ISO/IEC JTC 1/SC 32 N 1952a

Date: 2010-01-10

REPLACES: —

ISO/IEC JTC 1/SC 32

Data Management and Interchange

Secretariat: United States of America (ANSI)
Administered by Farance Inc. on behalf of ANSI

DOCUMENT TYPE	Summary of Voting/Table of Replies
TITLE	Summary of Voting on 32N1922 DCOR 9075-2:2008/Cor1:2009 - Information technology - Database languages - SQL - Part 2: Foundation (SQL/Foundation)
SOURCE	SC32 Secretariat
PROJECT NUMBER	1.32.03.06.02.00
STATUS	WG3 is requested to resolve the comments. The document obtained substantial support. REVISED 2010-02-02 TO ADD UK VOTE
REFERENCES	
ACTION ID.	ACT
REQUESTED ACTION	
DUE DATE	
Number of Pages	9
LANGUAGE USED	English
DISTRIBUTION	P & L Members SC Chair WG Conveners and Secretaries

Dr. Timothy Schoechle, Secretary, ISO/IEC JTC 1/SC 32
Farance Inc *, 3066 Sixth Street, Boulder, CO, United States of America
Telephone: +1 303-443-5490; E-mail: Timothy@Schoechle.org
available from the JTC 1/SC 32 WebSite http://www.jtc1sc32.org/
*Farance Inc. administers the ISO/IEC JTC 1/SC 32 Secretariat on behalf of ANSI

ISO/IEC JTC 1/SC 32 N1952a

Summary of Voting on Document SC 32 N 1922

Title: Summary of Voting on 32N1922 DCOR 9075-2:2008/Cor1:2009 - Information technology - Database languages - SQL - Part 2: Foundation (SQL/Foundation)

Project: 1.32.03.06.02.00

"P" Member	Approval	Approval with Comments	Disapproval with Comments	Abstention with Comments
Australia				1
Belgium				
Canada	1			
China				
Czech Republic	1			
Egypt				
Finland				
Germany	1			
India				
Japan	1			
Korea, Republic of	1			
Sweden				
United Kingdom	1			
United States			1	
Total "P"	**6**	**0**	**1**	**1**
"O" Member				
Austria				
France				
Indonesia				
Italy				
Kazakhstan				
Netherlands, The				
Norway				
Portugal				
Romania				
Russian Federation				
Switzerland				
Total "O"				

Dr. Timothy Schoechle, Secretary, ISO/IEC JTC 1/SC 32
Farance Inc *, 3066 Sixth Street, Boulder, CO, United States of America
Telephone: +1 303-443-5490; E-mail: Timothy@Schoechle.org
available from the JTC 1/SC 32 WebSite http://www.jtc1sc32.org/
*Farance Inc. administers the ISO/IEC JTC 1/SC 32 Secretariat on behalf of ANSI

COMMENTS:

Australia

Abstain. Lack of expertise and interest:

United States

No. See attached comment (below):

Title: Recursive window queries

Author: Fred Zemke
Source: U.S.A.
Status: Change proposal
Date: December 13, 2009

Abstract

This paper points out a problem with allowing window queries in the recursive portion of recursive queries. It responds to an [SQL:2008 TC] comment.

References

[Foundation:1999] Jim Melton (ed), "ISO International Standard (IS) Database Language SQL - Part 2: SQL/Foundation", ISO/IEC 9075-2:1999

[Foundation:2008] Jim Melton (ed), "ISO International Standard (IS) Database Language SQL - Part 2: SQL/Foundation", ISO/IEC 9075-2:2008

[Foundation CD] Jim Melton (ed), "Committee Draft (CD) Database Language SQL - Part 2: SQL/Foundation", ISO/IEC JTC1/SC32 WG3:CJU-003 = ANSI INCITS H2-2009-00007

[Foundation IWD] Jim Melton (ed), "Informal Working Draft (IWD) Database Language SQL - Part 2: SQL/Foundation", ISO/IEC JTC1/SC32 WG3:LCY-003 = ANSI INCITS DM32.2-2009-00097

[SQL:2008 TC] Stephen Cannan (ed.), "Draft Technical Corrigendum", ISO/IEC JTC1/SC32 WG3:LCY-011 = ANSI INCITS H2-2009-00999

[MCI-077] Shel Finkelstein et.al., "Expressing recursive queries in SQL", ISO/IEC JTC1/SC21 WG3 DBL:MCI-077 = ANSI X3H2-96-075r1

1. Discussion

This paper responds to the following DCOR ballot comment:

> severity: 1 - major technical
>
> reference: P02-07.13, <query expression>
>
> description:

> Window functions should be prohibited in the recursive portion of recursive queries.

[MCI-077] introduced the model for recursive queries in the standard since [Foundation:1999]. The model adopted is based on fixpoint semantics, which requires that iterations in the recursion must be monotonically increasing. This is stated in section 3.2 “Fixpoint semantics” on page 26 of that paper as follows:

> It has been shown that fixpoint theory assures that unique solutions really exist under the following:
>
> > *the transformation on the right hand side of a recursive definition must be monotonically increasing.*
>
> That is, adding elements to the input set will not remove or change an element of the result of the transformation.

The point is amplified in section 3.3.2 “Grouping and aggregation” page 30 as follows:

> As we have seen, problems are to be expected during computation of recursively defined tables if elements being added to the result affect elements collected so far. In the negation example this had the effect that elements were completely removed. Equally, already adopted elements should not be changed by newly adopted elements.

Section 3.3.2 “Grouping and aggregation” then discusses the problems that aggregation causes for recursion. Essentially, the problem is that one iteration of recursion might compute one set of values for an aggregate, but a later iteration in the recursion might add more rows, changing the contents of groups and therefore the values of aggregates. Thus it is impossible at any stage in the recursion to know the final group and the final value of aggregates, yet the value of the aggregates may determine what new rows are added at each stage of the recursion.

The solution in [MCI-077] is a prohibition on aggregates in the recursive step of recursive queries. The paper expresses it this way:

> Aggregation shall be applied only to those tables which are completely known prior to this operation [i.e., the recursion].

This principle is summarized in the phrase “aggregation shall not cross recursion”.

When windows and window functions were added in [Foundation:2003], we forgot to consider the implications for recursive queries. Window functions are a type of aggregate; indeed, window functions arre a generalization of aggregates, and with a little (unnatural) monkeying around, one can use a window function to compute an aggregate. For example:

```
SELECT DISTINCT Sausage, Egg
FROM ( SELECT Sausage, COUNT(*) OVER Easy AS Egg
       FROM T
       WINDOW Easy AS ( PARTITION BY Sausage
                        ROW BETWEEN UNBOUNDED PRECEDING
```

```
                                        AND UNBOUNDED FOLLOWING )
        ) AS X
```

The preceding query (using a <window function>) is equivalent to the more common query (using <set function specification>)

```
SELECT Sausage, COUNT(*) AS Egg
FROM T
GROUP BY Sausage
```

Aggregates are prohibited in the recursive portion of recursive queries; therefore <window function>s should be as well.

During discussion in the US national body (DM32.2) it was pointed out that it is not merely aggregate window functions that are problematic for recursion; in fact, all window functions are. This is because of the principle cited above, "... already adopted elements should not be changed by newly adopted elements". If window functions are permitted in recursion, then new rows will be added to window partitions, which potentially changes the result of rank functions (RANK, etc.), ROW_NUMBER, NTILE, LEAD, LAG, FIRST_VALUE, LAST_VALUE, and NTH_VALUE.

The rule that prohibits aggregates in recursive queries is 7.13 <query expression> 3)i)iii)4):

> 4) WQE_i shall not contain a <query specification> *QS* such that:
>
> A) *QS* immediately contains a <table expression> *TE* that contains a <query name> referencing WQN_j, and
>
> B) *QS* immediately contains a <select list> *SL* or *TE* immediately contains a <having clause> *HC* and *SL* or *TE* contain a <set function specification>.

Window functions can only appear in <select list> or <order by clause>. The latter is transformed away (6.10 <window function> SR 3)a)). Thus the proposed rule is

> **4.1) WQE_i shall not contain a <query specification> *QS* such that:**
>
> **A) *QS* immediately contains a <table expression> *TE* that contains a <query name> referencing WQN_j, and**
>
> **B) *QS* immediately contains a <select list> *SL* that contains a <window function>.**

2. Proposal conventions

This proposal uses the following conventions:

1. SMALLCAPS	denote numbered editorial instructions;
~~blue strikeout~~	denotes existing text to be deleted;

bold red	denotes new text to be inserted;
plain	denotes existing text to be retained;
[Note:...]	brackets enclose italicized notes to the proposal reader;
⌊_____⌋	boxes surround "editing tags," which are part of the document (not instructions to the editor) and may be deleted, inserted, modified or retained, depending on the typeface within the box.

3. Proposal for [Foundation IWD]

3.1 Changes to 7.13 <query expression>

1. EDIT SYNTAX RULE 3)I)III)4) AS FOLLOWS:

3) If <with clause> is specified, then:

a) ...

i) For a potentially recursive <with list> *WL* with *n* elements, and for *i* ranging from 1 (one) to *n*, let WLE_i be the *i*-th <with list element> of *WL*, let WQN_i be the <query name> immediately contained in WLE_i, let WQE_i be the <query expression> simply contained in WLE_i, let WQT_i be the table defined by WQE_i, and let *QNDG* be the query name dependency graph of *WL*.

i) ...

iii) For each *WLEi*, for *i* ranging from 1 (one) to *n*, and for each *WQNj* that belongs to the stratum of *WQEi*:

1) ...

4) WQE_i shall not contain a <query specification> *QS* such that:

A) *QS* immediately contains a <table expression> *TE* that contains a <query name> referencing WQN_j, and

B) *QS* immediately contains a <select list> *SL* or *TE* immediately contains a <having clause> *HC* and *SL* or *TE* contain a <set function specification>.

4.1) WQE_i shall not contain a <query specification> *QS* such that:

A) *QS* immediately contains a <table expression> *TE* that contains a <query name> referencing WQN_j, and

B) *QS* immediately contains a <select list> *SL* that contains a <window function>.

NOTE nnn: If a <window function> is contained in an <order by clause>, then the syntactic transformation in this Subclause that moves the <window function> to a <select sublist> is effectively applied before applying this rule.

4. Proposal for [SQL:2008 TC]

4.1 Corrections to [Foundation:2008] 7.13 <query expression>

1. ADD THE FOLLOWING CORRIGENDUM ITEM:

Rationale: window functions, like aggregates, must be prohibited in the recursive portion of recursive queries.

Source: WG3:KMG-022

Insert the following after SR 2)g)iii)4):

4.1) *WQE*$_i$ shall not contain a <query specification> *QS* such that:

A) *QS* immediately contains a <table expression> *TE* that contains a <query name> referencing *WQN*$_j$, and

B) *QS* immediately contains a <select list> *SL* that contains a <window function>.

5. Checklist

Concepts	n/a
Access Rules	n/a
Conformance Rules	n/a
Lists of SQL-statements by category	n/a
Table of identifiers used by diagnostics statements	n/a
Collation derivation for character strings	n/a
Closing Possible Problems	n/a
Any new Possible Problems clearly identified	none
Reserved and non-reserved keywords	n/a
SQLSTATE tables and Ada package	n/a
Information and Definition Schemas, including short-name views	n/a
Implementation-defined and –dependent Annexes	n/a
Incompatibilities Annex	handled through the TC
Embedded SQL and host language implications	n/a
Dynamic SQL issues: including descriptor areas	n/a
CLI issues	n/a

- End of paper -