March 5, 2021

ISO/IEC JTC1/SC32/WG3

ANSI INCITS DM32

Database

Title: Declared type for <boolean literal> and <boolean value expression>

Author: Gerald Venzl

Project: SQL: 202X-CD

Status: Proposal

References:

[Framework] WG3:W05-002 (DM32-2020-00411) *(ISO Committee Draft) Database Language SQL - Part 1: Framework (SQL/Framework),* Jim Melton (editor), November 2020.

[Foundation] WG3:W05-003 (DM32-2020-00412) *(ISO Committee Draft) Database Language SQL - Part 2: Foundation (SQL/Foundation),* Jim Melton (editor), November 2020.

[PSM] WG3:W05-004 (DM32-2020-00413) *(ISO Committee Draft) Database Language SQL - Part 4: Persistent Stored Modules (SQL/PSM),* Jim Melton (editor), November 2020.

[Schemata] WG3:W05-005 (DM32-2020-00414) *(ISO Committee Draft) Database Language SQL - Part 11: Information and Definition Schemas (SQL/Schemata),* Jim Melton (editor), November 2020.

[XML] WG3:W05-006 (DM32-2020-00415) *(ISO Committee Draft) Database Language SQL - Part 14: XML-Related Specifications (SQL/XML),* Jim Melton (editor), November 2020.

[PGQ] WG3:W05-007 (DM32-2020-00416) *(ISO Committee Draft) Database Language SQL - Part 16: Property Graph Queries (SQL/PGQ)*, Jim Melton (editor), November 2020.

[CD comments] WG3:W09-009 (DM32-2021-00077), Jim Melton (editor)

1 Introduction

This paper addresses the comment:

> #35 - P02-USA-070
> 2-Minor Technical
> P02-05.03, <literal>
> There is no (syntax) rule that establishes the declared type of a <boolean literal>.
>
> Solution: None provided with comment.

1.1 Rationale

Since its introduction in the ISO SQL 1999 standard, <boolean literal> has failed to provide a syntax rule for the declared type.

During the investigation it was also found that the same applies to 6.44 <boolean value expression> which also fails to provide a syntax rule for the declared type.

This paper proposes changes to close the ballot comment and address the newly identified problem as well.

Additionally, while researching both items above it became apparent to the author of this proposal that there are inconsistencies with the naming of Boolean type itself. The proposal lists these inconsistencies but does not recommend any further action on them. It is left up to the editor to decide the right course of action to resolve these inconsistencies.

1.2 Background

1.2.1 <boolean literal>

The Syntax Rules of section 5.3 <literal> specify declared types for:

1. <character string literal> (SR 17)
2. <binary string literal> (SR 20)
3. <exact numeric literal> (SR 29)
4. <approximate numeric literal> (SR 30)
5. <date literal> (SR 33)
6. <time literal> (SR 34)
7. <timestamp literal> (SR 35)
8. <interval literal> (SR 38)

Furthermore, it is specified by SR 8 that a <national character string literal> is equivalent to a <character string literal> and by SR 13 that a <Unicode character string literal> is equivalent to a <character string literal>.

These syntax rules cover the declared type for all but <boolean literal> definitions as follows:

<literal>
    <signed numeric literal>
        <unsigned numeric literal>
            <exact numeric literal> → SR 29
            <approximate numeric literal> → SR 30

    <general literal>
        <character string literal> → SR 17
        <national character string literal> → SR 8
        <Unicode character string literal> → SR 13
        <binary string literal> → SR 20
        <datetime literal>
            <date literal> → SR 33
            <time literal> → SR 34
            <timestamp literal> → SR 35
        <interval literal> → SR 38
        <boolean literal> → **missing SR**

As the values of a <boolean literal> are defined as TRUE | FALSE | UNKNOWN one cannot just assume that the declared type of such a literal should be of type Boolean but it has to be explicitly specified. This is what the comment points out and what this proposal implements by introducing a syntax rule that establishes that the declared type of <boolean literal> is of type Boolean.

1.2.2 <boolean value expression>

In second 6.28 <value expression>, Syntax Rule 1 specifies "*The declared type of a <value expression> is the declared type of the simply contained <common value expression>, <boolean value expression>, or <row value expression>.*"

While the <common value expression> and <row value expression> both provide syntax rules for the respective declared types, the <boolean value expression> does not.

The issue with the lack of a declared type for <boolean value expression> can be observed when evaluating the <value expression> and its dependents:

<value expression>
    <common value expression> → SR 2
    <boolean value expression> → **missing SR**
    <row value expression> →
        7.2 <row value expression>, SR 2

1.2.3 Inconsistent naming of Boolean, boolean, and BOOLEAN

During the research for 1.2.1 and 1.2.2 it became apparent that there are inconsistencies between the spelling of Boolean throughout [Foundation]:

1. “boolean”
    1. 4.6 Boolean types
    2. 4.8.3 Operations involving JSON values
    3. 5.3 <literal> GR 14 (Note 131)
    4. 6.1 <data type>
        1. SR 48
        2. GR 13
    5. 6.13 <cast specification>
        1. GR 11 f)
        2. GR 12 f)
        3. GR 21
        4. GR 21 b)
    6. 6.27 <JSON value function> SR 2
    7. 6.39 <JSON scalar> SR 2
    8. 6.44 <boolean value expression>
        1. SR 1
        2. GR 1
        3. GR 3
    9. 7.11 <JSON table> SR 1 e) ii)
    10. 7.18 <search or cycle clause>
        1. SR 3 b) ii) 3)
        2. CR 1
    11. 8.1 <predicate>
    12. 8.2 <comparison predicate> GR 8
    13. 9.1 Retrieval assignment
        1. SR 3
        2. GR 7 p)
    14. 9.2 Store assignment
        1. SR 3
        2. GR 3 b) xviii)
    15. 9.5 Result of data type combinations SR 3 f)
    16. 9.40 SQL/JSON path language: syntax and semantics
        1. GR 11 b) i) 2) A)
        2. GR 11 g) ii) 6) C) I) 1) d)
    17. 10.9 <aggregate function> SR 7 e)
    18. 11.5 <default clause>
        1. SR 4 a) vii)
        2. GR 2 a) vii)
    19. Annex A, SQL Conformance Summary 213) a) i)
    20. Annex B, Implementation-defined elements 83) a)
    21. Annex C, Implementation-dependent elements
        1. 80 a)
        2. 81 a)

2. “BOOLEAN”

    1. 4.2.2 Naming of predefined types
    2. 4.6 Boolean types
    3. 4.39 Host languages
    4. 6.1 <data type>
        1. SR 48
        2. GR 13
    5. 5.2 <token> and <separator>
    6. 6.44 <boolean value expression> CR 1
    7. 9.3 Passing a value from a host language to the SQL-server GR 5 e)
    8. 9.4 Passing a value from the SQL-server to a host language GR 5 g)
    9. 9.7 Type precedence list determination SR 15
    10. 9.9 Type name determination SR 2 t)
    11. 11.4 <column definition> GR 6 e) (Note 526)
    12. 11.7 <unique constraint definition> SR 6 e) i) (Note 532)
    13. 11.8 <referential constraint definition> SR 18 c) (Note 536)
    14. 11.9 <check constraint definition> SR 9 (Note 539)
    15. 11.19 <alter column data type clause> SR 15 j)
    16. 11.65 <user-defined ordering definition> GR 1 c)
    17. 13.3 <externally-invoked procedure> SR 10 e)
    18. 13.5 Data type correspondences
        1. Table 19 – Data type correspondences for Ada
        2. Table 20 – Data type correspondences for C
        3. Table 21 – Data type correspondences for Cobol
        4. Table 22 – Data type correspondences for Fortran
        5. Table 23 – Data type correspondences for M
        6. Table 24 – Data type correspondences for Pascal
        7. Table 25 – Data type correspondences for PL/I
    19. 20.1 Description of SQL descriptor areas
        1. SR 6 j)
        2. SR 7 p)
        3. GR 1 Table 29 – Codes used for SQL data types in Dynamic SQL
    20. 20.7 <prepare statement> GR 5 a) xii)
    21. 21.3 <embedded SQL Ada program>
        1. SR 4 a) vii)
        2. SR 4 b) iv)
    22. 21.6 <embedded SQL Fortran program> SR 5 f)
    23. 21.8 <embedded SQL Pascal program> SR 4 e)
    24. 25.3 Implied feature relationships of SQL/Foundation, Table 40 — Implied feature relationships of SQL/Foundation Feature, ID T133
    25. Annex F, SQL feature taxonomy, Table 44 — Feature taxonomy for optional features, line item 232

3. “Boolean”

    1. 3.6.21 JSON Boolean
    2. 4.3.3.3 Other operators involving character strings
    3. 4.6.1 Introduction to Boolean types
    4. 4.48.1 Introduction
    5. 4.48.3 SQL/JSON data model
    6. 4.48.4 SQL/JSON functions
    7. 6.13 <cast specification> SR 7
    8. 6.33 <JSON value constructor>
        1. GR 2 d) ii) 1) F) IV) 1)
        2. GR 3) c) ii) 1) B) IV) 1)
        3. GR 4) b) ii) 1) B) IV) 1)
    9. 10.11 <JSON aggregate function>
        1. GR 4 b) ii) 2) D) IV) 1)
        2. GR 5 f) ii) 2) D) I)
    10. 10.12 <JSON input expression> SR 3
    11. 13.3 <externally-invoked procedure> SR 10 e)
    12. Annex B, Implementation-defined elements 55) e)
    13. Annex B, Implementation-defined elements 91) c)

4. “Boolean-type”

    1. 9.3 Passing a value from a host language to the SQL-server GR 5 e) v) (Note 355)
    2. 9.4 Passing a value from SQL-server to a host language GR 5 g) v) (Note 359)

A search throughout the other parts of the standard provided the following insights:

1. No inconsistencies in Framework
2. No inconsistencies in PSM
3. Schemata:
    1. 6.7 YES_OR_NO domain 1) lowercase “boolean”
4. XML
    1. 8.1 <predicate> lowercase “boolean”
5. No inconsistencies in PGQ

This proposal won’t make any further comments on the inconsistencies and leaves it to the editor to decide what the right course of action for this would be.

1.3 Conventions

SMALLCAPS	denote numbered editorial instructions;
New Text	New text is identified by using a red bold font;
~~Deleted Text~~	Deleted text is identified by using a blue strikethrough font;
Plain	denotes existing text to be retained;
[Note]	denotes notes to the editor.

2 Proposal

2.1 Proposed changes for [Foundation]

2.1.1 Changes to 5.3 <literal>

1. ADD A NEW SYNTAX RULE AS SHOWN HERE

   N) The declared type of a <boolean literal> is Boolean.

   [Note to editor: please assign an appropriate number for N]

2.1.2 Changes to 6.44 <boolean value expression>

1. ADD A NEW SYNTAX RULE AS THE VERY FIRST SYNTAX RULE AS SHOWN HERE

   0.1) The declared type of a <boolean value expression> is Boolean.

3 Checklist

Interactions with other concurrent proposals identified and editorial assistance given	N/A
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 coercibility for character strings	N/A
Closing Possible Problems	N/A
Any new Possible Problems clearly identified	N/A
Reserved and non- reserved keywords	N/A
SQLSTATE tables and Ada package	N/A
Information and Definition Schemas	N/A
Implementation- defined and –dependent Annexes	N/A
Incompatibilities Annex	N/A
Embedded SQL and host language implications	N/A
Dynamic SQL issues: including descriptor areas	N/A
PSM impact	None
Schemata impact	None
XML impact	None
PGQ impact	None

- End of paper -