical use of transaction time includes audit-trail applications, e.g., who did the insertion of a tuple into the database and when. SQL-92 does not support transaction time.

Transaction time support is orthogonal to valid time in that one can create tables with either valid time or transaction time support or both simultaneously. For example, one can create an employee table with both valid time and transaction time:

```
create table Employee
(Empname char,
Department char,
Salary integer) as valid and transaction
```

Example 2.1

To find out when the fact that "Ben was hired in Sales department" was stored in the database:

```
select begin (transaction(e) )
from Employee (Empname, Department) e
where Department = 'Sales' and Empname = 'Ben'
```

Like "valid(e)", "transaction(e)" is a builtin function which extracts the transaction time value from the tuple.

Example 2.2

Suppose we want to list the employee information that was "incorrectly" recorded on 1/1/92 (but corrected now):

```
select *
from Employee e1, Employee e2
where  transaction(e1) overlaps date ‘1992-01-01’
       and transaction(e2) overlaps current_date
       and e1.Name = e2.Name
       and valid(e1) overlaps valid(e2)
       and (e1.Salary <> e2.Salary or e1.Department <> e2.Department)
```

“overlaps” is another builtin operator similar to “contains” operator discussed above.

Example 2.3

List all employee tuples where the update was pre- or post-dated by more than 5 days:

```
select *
from Employee e
where interval(period(begin(transaction(e)), begin(valid(e)))) > interval ‘5’ day
```

“period” is a function that constructs a period out of two datetimes. “begin” is a builtin function which extracts the starting time.

5.0 TSQL2 Status

A draft specification of the language was released to the database community in March, 1994 [2]. The language is now being revised so that it will be entirely upward compatible with SQL92. It is expected that the language design will be completed later this summer. A set of 26 commentaries will be released with the language specification. These commentaries motivate specific design decisions, discuss previous approaches, provide the semantics of the constructs, and give examples. A tutorial is also planned.