Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation)

TECHNICAL CORRIGENDUM 1

*Technologies de l'information — Langages de base de données — SQL —
Partie 2: Fondations (SQL/Fondations)*

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 9075-2:2016 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

ICS 35.060

Ref. No. ISO/IEC 9075-2:2016/Cor.1:2019(E)

Published in Switzerland

2   Normative references

2.2   Other international standards

1.   *Rationale: Replace invalid URI.*

     Replace the normative reference for RFC 7159 with:

> [RFC7159] Internet Engineering Task Force, RFC 7159, *The JavaScript Object Notation (JSON) Data Interchange Format*, March 2014; https://datatracker.ietf.org/doc/rfc7159/

4   Concepts

4.23   Integrity constraints

4.23.3 Table constraints

4.23.3.2 Unique constraints

1.   *Rationale: Clarify uniqueness in the presence of nulls.*

     Replace the 2nd, 3rd and 4th paragraphs with:

     Let *T* be a table and let *R1* and *R2* be two rows of *T*. If there is some column for which the corresponding values from *R1* and *R2* are not equal, *R1* and *R2* are *unique with nulls distinct*. If there is some column for which the corresponding values from *R1* and *R2* are distinct, *R1* and *R2* are *unique with nulls not distinct*. If all column values of both *R1* and *R2* are the null value, it is implementation-defined whether *R1* and *R2* are unique with nulls distinct, or unique with nulls not distinct. It is implementation-defined whether the *implementation uniqueness rule* is unique with nulls distinct or unique with nulls not distinct.

     If the table descriptor for base table *T* includes a unique constraint descriptor indicating that the unique constraint was defined with PRIMARY KEY, then the columns of that unique constraint constitute the *primary key* of *T*. A table that has a primary key cannot have a proper supertable.

     Let *T* be a base table and let *R1* and *R2* be two rows of *T*. Let *UC* be a unique constraint on *T*. If *T* is a system-versioned table, let *R1* and *R2* be two current system rows of *T*. If *UC* includes a <without overlap specification> *WOS*, let *ATPN* be the <application time period name> contained in *WOS*; *R1* and *R1* must additionally be such that the *ATPN* period values of *R1* and *R2* overlap. Let *UR1* and *UR2* be rows containing only the unique columns of *R1* and *R2* respectively.

     UC is satisfied if and only if for every such *UR1* and *UR2*, the implementation uniqueness rule holds.

4.39   SQL-statements

4.39.5 SQL-statement atomicity and statement execution contexts

This Subclause is modified by Subclause 4.10.4, "SQL-statement atomicity and statement execution contexts", in ISO/IEC 9075-4.

1. *Rationale: Clarify delta tables.*

    Add the following bullet to the bullet list of the 10th paragraph:

    — A set of old delta tables and new delta tables.

2. *Rationale: Correct the algorithm for CHECK OPTION enforcement.*

    Add the following bullet to the bullet list of the 10th paragraph:

    — A set of views to be checked.

6   Scalar expressions

6.10   <window function>

1. *Rationale: <ntile function> must be computed over the entire window partition*

    Replace General Rule 1) a) ii) 3) C) with:

    1)   ...

        a)   ...

            ii)   ...

                3)   ...

                    C)   Let *WDX1* be the window structure descriptor that describes the window defined by the <window specification>

```
( WNS RANGE BETWEEN UNBOUNDED PRECEDING

          AND UNBOUNDED FOLLOWING )
```

                    C.1) Let *T* be the collection of rows in the window frame of *R* as defined by *WDX1*, as specified by the General Rules of Subclause 7.15, "<window clause>".

2. *Rationale: Determine the correct sizes of the subdivisions of the window partition used by <ntile function>.*

Replace General Rule 1) a) ii) 3) E) II) with:

1) ...

    a) ...

        ii) ...

            3) ...

                E) ...

                    II) Otherwise, for each i, 1 (one) $\leq i \leq$ MOD(CT, NT), let NQ_i be CEILING(CAST(CT AS REAL) / NT), and for each i, MOD(CT, NT) $< i \leq NT$, let NQ_i be FLOOR(CAST(CT AS REAL) / NT).

3. *Rationale: <lead or lag function> (particularly LEAD) must be computed over the entire window partition.*

Replace General Rule 1) b) ii) with:

1) ...

    b) ...

        ii) Let *WDX1* be the window structure descriptor that describes the window defined by the <window specification>

```
( WNS RANGE BETWEEN UNBOUNDED PRECEDING

           AND UNBOUNDED FOLLOWING )
```

        ii.1) Let *T* be the collection of rows in the window frame of *R* as defined by *WDX1*, as specified by the General Rules of Subclause 7.15, "<window clause>".

4. *Rationale: <lead or lag function> (particularly LEAD) must be computed over the entire window partition.*

Replace General Rule 1) b) iii) 1) with:

1) ...

    b) ...

        iii) ...

            1) Case:

                A) If *NTREAT* is RESPECT NULLS, then let *TX* be the sequence of values that is the result of applying *VE1* to each row of *T* that follows the current row, ordered according to the window ordering of *WDX1*.

                B) Otherwise, let *TX* be the sequence of values that is the result of applying *VE1* to each row of *T* that follows the current row and eliminating null values, ordered according to the window ordering of *WDX1*.

6.12   <case expression>

This Subclause is modified by Subclause 6.4, "<case expression>", in ISO/IEC 9075-14.

1. *Rationale: Use correct terminology.*

   Replace Syntax Rule 2) h) with:

   2)   ...

   h)   If <else clause> is specified, then let *CEEC* be that <else clause>; otherwise, let *CEEC* be the zero-length character string.

6.32   <string value function>

This Subclause is modified by Subclause 6.4, "<string value function>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 6.8, "<string value function>", in ISO/IEC 9075-14.

1. *Rationale: Correct an error in the rule structuring.*

   Replace General Rule 6) with:

   6)   If <regular expression substring function> is specified, then let *C* be the result of the first <character value expression>, let *R* be the result of the second <character value expression>, and let *E* be the result of the <escape character>.

   Case:

   a)   If at least one of *C*, *R*, and *E* is the null value, then the result of the <regular expression substring function> is the null value.

   b)   If the length in characters of *E* is not equal to 1 (one), then an exception condition is raised: *data exception — invalid escape character*.

   c)   If *R* does not contain exactly two occurrences of the two-character sequence consisting of *E*, each immediately followed by <double quote>, then an exception condition is raised: *data exception — invalid use of escape character*.

   d)   Otherwise, let *R1*, *R2*, and *R3* be the substrings of *R*, such that

   ```
   'R' = 'R1' || 'E' || '"' || 'R2' || 'E' || '"' || 'R3'
   ```

   is _True_.

   Case:

   i)   If any one of *R1*, *R2*, or *R3* is not the zero-length character string and does not have the format of a <regular expression>, then an exception condition is raised: *data exception — invalid regular expression*.

   ii)   If the predicate

```
'C' SIMILAR TO 'R1' || 'R2' || 'R3' ESCAPE 'E'
```

is not <u>*True*</u>, then the result of the <regular expression substring function> is the null value.

iii) Otherwise, the result *S* of the <regular expression substring function> is computed as follows:

1) Let *S1* be the shortest initial substring of *C* such that there is a substring *S23* of *C* such that the value of the following <search condition> is <u>*True*</u>:

```
'C' = 'S1' || 'S23' AND
'S1' SIMILAR TO 'R1' ESCAPE 'E' AND
'S23' SIMILAR TO '(R2R3)' ESCAPE 'E'
```

2) Let *S3* be the shortest final substring of *S23* such that there is a substring *S2* of *S23* such that the value of the following <search condition> is <u>*True*</u>:

```
'S23' = 'S2' || 'S3' AND
'S2' SIMILAR TO 'R2' ESCAPE 'E' AND
'S3' SIMILAR TO 'R3' ESCAPE 'E'
```

3) The result of the <regular expression substring function> is *S2*.

6.33   <JSON value constructor>

1. *Rationale: Resolve an ambiguity in <JSON value constructor>.*

Insert a new Syntax Rule:

6)   ...

c.1) If <JSON array constructor by enumeration> immediately contains exactly 1 (one) <JSON value expression>, then that <JSON value expression> shall not be a <scalar subquery>.

> NOTE 202.1 — This Syntax Rule resolves an ambiguity in which a <JSON array constructor> (e.g., JSON_ARRAY((SELECT a FROM t))) otherwise might be interpreted either as a <JSON array constructor by enumeration> or as a <JSON array constructor by query>. The ambiguity is resolved by adopting the interpretation that such a <JSON array constructor> is a <JSON array constructor by query>.

2. *Rationale: Correct symbol.*

Replace General Rule 2) d) ii) E) with:

2)   ...

d)   ...

ii)   ...

5) Let JVE_i be the <JSON value expression> immediately contained in JNV_i and let $VJVE_i$ be the value of JVE_i.

7 Query expressions

7.6 <table reference>

This Subclause is modified by Subclause 7.1, "<table reference>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 7.1, "<table reference>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 7.1, "<table reference>", in ISO/IEC 9075-14.

1. *Rationale: Correct the calculation of the maximum of the cardinalities of arrays.*

   Replace Syntax Rule 7) g) with:

   7) ...

      h) Let $MCARD_1$ be $CARD_1$. Let $MCARD_j$, $2 \le j \le NCV$, be

   ```
   CASE
     WHEN CARD_j > MCARD_{j-1}
       THEN CARD_j
     ELSE MCARD_{j-1}
   END
   ```

2. *Rationale: Clarify which variant of CHECK OPTION is intended.*

   Replace Syntax Rule 10) b) i) with:

   10) ...

      b) ...

         i) The view descriptor of *TT* shall indicate CASCADED CHECK OPTION.

3. *Rationale: Clarify delta tables.*

   Insert the following General Rule:

   5) ...

      d) All old delta tables and all new delta tables in the most recent statement execution context are destroyed.

7.11 <JSON table>

1. *Rationale: Use the correct symbol JTQCD.*

   Replace Syntax Rule 1) f) ix) with:

1) ...

    f) ...

        ix) If *JTQCD* does not contain <JSON table formatted column error behavior>, then

        Case:

           1) If *JTEB* is ERROR ON ERROR, then the implicit <JSON table formatted column error behavior> of *JTQCD* is ERROR ON ERROR.

           2) Otherwise, the implicit <JSON table formatted column error behavior> of *JTQCD* is NULL ON ERROR.

7.17   <query expression>

This Subclause is modified by Subclause 7.2, "<query expression>", in ISO/IEC 9075-14.

1. *Rationale: Avoid relying on a missing definition of possibly non-deterministic for <query primary>.*

Replace Syntax Rule 24) c) with:

24) ...

    c) The <query expression> contains a <query specification>, <table value constructor> or <explicit table> that is possibly non-deterministic.

8   Predicates

8.1   <predicate>

This Subclause is modified by Subclause 8.1, "<predicate>", in ISO/IEC 9075-14.

1. *Rationale: Provide the missing definition of possibly non-deterministic for <predicate>.*

Insert the following Syntax Rule:

1) A <predicate> *P* is *possibly non-deterministic* if any of the following are true:

    a) *P* contains a <value expression>, <nonparenthesized value expression primary> or <query expression> that is possibly non-deterministic.

    b) *P* contains a <comparison predicate> , <overlaps predicate>, or <distinct predicate> simply containing <row value predicand>s *RVP1* and *RVP2* such that the declared types of *RVP1* and *RVP2* have corresponding constituents such that one constituent is datetime with time zone and the other is datetime without time zone.

c) *P* contains a <quantified comparison predicate> or a <match predicate> simply containing a <row value predicand> *RVP* and a <table subquery> *TS* such that the declared types of *RVP* and *TS* have corresponding constituents such that one constituent is datetime with time zone and the other is datetime without time zone.

d) *P* contains a <member predicate> simply containing a <row value predicand> *RVP* and a <multiset value expression> *MVP* such that the declared type of the only field *F* of *RVP* and the element type of *MVP* have corresponding constituents such that one constituent is datetime with time zone and the other is datetime without time zone.

e) *P* contains a <submultiset predicate> simply containing a <row value predicand> *RVP* and a <multiset value expression> *MVP* such that the declared type of the only field *F* of *RVP* and the declared type of *MVP* have corresponding constituents such that one constituent is datetime with time zone and the other is datetime without time zone.

f) *P* contains a <multiset value expression> that specifies or implies MULTISET UNION, MULTISET EXCEPT, or MULTISET INTERSECT such that the element types of the operands have corresponding constituents such that one constituent is datetime with time zone and the other is datetime without time zone.

8.11   <unique predicate>

1. *Rationale: Clarify uniqueness in the presence of nulls.*

   Replace General Rule 2) with:

   2)   Case:

      a)   If there are zero or one rows in *T*, the result of the <unique predicate> is *True*.

      b)   If for every two rows *R1* and *R2* in *T*, the implementation uniqueness rule holds, the result of the <unique predicate> is *True*.

      c)   Otherwise, the result of the <unique predicate> is *False*.

9   Additional common rules

9.18   Compilation of an invocation of a polymorphic table function

1. *Rationale: Use correct symbol.*

   Replace General Rules 2) b) iii) 2) B) and 2) b) iii) 2) C) with:

   2)   ...

      b)   ...

iii)   ...

2)   ...

B)   The COUNT and TOP_LEVEL_COUNT components of the header of *ODA* are set to *NOC*.

C)   The number and maximum number of SQL item descriptors of *ODA* are set to *NOC*.

9.38   SQL/JSON path language: lexical elements

1.   *Rationale: Use the correct tokens for the "not equals" comparison operator in the SQL/JSON path language.*

Replace the BNF for <SQL/JSON special symbol> with:

```
<SQL/JSON special symbol> ::=
    <alternative not equals operator>
  | <asterisk>
  | <at sign>
  | <comma>
  | <dollar sign>
  | <double ampersand>
  | <double equals>
  | <double vertical bar>
  | <exclamation mark>
  | <greater than operator>
  | <greater than or equals operator>
  | <left bracket>
  | <left paren>
  | <less than operator>
  | <less than or equals operator>
  | <minus sign>
  | <not equals operator>
  | <percent>
  | <period>
  | <plus sign>
  | <question mark>
  | <right bracket>
  | <right paren>
  | <solidus>

<alternative not equals operator> ::=
  !=
```

9.39   SQL/JSON path language: syntax and semantics

1.   *Rationale: Use the correct tokens for the "not equals" comparison operator in the SQL/JSON path language.*

Replace the BNF for <JSON comp op> with:

```
<JSON comp op> ::=
    <double equals>
  | <JSON path not equals operator>
  | <less than operator>
  | <greater than operator>
  | <less than or equals operator>
  | <greater than or equals operator>
```

NOTE 451 — Equality operators have the same precedence as inequality comparison operators, unlike [ECMAScript].

```
<JSON path not equals operator> ::=
    <not equals operator>
  | <alternative not equals operator>
```

2. *Rationale: Use the correct symbols.*

    Replace General Rule 11) g) ii) 3) F) II) i) with:

    11) ...

       g) ...

          ii)   ...

             3) ...

                F) ...

                   II) ...

                      1)   Let $TJSFROM_i$ be the result of implementation-defined truncation or rounding of $RJSFROM_i$ and let $TJSTO_i$ be the result of implementation-defined truncation or rounding of $RJSTO_i$.

3. *Rationale: Use the correct symbols.*

    Replace General Rule 11) g) ii) 3) F) III) i) with:

    11) ...

       g) ...

          ii)   ...

             3) ...

                F) ...

                   III) ...

                      1)   Let $RJSU$ be the list of integers formed by concatenating RJS_i, 1 (one) $\le i \le s$, in that order.

4. *Rationale: Use the correct tokens for the "not equals" comparison operator in the SQL/JSON path language.*

Replace General Rule 12) c) iii) 2) D) I) with:

12) ...

    c) ...

        iii)   ...

            2) ...

                D) ...

                    I)   If *JCO* is <double equals> or <JSON path not equals operator>, and AI_i and BI_j are not equality-comparable, then let *ERR* be <u>*True*</u>.

5. *Rationale: Use the correct tokens for the "not equals" comparison operator in the SQL/JSON path language.*

Replace the lead text of General Rule 12) c) iii) 2) D) IV) with:

12) ...

    c) ...

        iii)   ...

            2) ...

                D) ...

                    IV)   If *JCO* is <JSON path not equals operator> then

                              Case:

10 Additional common elements

10.4   <routine invocation>

This Subclause is modified by Subclause 9.1, "<routine invocation>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 7.1, "<routine invocation>", in ISO/IEC 9075-10.
This Subclause is modified by Subclause 9.4, "<routine invocation>", in ISO/IEC 9075-13.
This Subclause is modified by Subclause 11.1, "<routine invocation>", in ISO/IEC 9075-14.

1. *Rationale: Supply the correct parameters.*

Replace Syntax Rule 9) h) iii) 5) with:

9) ...

h) ...

    iii) ...

        5) ☐04 If XA_i is an <SQL parameter reference>, a <column reference>, or a <target array element specification>, then the Syntax Rules of Subclause 9.2, "Store assignment", are applied with XA_i as *TARGET* and P_i as *VALUE*.

        NOTE 480 — The <column reference> can only be a new transition variable column reference.

2. *Rationale: Supply the correct arguments.*

Replace Syntax Rule 9) h) v) with:

9) ...

    h) ...

        v) For each SQL parameter P_i, 1 (one) ≤ i ≤ *SRNP*, that is an input SQL parameter or both an input SQL parameter and an output SQL parameter, and each XA_i that is not a <contextually typed value specification>, the Syntax Rules of Subclause 9.2, "Store assignment", are applied with P_i as *TARGET* and XA_i as *VALUE*.

11 Schema definition and manipulation

11.4 <column definition>

This Subclause is modified by Subclause 10.4, "<column definition>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 12.1, "<column definition>", in ISO/IEC 9075-14.

1. *Rationale: Avoid relying on a missing definition of possibly non-deterministic for <generation expression>.*

Replace Syntax Rule 10) d) with:

10) ...

    d) *GE* shall not contain a possibly non-deterministic <value expression>.

11.28 <drop table period definition>

1. *Rationale: Use correct terminology.*

Replace General Rule 1) b) with:

1) ...

    13

b)   Let *ECN* be the <column name> of the system-time period end column of *T*. The following <alter table statement> is executed without further Access Rule checking:

```
ALTER TABLE TN DROP COLUMN ECN CASCADE
```

11.32 <view definition>

This Subclause is modified by Subclause 10.11, "<view definition>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 10.3, "<view definition>", in ISO/IEC 9075-13.
This Subclause is modified by Subclause 12.4, "<view definition>", in ISO/IEC 9075-14.

1.   *Rationale: Clarify which variant of CHECK OPTION is intended.*

Replace Syntax Rule 16) with:

16) If CASCADED CHECK OPTION is specified or implied, then the viewed table shall be effectively updatable.

2.   *Rationale: LOCAL CHECK OPTION cannot be enforced except on simply updatable views.*

Insert a new Syntax Rule.

16₁) If LOCAL CHECK OPTION is specified, then the viewed table shall be simply updatable.

3.   *Rationale: A generally underlying table of a view that specifies CHECK OPTION may not have an INSTEAD OF trigger.*

Insert a new Syntax Rule.

16₂) If CHECK OPTION is specified, then no generally underlying table of *QE* shall be a view whose view descriptor includes an indication that the view is trigger updatable, trigger insertable-into or trigger deletable.

4.   *Rationale: Clarify a rule by removing redundant specification.*

Replace Syntax Rule 18) with:

18) If WITH LOCAL CHECK OPTION is specified, then *QE* shall not generally contain a <query expression> *QE2* that is possibly non-deterministic unless *QE2* is generally contained in the hierarchical <query expression> of a viewed table that is a leaf underlying table of *QE*.

If WITH CASCADED CHECK OPTION is specified, then *QE* shall not generally contain a <query expression> or <query specification> that is possibly non-deterministic.

5.   *Rationale: Differentiate the restrictions for CASCADED CHECK OPTION and LOCAL CHECK OPTION in view hierarchies.*

Replace Syntax Rule 21) t) iv) with:

21) ...

   t)   ...

      iv)   If the view descriptor of *SUPERT* or any supertable of *SUPERT* includes an indication that CASCADED CHECK OPTION was specified, then *QS* shall not be possibly non-deterministic.

      iv.1)   If the view descriptor of *SUPERT* or any supertable of *SUPERT* includes an indication that LOCAL CHECK OPTION was specified, then *QS* shall not generally contain a <query expression> *QE2* that is possibly non-deterministic unless *QE2* is generally contained in the hierarchical <query expression> of a viewed table that is a leaf underlying table specification of *QS*.

6. *Rationale: Differentiate the restrictions for CASCADED CHECK OPTION and LOCAL CHECK OPTION; both rules must include all supertables.*

Replace Syntax Rule 21) t) v) with:

21) ...

   t)   ...

      v)   If *SUPERT* or any supertable of *SUPERT* is referenced in any check constraint descriptor, assertion descriptor, or the original <query expression> of any view having CASCADED CHECK OPTION, then *QS* shall not be possibly non-deterministic.

      v.1)   If *SUPERT* or any supertable of *SUPERT* is referenced, except as a leaf underlying table, in the original <query expression> of any view having LOCAL CHECK OPTION, then *QS* shall not be possibly non-deterministic.

7. *Rationale: LOCAL CHECK OPTION cannot be enforced except on simply updatable views.*

Insert the following Syntax Rule.

21) ...

   t)   ...

      v.2)   If the view descriptor of *SUPERT* or any supertable of *SUPERT* includes an indication that LOCAL CHECK OPTION was specified, then *QS* shall be simply updatable.

11.60 <SQL-invoked routine>

This Subclause is modified by Subclause 10.24, "<SQL-invoked routine>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 11.11, "<SQL-invoked routine>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 10.8, "<SQL-invoked routine>", in ISO/IEC 9075-13.
This Subclause is modified by Subclause 12.8, "<SQL-invoked routine>", in ISO/IEC 9075-14.

1. *Rationale: <locator indication> is specified in the parameter lists of PTF component procedures but not in the parameter list of the polymorphic table function itself. The default value of a private parameter must be assignable to the declared type of the default parameter.*

    Replace Syntax Rule 10) e) with:

    10) ...

    e) If <PTF private parameters> *PTFPRIV* is specified, then for each <SQL parameter declaration> *PTFPRIVPD* contained in *PTFPRIV*:

        i)   *PTFPRIVPD* shall not specify <parameter mode>, <locator indication>, RESULT, or a <parameter type> that is ROW, <generic table parameter type> or <descriptor parameter type>.

        ii)   If *PTFPRIVPD* contains a <parameter default> *PDEF*, then let *PV* be a variable whose declared type is specified by the <data type> contained in *PTFPRIVPD*. Syntax Rules of Subclause 9.2, "Store assignment", are applied with *PV* as *TARGET* and *PDEF* as *VALUE*.

        iii) The SQL parameter declared by *PTFPRIVPD* is a *private parameter* of *R*.

2. *Rationale: <locator indication> is specified in the parameter lists of PTF component procedures but not in the parameter list of the polymorphic table function itself.*

    Insert a new Syntax Rule:

    10) ...

    g) ...

        i.0) <locator indication> shall not be specified.

3. *Rationale: <locator indication> is specified in the parameter lists of PTF component procedures but not in the parameter list of the polymorphic table function itself.*

    Replace the lead text of Syntax Rule 10) h) with:

    10) ...

    h) In the following rules, a parameter list *ACTUALPL* is said to *conform* to a parameter list *TEMPLATEPL* if they have the same number of parameters, and, for each parameter APL_i of *ACTUALPL*, the <parameter type> of APL_i is the same as the <parameter type>, disregarding the presence or absence of <locator indication>, of the parameter TPL_i at the same ordinal position in *TEMPLATEPL*.

4. *Rationale: NPL must be defined for polymorphic table functions.*

    Insert a new Syntax Rule:

    10) ...

m1) Let *NPL* be the <SQL parameter declaration list> contained in the <function specification>.

12 Access Control

12.2   <grant privilege statement>

This Subclause is modified by Subclause 11.1, "<grant privilege statement>", in ISO/IEC 9075-13.

1. *Rationale: Grantor determination must be done in the Syntax Rules.*

   Insert the following Syntax Rules:

   3.1) Case:

       a)   If GRANTED BY is omitted, then let *G* be OMITTED.

       b)   Otherwise, let *G* be <grantor>.

   3.2) Let *A* be the result of applying the Syntax Rules of Subclause 12.8, "Grantor determination", with *G* as *GRANTOR*. *A* is the *grantor* of the <grant privilege statement>.

2. *Rationale: Supply missing Access Rules.*

   Insert the following Access Rules:

   1)   The applicable privileges for *A* shall include a privilege identifying *O*, or, if *O* is a table, a column of *O* or a table/method pair whose table is *O*.

   2)   If <privileges> contains a <privilege column list> *PCL*, then for every <column name> *CN* contained in *PCL*, the applicable privileges shall include a column privilege whose object is the column identified by *CN*.

   3)   If <privileges> contains a <privilege method list> *PML*, then for every <specific routine designator> *SRD* contained in *PML*, the applicable privileges shall include a privilege whose object is the method identified by *SRD*.

3. *Rationale: Grantor determination is now done in the Syntax Rules; privilege checking is now done in the Access Rules.*

   Delete General Rules 1), 2) and 3).

12.3   <privileges>

This Subclause is modified by Subclause 11.2, "<privileges>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 13.1, "<privileges>", in ISO/IEC 9075-9.

This Subclause is modified by Subclause 11.2, "<privileges>", in ISO/IEC 9075-13.

1. *Rationale:Clarify the grantor in the case of ALL PRIVILEGES.*

   Replace Syntax Rule 10) a) with the following:

   10) ...

   a) If ALL PRIVILEGES is specified, then let A be the grantor of the <grant privilege statement> or the <revoke statement> that simply contains <privileges>. ALL PRIVILEGES specifies the set union of the following sets of privilege descriptor kernels:

   i) The set of all privilege descriptor kernels whose object is O and whose action is one of the actions on O for which A has grantable privilege descriptors.

   ii) If O is a table, then all privilege descriptor kernels whose object $O2$ is a column of O and whose action is one of the actions on $O2$ for which A has grantable column privilege descriptors.

   iii) If O is a table, then all privilege descriptor kernels whose object $O3$ is a table/method pair in which the table is O and whose action is one of the actions on $O3$ for which A has grantable column privilege descriptors.

12.4   <role definition>

1. *Rationale:Grantor determination must be done in the Syntax Rules.*

   Add the following Syntax Rules:

   2) Case:

   a) If WITH ADMIN is omitted, then let G be OMITTED.

   b) Otherwise, let G be <grantor>.

   3) Let A be the result of applying the Syntax Rules of Subclause 12.8, "Grantor determination", with G as GRANTOR.

2. *Rationale:Grantor determination must be done in the Syntax Rules.*

   Delete General Rules 2) and 3).

12.5   <grant role statement>

1. *Rationale:Grantor determination must be done in the Syntax Rules.*

   Add the following Syntax Rules:

   2) Case:

a) If <grantor> is omitted, then let *G* be OMITTED.

b) Otherwise, let *G* be <grantor>.

3) Let *A* be the result of applying the Syntax Rules of Subclause 12.8, "Grantor determination", with *G* as GRANTOR.

2. *Rationale: Grantor determination must be done in the Syntax Rules.*

Delete General Rules 1) and 2).

12.6 <drop role statement>

1. *Rationale: <drop role statement> needs <drop behavior>.*

Replace the Format with:

```
<drop role statement> ::=
  DROP ROLE <role name> <drop behavior>
```

2. *Rationale: <drop role statement> needs <drop behavior>.*

Add the following Syntax Rule:

2) Let *DB* be the <drop behavior>.

3. *Rationale: <drop role statement> needs <drop behavior>.*

Replace General Rules 1) and 2) with:

1) For every <authorization identifier> *A* identified by a role authorization descriptor as having been granted to *R*, the following <revoke role statement> is effectively executed without further Access Rule checking:

```
REVOKE R FROM A DB
```

12.7 <revoke statement>

This Subclause is modified by Subclause 11.3, "<revoke statement>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 13.2, "<revoke statement>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 11.3, "<revoke statement>", in ISO/IEC 9075-13.

1. *Rationale: Grantor determination must be done in the Syntax Rules.*

Insert the following Syntax Rules:

2) Case:

    a)   If GRANTED BY is omitted, then let G be OMITTED.

    b)   Otherwise, let G be <grantor>.

3) Let A be the result of applying the Syntax Rules of Subclause 12.8, "Grantor determination", with G as GRANTOR. A is the *grantor* of the <revoke statement>.

2. *Rationale: Supply missing Access Rules.*

Insert the following Access Rules:

1) Case:

    a)   If the <revoke statement> is a <revoke privilege statement>, then:

        i)   The applicable privileges for A shall include a privilege identifying O, or if O is a table, a column of O or a table/method pair whose table is O.

        ii)   If <privileges> contains a <privilege column list> PCL, then for every <column name> CN contained in *PCL*, the applicable privileges shall include a column privilege whose object is the column identified by CN.

        iii)   If <privileges> contains a <privilege method list> PML, then for every <specific routine designator> SRD contained in *PML*, the applicable privileges shall include a privilege whose object is the method identified by SRD.

    b)   If the <revoke statement> is a <revoke role statement>, then, for every role R identified by a <role revoked>, there shall exist a grantable role authorization descriptor whose role name is R, and whose grantee is A or an applicable role of A.

3. *Rationale: Grantor determination must be done in the Syntax Rules.*

Delete General Rules 1), 2) and 3).

12.8   Grantor determination

1. *Rationale: Grantor determination must be done in the Syntax Rules.*

Replace the Subclause Signature and add a new Syntax Rule:

Subclause Signature

```
"Grantor determination" [Syntax Rules] (
  Parameter: "GRANTOR"
)
```

1) Let G be the *GRANTOR* in an application of the Rules of this Subclause.

2. *Rationale: Grantor determination must be done in the Syntax Rules.*

   Insert the following Syntax Rule:

   2) The grantor A is derived from G as follows.

   Case:

   a) If G is OMITTED, then

      Case:

      i) If there is a current user identifier, then A is the current user identifier.

      ii) Otherwise, A is the current role name.

   b) If G is CURRENT_USER, then

      Case:

      i) There shall be a current user identifier.

      ii) Otherwise, A is the current user identifier.

   c) If G is CURRENT_ROLE, then

      Case:

      i) There shall be a current role name.

      ii) Otherwise, A is the current role name.

3. *Rationale: Grantor determination must be done in the Syntax Rules.*

   Delete General Rules 1) and 2).

13 SQL-client modules

13.4   <SQL procedure statement>

This Subclause is modified by Subclause 12.2, "<SQL procedure statement>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 14.3, "<SQL procedure statement>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 13.2, "<SQL procedure statement>", in ISO/IEC 9075-14.

1. *Rationale: Clarify delta tables.*

   Insert the following General Rule:

   3.1) In *NEWSEC*, the set of old delta tables and new delta tables is initially empty.

2. *Rationale: Initialize the set of views to be tested for CHECK OPTION.*

   Insert the following General Rule:

   3.2) In *NEWSEC*, the set of views to be checked is initially empty.

14 Data manipulation

14.11 <insert statement>

This Subclause is modified by Subclause 14.4, "<insert statement>", in ISO/IEC 9075-14.

1. *Rationale: Clarify <override clause> in the presence of identity columns and system-time period columns.*

   Insert the following Syntax Rule.

   10.1) Let *OC* be any object column. Let *n* be the sequential number of the <column name> in the <insert column list> that identifies *OC*. Then *OC* is *defaulted* if either of the following is true:

   a) <from default> is specified.

   b) <from constructor> *FC* is specified, and, for every <contextually typed row value constructor> *CTRVC* simply contained in *FC*, *CTRVC* is a <contextually typed row value constructor> whose *n*-th <contextually typed row value constructor element> is <default specification>.

2. *Rationale: Clarify <override clause> in the presence of identity columns and system-time period columns.*

   Replace the lead text of Syntax Rule 11) with:

   11) In any of the following circumstances an <override clause> is permitted or required; if none of the following circumstances pertain then <override clause> shall not be specified.

3. *Rationale: Clarify <override clause> in the presence of identity columns and system-time period columns.*

   Replace Syntax Rules 11) b), c), d) and e) with:

   11) ...

   b) If some underlying column of an object column *OC* is an identity column whose values are always generated, and *OC* is not defaulted, then <override clause> shall be specified.

   c) If some underlying column of an object column is an identity column whose values are generated by default, and <override clause> is specified, then <override clause> shall specify OVERRIDING USER VALUE.

   d) If some underlying column of an object column *OC* is a system-time period start column or a system-time period end column whose values are always generated, then:

    i)   If *OC* is not defaulted, then <override clause> shall be specified.

    ii)   If <override clause> is specified, then <override clause> shall specify OVERRIDING USER VALUE.

4.   *Rationale: Invoke the new Subclause 15.20 "Checking of views that specify CHECK OPTION" for correct testing of CHECK OPTION.*

Replace General Rule 8) b) with:

8)   ...

    b)   If *T* is a viewed table, then:

        i)   The General Rules of Subclause 15.12, "Effect of inserting a table into a viewed table", are applied with *S* as *SOURCE* and *T* as *TARGET*.

        ii)   The General Rules of Subclause 15.20, "Checking of views that specify CHECK OPTION", are applied with *INSERT* as *OPERATION*.

14.12 <merge statement>

This Subclause is modified by Subclause 14.5, "<merge statement>", in ISO/IEC 9075-14.

1.   *Rationale: Provide syntax checks for each <merge when not matched clause> individually; clarify <override clause> in the presence of identity columns and system-time period columns.*

Delete Syntax Rule 13) and 14).

2.   *Rationale: Provide syntax checks for each <merge when not matched clause> individually; clarify <override clause> in the presence of identity columns and system-time period columns.*

Replace Syntax Rule 20) with:

20) For each <merge when not matched clause> *MWNMC*:

    a)   If an <insert column list> is omitted, then an <insert column list> that identifies all columns of *T* in the ascending sequence of their ordinal position within *T* is implicit.

    b)   Let *IOC* be any insert object column of *MWNMC*. Let *n* be the sequential number of the <column name> in the <insert column list> of *MWNMC* that identifies *IOC*. Then *IOC* is *defaulted* if the *n*-th <merge insert value element> of *MWNMC* is <default specification>.

    c)   Let *NI* be the number of <merge insert value element>s contained in <merge insert value list> of *MWNMC*. Let EXP_1, EXP_2, ... , EXP_{NI} be those <merge insert value element>s.

    d)   The number of <column name>s in the <insert column list> of *MWNMC* shall be equal to *NI*.

    e)   The declared type of every <contextually typed value specification> *CVS* in a <merge insert value list> is the data type *DT* indicated in the column descriptor for the positionally corresponding insert object column of *MWNMC* in the explicit or implicit <insert column list>. If

CVS is an <empty specification> that specifies ARRAY, then *DT* shall be an array type. If *CVS* is an <empty specification> that specifies MULTISET, then *DT* shall be a multiset type.

f) Every insert object column of *MWNMC* of which some underlying column is a generated column shall be defaulted.

g) For 1 (one) ≤ *i* ≤ *NI*, the Syntax Rules of Subclause 9.2, "Store assignment", are applied with *EXP$_i$* as *VALUE* and the column of table *T* identified by the *i*-th <column name> in the <insert column list> of *MWNMC* as *TARGET*.

h) In any of the following circumstances an <override clause> is permitted or required; if none of the following circumstances pertain then <override clause> shall not be specified.

　　i) If some underlying column of an insert object column of *MWNMC* is a system-generated self-referencing column or a derived self-referencing column, then *MWNMC* shall specify <override clause>.

　　ii) If some underlying column of an insert object column *IOC* of *MWNMC* is an identity column whose values are always generated, and *IOC* is not defaulted, then *MWNMC* shall specify <override clause>.

　　iii) If some underlying column of an insert object column of *MWNMC* is an identity column whose values are generated by default and *MWNMC* specifies <override clause>, then that <override clause> shall specify OVERRIDING USER VALUE.

　　iv) If some underlying column of an insert object column *IOC* of *MWNMC* is a system-time period start column or a system-time period end column whose values are always generated, then:

　　　　1) If *IOC* is not defaulted, then *MWNMC* shall specify <override clause>.

　　　　2) If *MWNMC* specifies <override clause> OC, then OC shall specify OVERRIDING USER VALUE.

3. *Rationale: Clarify delta tables.*

Replace General Rule 6) a) i) 6) with:

6) ...

　　a) ...

　　　　i) ...

　　　　　　6) Let *ND$_j$* be a copy of the new delta table of update operation on *T*, if any. The new delta table of update operation on *T* is destroyed.

4. *Rationale: Clarify delta tables.*

Replace General Rule 6) a) ii) 9) with:

6) ...

　　a) ...

ii) ...

9) Let ND_j be a copy of the new delta table of insert operation on T, if any. The new delta table of insert operation on T is destroyed.

5. *Rationale: Invoke the new Subclause 15.20 "Checking of views that specify CHECK OPTION" for correct testing of CHECK OPTION.*

Insert a new General Rule.

6.1) The General Rules of Subclause 15.20, "Checking of views that specify CHECK OPTION", are applied with *MERGE* as *OPERATION*.

14.14 <update statement: searched>

This Subclause is modified by Subclause 14.7, "<update statement: searched>", in ISO/IEC 9075-14.

1. *Rationale: Invoke the new Subclause 15.20 "Checking of views that specify CHECK OPTION" for correct testing of CHECK OPTION.*

Replace General Rule 14) b) with:

14) ...

b) If T is a viewed table, then:

i) The General Rules of Subclause 15.15, "Effect of replacing some rows in a viewed table", are applied with *TT* as *VIEW NAME* and the replacement set for T as *REPLACEMENT SET FOR VIEW NAME*.

ii) The General Rules of Subclause 15.20, "Checking of views that specify CHECK OPTION", are applied with *UPDATE* as *OPERATION*.

14.15 <set clause list>

1. *Rationale: The self-referencing column cannot be updated.*

Insert the following Syntax Rule:

3.1) No <object column> shall reference a column of which some underlying column is a self-referencing column.

15 Additional data manipulation rules

15.6   Effect of a positioned update

1. *Rationale: Invoke the new Subclause 15.20 "Checking of views that specify CHECK OPTION" for correct testing of CHECK OPTION.*

    Replace General Rule 17) b) with:

    17) ...

    b)   If *LUT* is a viewed table, then:

      i)   The General Rules of Subclause 15.15, "Effect of replacing some rows in a viewed table", are applied with *TT* as *VIEW NAME* and the replacement set for *T* as *REPLACEMENT SET FOR VIEW NAME*.

      ii)   The General Rules of Subclause 15.20, "Checking of views that specify CHECK OPTION", are applied with *UPDATE* as *OPERATION*.

15.10 Effect of inserting tables into base tables

This Subclause is modified by Subclause 15.2, "Effect of inserting tables into base tables", in ISO/IEC 9075-9.

1. *Rationale: Correct a typo.*

    Replace General Rule 3) a) with:

    3)   ...

    a)   If a state change *SC* exists in *SSC* with subject table *BT* and trigger event INSERT, then the rows in the source table for *BT* are added to the set of transitions of *SC*.

2. *Rationale: Correct definition of the new delta table of insert operation during a <merge statement>.*

    Replace General Rule 5) b) with:

    5)   ...

    b)   Case:

      i)   If *BT* has no new delta table of insert operation on *BT*, then *SOT* is the *new delta table of insert operation* on *BT*.

      ii)   Otherwise, the rows of *SOT* are added to the new delta table of insert operation on *BT*.

15.12 Effect of inserting a table into a viewed table

1. *Rationale: Ensure that the new Subclause 15.20, "Checking of views that specify CHECK OPTION", operates on the correct set of views.*

    Replace General Rule 3) b) with:

    3)   ...

    b)   Otherwise,

    i)   If *TD* indicates CHECK OPTION, then *TD* is added to the set of views to be checked in the current statement execution context.

    ii)   The General Rules of Subclause 15.11, "Effect of inserting a table into a derived table", are applied with *S* as *SOURCE* and *T* as *TARGET*.

15.13 Effect of replacing rows in base tables

This Subclause is modified by Subclause 15.3, "Effect of replacing rows in base tables", in ISO/IEC 9075-9.

1. *Rationale: Clarify delta tables.*

    Replace General Rule 9) b) with:

    9)   ...

    b)   Otherwise, for every row *R* that is identified for replacement in *T*, *R* is replaced by its corresponding replacement row. *R* is no longer identified for replacement. *T* is no longer identified for replacement processing, with or without subtables. Let *SUP* be the set consisting of every replacement row of every *R*.

    Case:

    i)   If *T* has no new delta table of update operation, then *SUP* is the *new delta table of update operation* on *T*.

    ii)   Otherwise, the rows of *SUP* are added to the new delta table of update operation on *T*.

    NOTE 694.1 — The latter scenario can arise when updating a view *V* with proper subtables, since this subclause will be invoked for each operand of a UNION ALL CORRESPONDING in the hierarchical <query expression> of *V*.

2. *Rationale: Define the subtable relationships between new delta tables of update operation.*

    Insert a new General Rule.

    9.1) For every pair of tables *T1* and *T2* in *TT2*, if *T1* is a direct subtable of *T2*, then the new delta table of update operation on *T1* is a direct subtable of the new delta table of update operation on *T2*.

15.15 Effect of replacing some rows in a viewed table

1. *Rationale: Ensure that the new Subclause 15.20, "Checking of views that specify CHECK OPTION", operates on the correct set of views.*

    Replace General Rule 3) b) with:

    3) ...

    b) Otherwise,

        i) If *TD* indicates CHECK OPTION, then *TD* is added to the set of views to be checked in the current statement execution context.

        ii) The General Rules of Subclause 15.14, "Effect of replacing some rows in a derived table", are applied with *QE* as *TABLE* and *RS* as *REPLACEMENT SET FOR TABLE*.

15.20 Checking of views that specify CHECK OPTION

1. *Rationale: Specify the new Subclause 15.20 "Checking of views that specify CHECK OPTION" for correct testing of CHECK OPTION.*

    Insert the following new Subclause:

15.20 Checking of views that specify CHECK OPTION

Subclause Signature

```
"Checking of views that specify CHECK OPTION" [General Rules] (
  Parameter: "OPERATION"
)
```

Function

Check views that specify CHECK OPTION after an insert or update operation.

Syntax Rules

None.

Access Rules

None.

General Rules

1) Let *OP* be the *OPERATION* in an application of the General Rules of this Subclause.

2) Let *LOV* be the set of views to be checked in the current execution context.

3) If *OP* is MERGE, then for every target leaf underlying table *TLUT* of every view in *LOV*, the new delta table of merge operation on *TLUT* is the multiset union of the new delta table of insert operation on *TLUT*, if any, with the new delta table of update operation on *TLUT*, if any.

> NOTE 709.1 — If the <target table> of a <merge statement> identifies a view *V*, then the new delta table of merge operation of *V* has been defined in the General Rules of 14.12 <merge statement>, but not for any of the base tables that the <merge statement> targets. If the operation is INSERT or UPDATE, then the relevant new delta table of the target table(s) has been defined in either 15.10 "Effect of inserting tables into base tables" or 15.13 "Effect of replacing rows in base tables", respectively.

4) For every view *V* in *LOV*:

   a) Let *GGUTS* be the graph of generally underlying table specifications of the original <query expression> of *V*. Let *N* be one plus the number of generally underlying table specifications in *GGUTS* that are views. Let V_1 be *V*, and let $V_2, ... , V_N$ be an enumeration of the views that are generally underlying table specifications of *V*.

   b) For all *i*, 1 (one) $\leq i \leq N$,

      i) Let *s(i)* be the number of subviews of V_i.

         > NOTE 709.2 — Every view is a subview of itself, therefore *s(i)* is at least 1.

      ii) For all *j*, 1 (one) $\leq j \leq s(i)$, let $SV_{i,j}$ be an enumeration of the subviews of V_i, chosen so that SV_{i1} is V_i.

      iii) For all *j*, 1 (one) $\leq j \leq s(i)$,

         1) Let $SVX_{i,j}$ be a view descriptor that is a copy of the view desciptor of $SV_{i,j}$.

         2) The name of the view described by $SVX_{i,j}$ is set to a distinct effective view name $EVN_{i,j}$.

         3) Case:

            A) If *j* = 1 (one), then $SVX_{i,j}$ is set to indicate that it has no direct superview.

               > NOTE 709.3 — *V* may have a proper superview, but only subviews matter for checking of CHECK OPTION. Hence for each *i*, these rules construct a family of subviews whose maximal superview is SVX_{i1}.

            B) Otherwise, if the direct superview of $SV_{i,j}$ is $SV_{i,k}$, then the direct superview of $SVX_{i,j}$ is set to $SVX_{i,k}$.

         4) For every <table name> *TN* contained in the original <query expression> of $SVX_{i,j}$:

            Case:

            A) If *TN* references some view $SV_{f,g}$, then *TN* is replaced by $EVN_{f,g}$.

               > NOTE 709.4 — That is, all view names in the copied view descriptors are renamed so that they reference views in the set of copied view descriptors.

      B)   If *TN* references a base table *BT* and *BT* is a target leaf generally underlying table of the hierarchical <query expression> of *V*, then

         Case:

          I)   If *OP* is INSERT and *BT* has a new delta table of insert operation *NDTI*, then *TN* is replaced by a distinct effective name for *NDTI*.

          II)   If *OP* is UPDATE and *BT* has a new delta table of update operation *NDTU*, then *TN* is replaced by a distinct effective name for *NDTU*.

          III)   If *OP* is MERGE and *BT* has a new delta table of merge operation *NDTU*, then *TN* is replaced by a distinct effective name for *NDTU*.

      C)   Otherwise, *TN* is not replaced.

   5)   If *V* specifies LOCAL CHECK OPTION and i > 1 (one), then every <where clause> is removed from the original <query expression> of $SVX_{i,j}$.

> NOTE 709.5 — If a target leaf underlying table *TLUT* of *V* is a view that also specifies CHECK OPTION, then *TLUT* is in *LOV* and will be checked in its own right; there is no need to check it when checking *V*. If *V* specifies CASCADED CHECK OPTION, then no <where clause>s are removed, effectively checking all views that are target leaf underlying tables of *V*. Some views may be redundantly checked twice because of these considerations.

iv)   General Rules of Subclause 9.32, "Generation of the hierarchical <query expression> of a view", are applied with $SVX_{1,1}$ as *VIEW*.

> NOTE 709.6 — This is the only hierarchical <query expression> that is required to check *V*. Note that *V* = V_1 = $SV_{1,1}$, which was copied and edited to form $SVX_{1,1}$. For *i* > 1 (one), only the original <query expression>s are required, though there would be no harm in forming their hierarchical <query expression>s.

v)   Let *NVDX* be the number of rows in

```
SELECT *
FROM EVN1,1
```

vi)   Let *NDT* be the number of rows in

      Case:

   1)   If *OP* is INSERT, then the new delta table of insert operation of the target leaf underlying table of the original <query expression> of *V*.

   2)   If *OP* is UPDATE, then the new delta table of update operation of the target leaf underlying table of the original <query expression> of *V*.

   3)   If *OP* is MERGE, then the new delta table of merge operation of the target leaf underlying table of the original <query expression> of *V*.

vii)   If *NVDX* does not equal *NDT*, then an exception condition is raised: *with check option violation*

Conformance Rules

None.

20 Dynamic SQL

20.7 <prepare statement>

This Subclause is modified by Subclause 16.1, "<prepare statement>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 17.2, "<prepare statement>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 17.4, "<prepare statement>", in ISO/IEC 9075-14.

1. *Rationale: Use correct terminology.*

   Replace the lead text of General Rule 5) with:

   5) 04 Let *MP* be the implementation-defined maximum value of <precision> for the NUMERIC data type. Let *ML* be the implementation-defined maximum length of variable-length character strings. For each <value expression> *DP* in *P* or *PS* that meets the criteria for *DPV*, let *DT* denote its declared type. The syntactic substitutions specified in Subclause 14.15, "<set clause list>", shall not be applied until the data types of <dynamic parameter specification>s are determined by this General Rule.

21 Embedded SQL

21.2 <embedded exception declaration>

1. *Rationale: Correct an error in the rule structuring.*

   Replace General Rules 1) c) and 1) d) with:

   1) ...

      c) Case:

         i) If the execution of the <SQL procedure statement> caused the violation of one or more constraints or assertions, then:

            1) Let *ECN* be the set of <embedded exception declaration>s in *E* that specify CONSTRAINT and the <constraint name> of a constraint that was violated by execution of *STMT*.

            2) If *ECN* contains more than one <embedded exception declaration>, then an implementation-dependent <embedded exception declaration> is chosen from *ECN*; otherwise, the single <embedded exception declaration> in *ECN* is chosen.

            3) A GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> chosen from *ECN*.

      ii)   Otherwise, let *ECS* be the set of <embedded exception declaration>s in *E* that specify SQLSTATE, an <SQLSTATE class code>, and an <SQLSTATE subclass code>.

Case:

1) If *ECS* contains an <embedded exception declaration> *EY* that specifies an <SQL-STATE class code> identical to *CV* and an <SQLSTATE subclass code> identical to *SCV*, then a GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> *EY*.

2) Otherwise, let *EC* be the set of <embedded exception declaration>s in *E* that specify SQLSTATE and an <SQLSTATE class code> without an <SQLSTATE subclass code>.

Case:

A) If *EC* contains an <embedded exception declaration> *EY* that specifies an <SQLSTATE class code> identical to *CV*, then a GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> *EY*.

B) Otherwise, let *EX* be the set of <embedded exception declaration>s in *E* that specify SQLEXCEPTION.

Case:

I) If *EX* contains an <embedded exception declaration> *EY* and *CV* belongs to Category X in Table 38, "SQLSTATE class and subclass codes", then a GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> *EY*.

II) Otherwise, let *EW* be the set of <embedded exception declaration>s in *E* that specify SQLWARNING.

Case:

1) If *EW* contains an <embedded exception declaration> *EY* and *CV* belongs to Category W in Table 38, "SQLSTATE class and subclass codes", then a GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> *EY*.

2) Otherwise, let *ENF* be the set of <embedded exception declaration>s in *E* that specify NOT FOUND. If *ENF* contains an <embedded exception declaration> *EY* and *CV* belongs to Category N in Table 38, "SQLSTATE class and subclass codes", then a GO TO statement of the host language is performed, specifying the <host label identifier>, <host PL/I label variable>, or <unsigned integer> of the <go to> specified in the <embedded exception declaration> *EY*.

23 Diagnostics management

23.1 <get diagnostics statement>

This Subclause is modified by Subclause 18.1, "<get diagnostics statement>", in ISO/IEC 9075-4.
This Subclause is modified by Subclause 23.1, "<get diagnostics statement>", in ISO/IEC 9075-9.
This Subclause is modified by Subclause 20.1, "<get diagnostics statement>", in ISO/IEC 9075-14.

1. *Rationale: Clarify that both options of CHECK OPTION apply.*

   Replace General Rule 4) j) with:

4) ...

   j) If the value of RETURNED_SQLSTATE corresponds to *with check option violation*, then the values of CATALOG_NAME, SCHEMA_NAME, and TABLE_NAME are the <catalog name>, the <unqualified schema name> of the <schema name> of the schema that contains the view that caused the violation of the CHECK OPTION, and the <qualified identifier> of that view, respectively.

Annex B
(informative)

Implementation-defined elements

This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-3.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-4.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-9.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-10.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-11.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-13.
This Annex is modified by Annex B, "Implementation-defined elements", in ISO/IEC 9075-14.

1. *Rationale: Add implementation-defined elements introduced by the clarification of uniqueness in the presence of nulls.*

   Insert the following item:

131) Subclause 4.23.3.2, "Unique constraints"

   a) It is implementation-defined whether two rows whose column values are all the null value are considered unique with nulls distinct, or unique with nulls not distinct.

   b) It is implementation-defined whether the *implementation uniqueness rule* is unique with nulls distinct or unique with nulls not distinct.