INTERNATIONAL STANDARD ISO/IEC 9075:1992
TECHNICAL CORRIGENDUM 1

Published 1996-08-15

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION
INTERNATIONAL ELECTROTECHNICAL COMMISSION • МЕЖДУНАРОДНАЯ ЭЛЕКТРОТЕХНИЧЕСКАЯ КОМИССИЯ • COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

Information technology — Database languages — SQL

TECHNICAL CORRIGENDUM 1

Technologies de l'information — Langages de bases de données — SQL

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to International Standard ISO/IEC 9075:1992 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 21, *Open systems interconnection, data management and open distributed processing.*

Relation to previous technical corrigenda:

This Corrigendum contains the cumulative set of corrections to ISO/IEC 9075:1992.
It cancels and replaces Technical Corrigendum 1:1994.

Statement of purpose for rationale:

A statement indicating the rationale for each change to ISO/IEC 9075:1992 is included. This is to inform the users of that standard as to the reason why it was judged necessary to change the original wording. In many cases the reason is editorial or to clarify the wording; in some cases it is to correct an error or an omission in the original wording.

Notes on rule numbering:

This Corrigendum introduces some new Syntax, Access, General and Leveling Rules. The new Rules in this Corrigendum have been numbered as follows:
Rules inserted between, for example, Rules 7) and 8) (in ISO/IEC 9075:1992) are numbered 7.1), 7.2), etc. [or 7) a.1), 7) a.2), etc.]. Those inserted before Rule 1) are numbered 0.1), 0.2). etc.

ICS 35.060 **Ref. No. ISO/IEC 9075:1992/Cor.1:1996(E)**

Descriptors: data processing, text processing, factual data bases, information interchange, documentation, documents, logical structure, artificial languages, database languages, SQL.

Printed in Switzerland

Contents

Introduction

1. *Rationale: In the list of significant new features, the wording incorrectly implies that all the examples listed in item 10) are referential integrity facilities.*

On page xiv, in Significant new feature 10), replace "referential integrity" with "integrity".

2 Normative references

1. *Rationale: Editorial.*

Add the following reference after the reference to "ISO 8601:1988":

— ISO 8649:1988, *Information processing systems—Open Systems Interconnection—Service definition for the Association Control Service Element.*

2. *Rationale: The newly revised Ada language standard (ISO/IEC 8652:1995, Information technology—Programming languages—Ada) contains support for decimal-encoded numeric data and variable length character strings. The revised interface allows newly written applications in the revised Ada language access to these features of SQL; previously written Ada applications, conformant with the earlier Ada interface, are conformant with the revised interface.*

Replace the reference to ISO/IEC 8652:1987) with the following:

— ISO/IEC 8652:1995, *Information technology—Programming languages—Ada.*

3. *Rationale: Editorial.*

Add the following reference after the reference to ISO/IEC 8824:1990:

— ISO/IEC 9579-1:1993, *Information technology—Open Systems Interconnection—Remote Database Access—Part 1: Generic Model, Service and Protocol.*

Add the following reference after the reference to ISO/IEC 9899:

— ISO/IEC 10026-2:1996, *Information technology—Open Systems Interconnection—Distributed Transaction Processing—Part 2: OSI TP Service.*

3.3.4.3 Terms denoting rule requirements

1. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

In the first and second paragraphs, replace "*syntax error or access rule violation* (if this situation occurs during dynamic execution of an SQL-statement, then the exception that is raised is *syntax error or access rule violation in dynamic SQL statement*; if the situation occurs during direct invocation of an SQL-statement, then the exception that is raised is *syntax or access rule violation in direct SQL statement*)" with "*syntax error or access rule violation*".

4.2 Character strings

1. Rationale: Editorial.

In the second paragraph, replace the last sentence with the following:

Character sets defined by standards or by implementations reside in the Information Schema (named INFORMATION_SCHEMA) in each catalog, as do collations and translations defined by standards and collations, translations, and form-of-use conversions defined by implementations.

4.2.1 Character strings and collating sequences

1. Rationale: The following changes make the definitions of character set and collation descriptors more precise.

Replace the text on page 17 that occurs after the first paragraph with the following:

A character set is described by a character set descriptor. A character set descriptor includes:

— the name of the character set or character repertoire,

— if the character set is a character repertoire, then the name of the form-of-use,

— an indication of what characters are in the character set, and

— whether or not the character set uses the DEFAULT collation for its character repertoire, and,

— if the character set does not utilize the DEFAULT collation for its character repertoire, then the <translation name> contained in the character set's <translation collation>, if any, the <collation name> contained in the character set's <collate clause> or <limited collation definition>, if any, and, whether or not DESC was specified in the reference to the collation

For every character set, there is at least one collation. A collation is described by a collation descriptor. A collation descriptor includes:

— the name of the collation,

— the name of the character repertoire on which the collation operates,

— whether the collation has the NO PAD or the PAD SPACE attribute, and

— whether or not this collation utilizes the DEFAULT collation for its character repertoire,

— if the collation does not utilize the DEFAULT collation for its character repertoire, then the <translation name> contained in the collation's <translation collation>, if any, the <collation name> contained in the collation's <collation source>, if any, and whether or not DESC was specified in the definition of the collation.

4.4 Numbers

1. Rationale: Clarification.

Add the following sentence immediately before the heading of Subclause 4.4.1 Characteristics of Numbers:

A value described by a numeric data type descriptor is always signed.

4.5 Datetimes and intervals

1. Rationale: Clarification.

Add the following sentence before the paragraph starting "Every datetime ...":

A value described by an interval data type descriptor is always signed.

4.5.1 Datetimes

1. Rationale: Editorial.

In the penultimate paragraph on page 24, replace "Universal Coordinated Time" with "Coordinated Universal Time".

4.5.2 Intervals

1. Rationale: Editorial.

In Table 7, replace "<interval leading field precision" with "<interval leading field precision>" (two occurrences).

4.8 Columns

1. Rationale: Editorial.

In the penultimate paragraph on page 28, replace "<row value constructor expression>" with "<row value constructor element>".

4.9 Tables

1. Rationale: There is no named derived table other than a viewed table.

After the paragraph that begins with "A derived table descriptor describes a derived table.", delete the first item ("— if the table is named, then the name of the table;").

2. Rationale: There is no named derived table other than a viewed table.

After the paragraph that begins with "A view descriptor describes a view.", insert "— the name of the view, and" before the existing item.

4.10.2 Table constraints

1. Rationale: Editorial.

In the **Note**, replace "<match option>" with "<match type>".

2. Rationale: Editorial.

In the paragraph that begins with "A referential constraint is satisfied", replace "<match option>" with "<match type>".

4.18.1 Status parameters

1. Rationale: To insure that the value returned to the user in SQLSTATE is representative of the actual state of the transaction or SQL-statement.

Add the following as the last paragraph:

> For the purpose of choosing status parameter values to be returned, *exceptions* for transaction rollback have precedence over *exceptions* for statement failure. Similarly, completion condition *no data* has precedence over *warning*, which has precedence over *successful completion*. All *exceptions* have precedence over all completion conditions. The values assigned to SQLSTATE shall obey these precedence rules.

4.22.6 SQL-statements and transaction states

1. Rationale: No statement can be both transaction-initiating and not transaction-initiating.

In the first dashed list (of transaction-initiating SQL-statements), in the bulleted sublist of SQL-data statements, delete the entry for <dynamic select statement>.

4.24 SQL dynamic statements

1. Rationale: Editorial.

In the fourth paragraph, replace "<target specification>s" with "<simple value specification>s".

2. Rationale: Editorial.

In the eighth paragraph, replace the first occurrence of "<SQL statement>s" with "<SQL procedure statement>s", and replace the second occurrence of "<SQL statement>s" with "<embedded SQL statement>s".

4.26 Privileges

1. Rationale: Editorial.

In the fourth paragraph on page 52, replace "<module authorization identifier> is" with "<schema authorization identifier> is".

2. *Rationale: Provide missing rules that cover the acquisition of the necessary privileges to acquire the WITH GRANT OPTION on views through a grant to PUBLIC.*

Add the following before the antepenultimate paragraph of this Subclause:

The phrase *user privileges* refers to the set of privileges defined by the privilege descriptors whose grantee is either the identified <authorization identifier> or PUBLIC.

4.28 SQL-transactions

1. *Rationale: Clarification.*

In the paragraph that begins "In some environments (e.g., remote database access)", replace all occurrences of "SQL-environment" with "SQL-implementation".

4.29 SQL-connections

1. *Rationale: Editorial.*

Replace the second paragraph with the following:

An SQL-connection is an *active SQL-connection* if any SQL-statement that initiates or requires an SQL-transaction has been executed at its SQL-server via that SQL-connection during the current SQL-transaction.

2. *Rationale: Clarification.*

In the last sentence of the penultimate paragraph, replace "SQL-environment" with "SQL-implementation".

4.31 Client-server operation

1. *Rationale: Clarification.*

Replace the first sentence with the following:

As perceived by an SQL-agent, an SQL-implementation consists of one or more SQL-servers and one SQL-client through which SQL-connections can be made to them.

5.2 <token> and <separator>

1. *Rationale: The maximum length of an <identifier> is intended to be 128 characters.*

Replace Syntax Rule 8) with the following:

8) In a <regular identifier>, the number of <underscore>s plus the number of <identifier part>s shall be less than 128.

2. *Rationale: A <regular identifier> shall not contain any <quote> or <double quote>. Thus, a <delimited identifier> with a <delimited identifier body> containing a <quote> or <double quote> is not equivalent to any <regular identifier>.*

In Syntax Rule 13), delete the expression "(with all occurrences of <quote> replaced by <quote symbol> and all occurrences of <doublequote symbol> replaced by <double quote>)".

3. *Rationale: Correct the incorrect references to "<quote>" and "<quote symbol>" and delete the redundant references to "<double quote>"s and "<double quote symbol>"s in Syntax Rule 14.*

In Syntax Rule 14), delete "(with all occurrences of <quote> replaced by <quote symbol> and all occurrences of <doublequote symbol> replaced by <doublequote>)".

4. *Rationale: A <character representation> does not appear in a <regular identifier> or in a <delimited identifier body>.*

Replace Leveling Rule 2) a) with the following:

a) The number of <underscore>s plus the number of <identifier part>s contained in a <regular identifier> shall be less than 18.

Insert the following Leveling Rule 2) a.1):

a.1) The <delimited identifier body> of a <delimited identifier> shall not comprise more than 18 <delimited identifier part>s.

5.3 <literal>

1. *Rationale: Support changes to Subclause 6.10.*

In Format, replace the BNF for <date string>, <time string>, <timestamp string>, <interval string> with the following:

<date string> ::= <quote> <unquoted date string> <quote>

<time string> ::= <quote> <unquoted time string> <quote>

<timestamp string> ::= <quote> <unquoted timestamp string> <quote>

<interval string> ::= <quote> <unquoted interval string> <quote>

Add the following to Format:

<unquoted date string> ::= <date value>

<unquoted time string> ::= <time value> [<time zone interval>]

<unquoted timestamp string> ::= <unquoted date string> <space> <unquoted time string>

<unquoted interval string> ::= [<sign>] { <year-month literal> | <day-time literal> }

Add the following sentence at the end of General Rule 5):

If a <sign> is specified in both possible locations in an <interval literal>, the sign of the literal is determined by the normal mathematical interpretation of multiple sign operators, i.e., double negation results in a positive literal.

2. *Rationale: Ellipses were placed both in the definition of <separator> and in the usage of <separator> in several types of string literals. This leads to a small ambiguity when two separator characters follow one another. The following changes remove the ellipses in the latter of these two places:*

In Format, in each of the productions for <character string literal>, <national character string literal>, <bit string literal>, <hex string literal>, replace "<separator> ..." with "<separator>".

In Syntax Rules 1) - 3) replace "<separator> ..." with "<separator>".

5.4 Names and identifiers

1. *Rationale: Editorial.*

In Syntax Rule 1), replace "identified" with "indicated".

2. *Rationale: Add the rule for the comparison of <character set name>s missing since <character set name> was decoupled from <qualified name>.*

Add the following Syntax Rule:

10.5) Two <character set names> are equal if they have the same <SQL language identifier> and the same <schema name>, regardless of whether the <schema name>s are implicit or explicit.

6.1 <data type>

1. *Rationale: Clarify the error condition that should be returned if a datetime overflow occurs.*

Delete General Rule 6).

2. *Rationale: Clarification.*

Add the following Note after Syntax Rule 28):

Note: The length of interval data types is specified in the General Rules of 10.1, "<interval qualifier>".

6.2 <value specification> and <target specification>

1. *Rationale: Identify possible upward incompatibility.*

Add the following at the end of Syntax Rule 3):

Note: In an environment where the SQL-implementation conforms to Entry SQL, conforming SQL language that contains either:

a) a specified or implied <comparison predicate> that compares the <value specification> USER with a <value specification> other than USER, or

b) a specified or implied assignment in which the "value" (as defined in Subclause 9.2 Store assignment) contains the <value specification> USER

will become non-conforming in an environment where the SQL-implementation conforms to Intermediate SQL or Full SQL, unless the character repertoire of the implementation-defined character set in that environment is identical to the character repertoire of SQL_TEXT.

6.3 <table reference>

1. Rationale: Editorial.

In Syntax Rule 2) a), replace "with no intervening" with "without an intervening".

2. Rationale: Editorial.

In Syntax Rule 2) a), replace "The scope clause of the exposed" with "The scope of the exposed".

3. Rationale: Editorial.

In Syntax Rule 2) b), replace "with no intervening" with "without an intervening".

4. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

8.1) Let *T* be the table identified by the <table name> immediately contained in <table reference>. If the <table reference> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*. If the <table reference> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*, or *S* shall include a <schema element> that creates the descriptor of *T*.

5. Rationale: Editorial.

Delete the first sentence of Access Rule 1).

6.6 <numeric value function>

1. Rationale: Correct Format for <position expression> to support <bit value expression>s in addition to <character value expression>s.

In the Format replace the production for <position expression> with the following:

```
<position expression> ::=
        POSITION <left paren> <string value expression>
            IN <string value expression> <right paren>
```

Replace Syntax Rule 1) with the following:

1) If <position expression> is specified, then both <string value expression>s shall be <bit value expression>s or both shall be <character value expression>s having the same character repertoire.

Replace General Rules 1) and 2) with the following:

1) If <position expression> is specified and neither <string value expression> is the null value, then

   Case:

   a) If the first <string value expression> has a length 0, then the result is 1.

   b) If the value of the first <string value expression> is equal to an identical-length substring of contiguous characters or bits from the value of the second <string value expression>, then the result is 1 greater than the number of characters or bits within the value of the second <string value expression> preceding the start of the first such substring.

   c) Otherwise, the result is 0.

2) If <position expression> is specified and either <string value expression> is the null value, then the result is the null value.

6.8 <datetime value function>

1. *Rationale: The following change clarifies that <datetime value function> is effectively evaluated only once per SQL statement.*

Replace General Rule 3) with the following:

3) If an SQL-statement causes the evaluation of one or more <datetime value function>s, then all such evaluations are effectively performed simultaneously. The time of evaluation of the <datetime value function> during the execution of the SQL-statement is implementation-dependent.

6.10 <cast specification>

1. *Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.*

Add the following Syntax Rule:

8.1) If <domain name> is specified, then let *D* be the domain identified by the <domain name>. If the <cast specification> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <domain name> shall include the descriptor of *D*. If the <cast specification> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <domain name> shall include the descriptor of *D*, or *S* shall include a <schema element> that creates the descriptor of *D*.

2. *Rationale: Legalize a simpler format for character strings being cast to temporal data types.*

In General Rule 9) a) i), replace "rules for <literal>" with "rules for <literal> or for <unquoted date string>".

3. *Rationale: Clarify the error condition that should be returned if a datetime overflow occurs.*

Replace General Rule 9) a) ii) with the following:

ii) Otherwise,

1) If a <datetime value> does not conform to the natural rules for dates or times according to the Gregorian calendar, an exception condition is raised: *data exception—invalid datetime format.*

2) Otherwise, an exception condition is raised: *data exception—invalid character value for cast.*

4. *Rationale: Legalize a simpler format for character strings being cast to temporal data types.*

In General Rule 10) a) i), replace "rules for <literal>" with "rules for <literal> or for <unquoted time string>".

5. *Rationale: Clarify the error condition that should be returned if a datetime overflow occurs.*

Replace General Rule 10) a) ii) with the following:

ii) Otherwise,

1) If a <datetime value> does not conform to the natural rules for dates or times according to the Gregorian calendar, an exception condition is raised: *data exception—invalid datetime format.*

2) Otherwise, an exception condition is raised: *data exception—invalid character value for cast.*

6. *Rationale: Legalize a simpler format for character strings being cast to temporal data types.*

In General Rule 11) a) i), replace "rules for <literal>" with "rules for <literal> or for <unquoted timestamp string>".

7. *Rationale: Clarify the error condition that should be returned if a datetime overflow occurs.*

Replace General Rule 11) a) ii) with the following:

ii) Otherwise,

1) If a <datetime value> does not conform to the natural rules for dates or times according to the Gregorian calendar, an exception condition is raised: *data exception—invalid datetime format.*

2) Otherwise, an exception condition is raised: *data exception—invalid character value for cast.*

8. *Rationale: Legalize a simpler format for character strings being cast to temporal data types.*

In General Rule 12) b) i), replace "rules for <literal>" with "rules for <literal> or for <unquoted interval string>".

9. *Rationale: Clarify the error condition that should be returned if a datetime overflow occurs.*

Replace General Rule 12) b) ii) with the following:

ii) Otherwise,

1) If a <datetime value> does not conform to the natural rules for intervals according to the Gregorian calendar, an exception condition is raised: *data exception—invalid interval format.*

2) Otherwise, an exception condition is raised: *data exception—invalid character value for cast.*

6.12 <numeric value expression>

1. *Rationale: Editorial.*

In General Rule 2), replace "then the value of" with "then the result of".

6.14 <datetime value expression>

1. *Rationale: The expression "X AT TIME ZONE Y - Z" may be parsed as either "(X AT TIME ZONE Y) - Z" or "X AT TIME ZONE (Y - Z)". The following clarifies that it should be the former.*

Replace the Format for <time zone specifier> with the following:

```
<time zone specifier>::=
        LOCAL
      | TIME ZONE <interval primary>
```

2. *Rationale: Editorial.*

In Syntax Rule 4), replace "<interval value expression>" with "<interval primary>".

3. *Rationale: Clarify the meaning of the list of data types given in Syntax Rule 5).*

Add a Note following Syntax Rule 5) b) as follows:

Note: In the preceding Syntax Rule, TIME does not include TIME WITH TIME ZONE and TIMESTAMP does not include TIMESTAMP WITH TIME ZONE.

4. *Rationale: If <time zone> is implied, then <time zone specifier> shall not be specified.*

In General Rule 2), replace "specified or implied" with "specified".

5. *Rationale: Editorial.*

In General Rule 2), replace "<interval value expression>" with "<interval primary>".

In General Rule 4) b), replace "<interval value expression>" with "<interval primary>".

6.15 <interval value expression>

1. *Rationale: Clarification of the data type of a result.*

Replace Syntax Rule 3) with the following:

3) Case:

a) If the <interval value expression> simply contains an <interval qualifier> *IQ*, then the data type of the result is
INTERVAL *IQ*

b) If the <interval value expression> is an <interval term>, then the result of the <interval value expression> contains the same interval fields as the <interval primary>.

c) If <interval term.1> is specified, then the result contains every interval field that is contained in the result of either <interval value expression 1> or <interval term 1>, and, if both contain a seconds field, then the fractional seconds precision of the result is the greater of the two fractional seconds precisions.

Note: Interval fields are effectively defined by Table 5 - Fields in year-month INTERVAL items, and Table 6 Fields in day-time INTERVAL items.

7.1 <row value constructor>

1. *Rationale: To clarify the value of a <default specification> by reference to the General Rules of 11.5 <default clause> as is also done in 13.9.*

Replace General Rule 2) with the following:

2) The value of a <default specification> is determined according to the General Rules of Subclause 11.5, "<default clause>".

2. *Rationale: Editorial.*

In Leveling Rule 1) a), replace "<table value constructor>" with "<table value constructor> or an <overlaps predicate>".

3. *Rationale: The 1989 standard prohibited the use of a <row subquery> in a predicate. A leveling rule was omitted that would only allow this use of <row subquery> in Full level.*

Add the following to Leveling Rule 1):

a.1) A <row value constructor> shall not be a <row subquery>.

7.4 <from clause>

1. Rationale: Editorial.

In Syntax Rule 1)a), replace "with no intervening" with "without an intervening".

In Syntax Rule 1)b), replace "with no intervening" with "without an intervening".

In General Rule 1)a), replace "with no intervening" with "without an intervening".

In General Rule 1)b), replace "with no intervening" with "without an intervening".

2. Rationale: Editorial.

In Leveling Rule 2) a), replace "<table name>" with "<table reference>".

7.5 <joined table>

1. Rationale: Syntax Rule 6) [If NATURAL ...] does not adequately address the following:

i) the case when the set of corresponding columns is empty,

ii) the cases when either set of non-corresponding columns is empty,

iii) the case when a set of non-corresponding join columns contains columns with the same <column name>.

Syntax Rule 6) also conflicts with Syntax Rules 7), 8) and 9) with respect to the definition of the nullability characteristics of a column.

In Syntax Rule 6) d), replace "Let" by "If there is at least one corresponding join column, then let"

Replace Syntax Rule 6) e) with the following:

e) 1) If T_1 contains at least one column that is not a corresponding join column, then let SLT_1 be a <select list> of <derived column>s of the form:

$TA.C$

for every column C of T_1 that is not a corresponding join column, taken in order of their ordinal position in T_1.

2) If T_2 contains at least one column that is not a corresponding join column, then let SLT_2 be a <select list> of <derived column>s of the form:

$TB.C$

for every column C of T_2 that is not a corresponding join column, taken in order of their ordinal position in T_2.

Replace Syntax Rule 6) f) with the following:

f) Let the <select list> SL be defined as:

Case:

i) If all the columns of T_1 and T_2 are corresponding join columns, then let SL be $SLCC$.

ii) If T_1 contains no corresponding join columns and T_2 contains no corresponding join columns, then let SL be SLT_1, SLT_2.

iii) If T_1 contains no columns other than corresponding join columns, then let SL be $SLCC$, SLT_2.

iv) If T_2 contains no columns other than corresponding join columns, then let *SL* be *SLCC*, SLT_1

v) Otherwise, let *SL* be *SLCC*, SLT_1, SLT_2

The descriptors of the columns of the result of the <joined table>, with the possible exception of the nullability characteristics of the column, are the same as the descriptors of the columns of the result of
SELECT *SL* FROM TR_1, TR_2

2. *Rationale: General Rule 6) a) [If NATURAL is ...] uses the <select list> definitions SLCC, SLT_1 and SLT_2 which are not applicable to the table SN.*

Replace General Rule 6) a) with the following:

a) If NATURAL is specified or a <named columns join> is specified, then:

i) Let CS_i be a name for the *i*-th column of *S*. Column CS_i of *S corresponds* to the *i*-th column of T_1 if *i* is less than or equal to D_1. Column CS_j of *S corresponds* to the $(j\text{-}D_1)$-th column of T_2 for *j* greater than D_1.

ii) Let *SLN* be the <select list> derived from *SL* by replacing each <derived column> of the form
TA.C or *TB.C*
in *SL* by the name CS_i of the column of *S* that corresponds to the column *C* of T_1 or T_2, respectively.

iii) The result of the <joined table> is the multiset of rows resulting from
SELECT *SLN* FROM *SN*

7.6 <where clause>

1. *Rationale: Remove the ambiguity in Leveling Rule 2).*

Replace Leveling Rule 2) with:

2) The following restrictions apply for Entry SQL in addition to any Intermediate SQL restriction:

a) A <value expression> directly contained in the <search condition> shall not contain a <column reference> that references a <derived column> that generally contains a <set function specification>.

7.9 <query specification>

1. *Rationale: The rules for expansion of "<qualifier>.*" need to limit the expansion to the table identified by the <qualifier>, rather than including the entire <table expression>.*

Replace Syntax Rule 4) with the following:

4) If the <select sublist> "<qualifier>.*" is specified, then let *Q* be the <qualifier> of that <select sublist>. *Q* shall be a <table name> or <correlation name> exposed by a <table reference> immediately contained in the <from clause> of *T*. Let *TQ* be the table associated with *Q*. That <select sublist> is equivalent to a <value expression> sequence in which each <value expression> is a <column reference> *CR* that references a column of *TQ* that is not a common column of a <joined table>. Each column of *TQ* that is not a common column of a <joined table> shall be referenced exactly once. The columns shall be referenced in the ascending order of their ordinal positions with *TQ*.
Note: *common column of a <joined table>* is defined in Subclause 7.5, "<joined table>".

2. *Rationale: Clarification of the definition of outer reference.*

Replace Syntax Rule 7) with the following:

7) If *T* is a grouped table, then in each <value expression>, each <column reference> that references a column of *T* shall reference a grouping column or be specified within a <set function specification>. If *T* is not a grouped table and any <value expression> contains a <set function specification> that contains a reference to a column of *T* or any <value expression> directly contains a <set function specification> that does not contain an outer reference, then in each <value expression>, each <column reference> that references a column of *T* shall be specified within a <set function specification>.

3. *Rationale: Editorial.*

In Syntax Rule 9) a), replace the two occurrences of "*C*" with "*CN*".

In Syntax Rule 9) b), replace "*C*" with "the <column name> of the column designated by the <column reference>".

4. *Rationale: A <dynamic parameter specification> is possibly nullable.*

In Syntax Rule 10), insert " a <dynamic parameter specification>," after "an <indicator variable>,".

5. *Rationale: Editorial.*

In General Rule 1) a) ii) 1), replace "the table is the result of the <query specification>" with "the result of the <query specification> is that table".

In General Rule 1) b) ii) 1), replace "the table is the result of the <query specification>" with "the result of the <query specification> is that table".

7.10 <query expression>

1. *Rationale: Editorial.*

Replace Syntax Rule 14) with the following:

14) The *simply underlying tables* of a <query expression> are the tables identified by those <table name>s, <query specification>s, and <derived table>s contained in the <query expression> without an intervening <derived table> or an intervening <join condition>.

7.11 <scalar subquery>, <row subquery>, and <table subquery>

1. *Rationale: Editorial.*

Replace Leveling Rule 1) with the following:

1) The following restrictions apply for Intermediate SQL:

None.

2. *Rationale: SQL-89 prohibited the use of a subquery with degree greater than one as the argument of an <exists predicate>. It was intended to relax this restriction in Intermediate SQL, but the required Leveling Rule for Entry SQL was inadvertently omitted.*

Add the following Leveling Rule:

b.1) If a <table subquery> is simply contained in an <exists predicate>, then the <select list> of the <query specification> directly contained in the <table subquery> shall comprise either an <asterisk> or a single <derived column>.

8.3 <between predicate>

1. Rationale: Editorial.

In Syntax Rule 2), replace "two" with "three".

8.7 <quantified comparison predicate>

1. Rationale: Editorial.

In Syntax Rule 2), replace "columns" with "corresponding columns".

9.1 Retrieval assignment

1. Rationale: Clarification of the treatment of datetimes.

Replace General Rule 3) l) with the following:

l) If the data type of *T* is datetime, then:

Case:

i) If *V* is a member of the data type of *T*, then *T* is set to *V*.

ii) If a member of the data type of *T* can be obtained from *V* by rounding, then *T* is set to that value.

iii) Otherwise, an exception condition is raised: *data exception—datetime field overflow*.

9.2 Store assignment

1. Rationale: Clarification of the treatment of datetimes.

Replace General Rule 3) l) with the following:

l) If the data type of *T* is datetime, then:

Case:

i) If *V* is a member of the data type of *T*, then *T* is set to *V*.

ii) If a member of the data type of *T* can be obtained from *V* by rounding, then *T* is set to that value.

iii) Otherwise, an exception condition is raised: *data exception—datetime field overflow*.

10.1 <interval qualifier>

1. Rationale: Editorial.

Replace "Syntax rules" with "Syntax Rules".

2. Rationale: Clarification of the definition of "significance" of fractional seconds precision.

Replace Syntax Rule 1) with:

1) There is a significance of ordering of <datetime field>s. In order from most significant to least significant, the ordering is: YEAR, MONTH, DAY, HOUR, MINUTE, and SECOND. A <start field> or <single datetime field> with an <interval leading field precision> *i* is more significant than a <start field> or <single datetime field> with an <interval leading field precision> *j* if *i*<*j*. An <end field> or <single datetime field> with an <interval fractional seconds precision> *i* is less significant than an <end field> or <single datetime field> with an <interval fractional seconds precision> *j* if *i*>*j*.

3. Rationale: There is a contradiction between Syntax Rule 2) a) and General Rule 4. The Syntax Rule requires the <start field> to be more significant than the <end field> while the General Rule allows them to be the same. The Syntax Rule is modified to remove the contradiction.

Replace Syntax Rule 2) with the following:

2) If TO is specified, then:
a) <start field> shall be more significant than, or equally significant to, the <end field>,
b) if <start field> specified MONTH, then <end field> shall specify MONTH, and
c) if <start field> specified YEAR, then <end field> shall specify either YEAR or MONTH.

10.4 <character set specification>

1. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

3.1) Let *C* be the <character set name> contained in the <character set specification>. If the <character set specification> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <character set name> shall include the descriptor of *C*. If the <character set specification> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <character set name> shall include the descriptor of *C*, or *S* shall include a <schema element> that creates the descriptor of *C*.

10.5 <collate clause>

1. Rationale: Clarify that references to non-existing objects are only allowed within the same <schema definition>.

Add the following Syntax Rule:

0.1) Let *C* be the <collation name> contained in the <collate clause>. If the <collate clause> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <collation name> shall include the descriptor of *C*. If the <collate clause> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <collation name> shall include the descriptor of *C*, or *S* shall include a <schema element> that creates the descriptor of *C*.

10.6 <constraint name definition> and <constraint attributes>

1. Rationale: Editorial.

In the **Note** following General Rule 3), replace "<SQL statement>" with "SQL-statement".

2. Rationale: Editorial.

In Leveling Rule 2) a), replace "Intermediate" with "Entry".

11.1 <schema definition>

1. Rationale: Editorial.

In Leveling Rule 2) a), replace "Intermediate" with "Entry".

11.2 <drop schema statement>

1. Rationale: Editorial.

In General Rule 5), replace "collation definition" with "collating sequence".

11.4 <column definition>

1. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

4.1) If the <column definition> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <domain name> shall include the descriptor of *D*. If the <column definition> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <domain name> shall include the descriptor of *D*, or *S* shall include a <schema element> that creates the descriptor of *D*.

2. Rationale: Editorial.

In Access Rule 1), replace "*D*.." with "*D*.".

3. Rationale: Editorial.

In Leveling Rule 2) c), replace "Intermediate" with "Entry".

11.5 <default clause>

1. Rationale: Editorial.

In Syntax Rule 1), replace "<alter domain definition>" with "<alter domain statement>".

2. Rationale: The following changes correct the fact that the current rules cause the value of, for example, CURRENT_USER to be evaluated at the time of the <table definition> or <column definition> and forever more used as the default value for the associated column in the table. The value of CURRENT_USER, etc., and the <datetime value function>s are being put directly into the column or domain descriptor at the time that the <default clause> is processed. It was intended that those things be evaluated at the time that the row was inserted into the table (or updated in the table). Additionally the new rule 2.1 defines how a default value of null is represented in COLUMN_DEFAULT in the DOMAINS and COLUMNS base table. The new Case c) of General Rule 2) covers the case of a derived table.

Replace General Rules 1) and 2) with the following General Rules:

1) The default value inserted in the column descriptor, if the <default clause> is to apply to a column, or in the domain descriptor, if the <default clause> is to apply to a domain, is the <default option>. If the subject data type is bit string with fixed length, the <default clause> specifies a <bit string literal>, and the length of the <bit string literal> is less than the fixed length of the column, then a completion condition is raised: *warning—implicit zero-bit padding*.

2) The default value of a column is

Case:

a) If the column descriptor of a column indicates a default value derived from a <default option>, then the value in the column descriptor;

b) If the column descriptor includes a domain name that identifies a domain descriptor that includes a default value derived from a <default option>, then the value in the domain descriptor;

c) If the default value is for a column *C* of a candidate row for insertion into or update of a derived table *DT* and *C* has a single counterpart column *CC* in a leaf generally underlying table of *DT*, then the default value of *CC* obtained by applying the General Rules of this Subclause.

d) Otherwise, the null value.

2.1) When a default value is required for a column in a row of a table, the default value for the column is derived from the default value in the appropriate column descriptor or domain descriptor. Let *D* be that descriptor.

Case:

a) If *D* contains NULL, then the null value.

b) If *D* contains a <literal>, then

Case:

i) If the subject data type is numeric, then the numeric value of the <literal>.

ii) If the subject data type is character string with variable length, then the value of the <literal>.

iii) If the subject data type is character string with fixed length, then the value of the <literal>, extended as necessary on the right with <space>s to the length in characters of the subject data type.

iv) If the subject data type is bit string with variable length, then the value of the <literal>.

v) If the subject data type is bit string with fixed length, then the value of the <literal> extended as necessary on the right with 0-valued bits to the length of the subject data type.

vi) If the subject data type is datetime or interval, then the value of the <literal>.

c) If the column or domain descriptor specifies CURRENT_USER, SESSION_USER, or SYSTEM_USER, then

Case:

i) If the subject data type is character string with variable length, then the value obtained by an evaluation of CURRENT_USER, SESSION_USER, or SYSTEM_USER at the time that the default value is required.

ii) If the subject data type is character string with fixed length, then the value obtained by an evaluation of CURRENT_USER, SESSION_USER, or SYSTEM_USER at the time that the default value is required, extended as necessary on the right with <space>s to the length in characters of the subject data type.

d) If the column or domain descriptor contains a <datetime value function>, then the value of an evaluation of the <datetime value function> at the time that the default value is required.

e) Otherwise, the null value.

3. *Rationale: The following helps define the action taken when a default value cannot be represented in the Information Schema.*

Add the following General Rule:

3.1) If the <default clause> is contained in an <schema definition>, and if the character representation of the <default option> cannot be represented in the Information Schema without truncation, then a completion indication is raised: *warning—default option too long for information schema.*

11.6 <table constraint definition>

1. *Rationale: Editorial.*

In Leveling Rule 2) a), replace "Intermediate" with "Entry".

11.8 <referential constraint definition>

1. *Rationale: Clarify that references to non-existing objects are only allowed within the same <schema definition>.*

Add the following Syntax Rule:

9.1) Let *T* be the referenced table. If the <referential constraint definition> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*. If the <referential constraint definition> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*, or *S* shall include a <schema element> that creates the descriptor of *T*.

11.9 <check constraint definition>

1. *Rationale: Editorial.*

Replace Syntax Rule 1) with the following:

1) The <search condition> shall not contain a <parameter specification>, a <variable specification>, or a <dynamic parameter specification>.

11.11 <add column definition>

1. *Rationale: Editorial.*

Add the following General Rule:

0.1) Let *T* be the table identified by the <table name> immediately contained in the containing <alter table statement>.

11.16 <add table constraint definition>

1. Rationale: Provide General Rules for maintaining the nullability characteristic in a column definition.

Add the following General Rule:

2.1) Let *TC* be the table constraint added to *T*. If *TC* causes some column *CN* to be known not nullable and no other constraint causes *CN* to be known not nullable, then the nullability characteristic of the column descriptor of *CN* is changed to known not nullable.
Note: The nullability characteristic is defined in Subclause 4.8, "Columns".

11.17 <drop table constraint definition>

1. Rationale: Provide General Rules for maintaining the "nullability characteristic" in a column definition.

Add the following General Rule:

3.1) If *TC* causes some column *CN* to be known not nullable and no other constraint causes *CN* to be known not nullable, then the nullability characteristic of the column descriptor of *CN* is changed to possibly nullable.
Note: The nullability characteristic is defined in Subclause 4.8, "Columns".

11.18 <drop table statement>

1. Rationale: A <drop behavior> of RESTRICT shall not be allowed if there are referential constraints that reference the table being dropped, and shall not consider check constraints of the table being dropped.

Replace Syntax Rule 4) with the following:

4) If RESTRICT is specified, then *T* shall not be referenced in the <query expression> of any view descriptor, the <search condition> of any table check constraint descriptor of any table other than *T*, or the referenced table of any referential constraint descriptor of any table other than *T*.
Note: If CASCADE is specified, then such referencing objects will be dropped by the execution of the <revoke statement> specified in the General Rules of this Subclause.

11.19 <view definition>

1. *Rationale: A predicate specified for LOCAL CHECK OPTION is tested only for rows that would pass all lower level predicates. This does not reflect the correct semantics for LOCAL.*

Replace General Rule 9) with the following:

9) Let *V1* be an updatable view. Let *V2* be the simply underlying table of the simply underlying table of the <query expression> of *V1*. Let *S* be the set of check conditions of *V1* that is used for the validation of WITH CHECK OPTION. Each check condition of *S* is a primary <search condition> of a view. The primary <search condition> of a view is the <search condition> specified in the <where clause> of the <table expression> directly contained in the <query expression> of the view. If the <table expression> does not contain a <where clause>, then the view has an implicit primary <search condition> that is always true.

Case:

a) If *V2* is a base table and the descriptor of *V1* does not include WITH CHECK OPTION, then *S* is empty.

b) If *V2* is a base table and the descriptor of *V1* includes any form of WITH CHECK OPTION, then *S* consists of the primary <search condition> of *V1*.

c) If *V2* is a view and the descriptor of *V1* does not include WITH CHECK OPTION, then *S* is identical to the set of check conditions of *V2*.

d) If *V2* is a view and the descriptor of *V1* contains WITH CASCADED CHECK OPTION, then *S* consists of the primary <search condition> of *V1* and the primary <search condition> of each view that is a generally underlying table of *V1*.

e) If *V2* is a view and the descriptor of *V1* contains WITH LOCAL CHECK OPTION, then *S* consists of the primary <search condition> of *V1* and the set of check conditions of *V2*.

Replace General Rule 11) with the following:

11) During an update operation on *V1*, every check condition in *S* is applied to every inserted or updated row. If any check condition is not true, then an exception condition is raised: *with check option violation*.

11.21 <domain definition>

1. *Rationale: In General Rule 3), a domain constraint descriptor is referenced, but it has not been created.*

Insert the following General Rule:

2.1) For every <domain constraint> immediately contained in the <domain definition>:
A domain constraint descriptor is created that describes the domain constraint being defined. The domain constraint descriptor includes the <constraint name> contained in the explicit or implicit <constraint name definition>. The domain constraint descriptor includes an indication of whether the constraint is deferrable or not deferrable, and whether the initial constraint mode of the constraint is *deferred* or *immediate*. The domain constraint descriptor includes the <search condition> immediately contained in the <check constraint definition> immediately contained in the <domain constraint>.

2. *Rationale: The following change specifies that when overriding the collation of a contained character set, a grantable USAGE privilege on that overriding collation is not needed to acquire a grantable USAGE privilege on the domain.*

Replace General Rule 4) with the following:

4) A privilege descriptor is created that defines the USAGE privilege on this domain to the <authorization identifier> of the <schema> or <module> in which the <domain definition> appears. This privilege is grantable if and only if the applicable privileges include a grantable REFERENCES privilege for each <column reference> included in the domain descriptor and a grantable USAGE privilege for each <domain name>, <collation name>, <character set name>, and <translation name> contained in the <search condition> of any domain constraint descriptor included in the domain descriptor. The grantor of the privilege descriptor is set to the special grantor value "_SYSTEM".

11.25 <add domain constraint definition>

1. *Rationale: Provide General Rules for maintaining the "nullability characteristic" in a column definition.*

Add the following General Rule:

1.1) If *DC* causes some column *CN* to be known not nullable and no other constraint causes *CN* to be known not nullable, then the nullability characteristic of the column descriptor of *CN* is changed to known not nullable.
Note: The nullability characteristic is defined in Subclause 4.8, "Columns".

11.26 <drop domain constraint definition>

1. *Rationale: Provide General Rules for maintaining the "nullability characteristic" in a column definition.*

Add the following General Rule:

2.1) If *DC* causes some column *CN* to be known not nullable and no other constraint causes *CN* to be known not nullable, then the nullability characteristic of the column descriptor of *CN* is changed to possibly nullable.
Note: The nullability characteristic is defined in Subclause 4.8, "Columns".

11.27 <drop domain statement>

1. *Rationale: When a domain is destroyed it is necessary to destroy its domain constraint descriptors.*

Replace General Rule 3) with the following:

3) The identified domain is destroyed by destroying its descriptor, its data type descriptor, and its domain constraint descriptors.

11.28 <character set definition>

1. *Rationale: Editorial.*

In Syntax Rule 3), replace "character set descriptor" with "character set".

11.29 <drop character set statement>

1. *Rationale: The following change adds missing restrictions to deal with the convention that absence of a <drop behavior> production in the BNF of a schema manipulation statement indicates that the respective schema manipulation statement cannot drop objects while other objects depend on their existence.*

Replace Syntax Rule 3) with the following:

3) *C* shall not be referenced in the <query expression> of any view descriptor or in the <search condition> of any constraint descriptor, nor be included in any collation descriptor or translation descriptor or schema descriptor, nor be referenced in the <module character set specification> of any module, nor be included in any column's or any domain's data type descriptor.

11.30 <collation definition>

1. *Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.*

Add the following Syntax Rule:

10.1) If <translation name> is specified and the <collation definition> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <translation name> shall include the descriptor of *T*. If the <collation definition> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <translation name> shall include the descriptor of *T*, or *S* shall include a <schema element> that creates the descriptor of *T*.

11.31 <drop collation statement>

1. *Rationale: By convention, the absence of a <drop behavior> in a schema manipulation statement that executes a drop implies that no object that is dependent upon a dropped object will be dropped. In the absence of the following, constraints and views would be dropped when the <revoke statement> effectively executed by the General Rules of 11.31 <drop collation statement> is executed. Additionally, the COLLATION may not be dropped if it is used in a view definition or in a constraint definition.*

Add the following Syntax Rule:

2.1) *C* shall not be referenced in the <query expression> included in any view descriptor or in the <search condition> included in any constraint descriptor.

2. *Rationale: The following changes address the problems that string manipulations are specified on descriptors that are not defined to contain character strings, and that a convention is violated that absence of a <drop behavior> production implies that the Schema Manipulation Language fails if it causes other objects to be dropped. To achieve this, the defined contents of descriptors must be altered. Then, General Rules are reordered such that the effectively executed revoke statement happens after any domain descriptors that depend upon the collation being dropped have been altered to no longer have that dependency.*

Replace all the General Rules with the following:

1) For every collation descriptor *CD* that includes *CN*, *CD* is modified such that it does not include *CN*. If *CD* does not include any translation name, then *CD* is modified to indicate that it utilizes the DEFAULT collation for its character repertoire.

2) For every character set descriptor *CSD* that includes *CN*, *CSD* is modified such that it does not include *CN*. If *CSD* does not include any translation name, then *CSD* is modified to indicate that it utilizes the DEFAULT collation for its character repertoire.

3) For every column descriptor or domain descriptor *DD* that includes *CN*, *DD* is modified such that it does not contain *CN*.
Note: This causes the column or domain that *DD* is the descriptor for to revert to the default collation for its character set.

4) Let *A* be the current <authorization identifier>. The following <revoke statement> is effectively executed with a current <authorization identifier> of "_SYSTEM" and without further Access Rule checking:

REVOKE USAGE ON COLLATION *CN* FROM A CASCADE

5) The descriptor of *C* is destroyed.

11.32 <translation definition>

1. Rationale: Editorial.

In Syntax Rule 5), replace "identify a" with "identify some".

11.34 <assertion definition>

1. Rationale: Editorial.

Replace Syntax Rule 4) with the following:

4) The <search condition> shall not contain a <parameter specification>, a <variable specification>, or a <dynamic parameter specification>.

11.36 <grant statement>

1. *Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.*

Add the following Syntax Rule:

3.1) If the <grant statement> is not contained in a <schema definition>, then the schema identified by the explicit or implicit qualifier of the <object name> shall include the descriptor of *O*. If the <grant statement> is contained in a <schema definition> *S*, then the schema identified by the explicit or implicit qualifier of the <object name> shall include the descriptor of *O*, or *S* shall include a <schema element> that creates the descriptor of *O*.

2. *Rationale: Editorial.*

In General Rule 7), replace "<grantor>" with "grantor" and replace "<object>" with "object".

3. *Rationale: Provide missing rules that cover the acquisition of the necessary privileges to acquire the WITH GRANT OPTION on views through a grant to PUBLIC.*

In General Rule 7), replace "if *G* has been granted" with "if the user privileges of *G* contain the".

4. *Rationale: The following changes addresses the fact that <grant statement> and <revoke statement> are inconsistent with <collation definition> with regard to their treatment of the grantability of USAGE privilege on collations. The first change makes grantability of USAGE privilege against a domain more consistent with that of the grantability of USAGE privilege against a column, while the second change makes a collation USAGE privilege grantable only when the owner of that collation has a grantable USAGE privilege on both its contained translation, if any, and on its contained character set. For both changes, missing rules are provided that cover the acquisition of the necessary privileges to acquire the WITH GRANT OPTION on views through a grant to PUBLIC.*

Replace General Rule 8) with the following:

8) For every <grantee> *G* and for every domain *D1* owned by *G*, if the user privileges of *G* contain the REFERENCES privilege WITH GRANT OPTION on every column referenced in the <search condition> included in a domain constraint descriptor included in the domain descriptor of *D1* and a grantable USAGE privilege on all domains, character sets, collations, and translations whose <domain name>s, <character set name>s, <collation name>s, and <translation name>s, respectively, are included in the domain descriptor, then for every privilege descriptor with <privileges> USAGE, a grantor of "_SYSTEM", object *D1*, and <grantee> *G* that is not grantable, the following <grant statement> is effectively executed with a current <authorization identifier> of "_SYSTEM" and without further Access Rule checking:

GRANT USAGE ON DOMAIN *D1* TO *G* WITH GRANT OPTION

Replace General Rule 9) with the following:

9) For every <grantee> *G* and for every collation *C1* owned by *G*, if the user privileges of *G* contain a grantable USAGE privilege for the character set name included in the collation descriptor of *C1* and a grantable USAGE privilege for the translation name, if any, included in the collation descriptor of *C1*, then for every privilege descriptor with a <privileges> USAGE, a grantor of "_SYSTEM", object of *C1*, and <grantee> *G* that is not grantable, the following <grant statement> is effectively executed with a current <authorization identifier> of "_SYSTEM" and without further Access Rule checking:

GRANT USAGE ON COLLATION *C1* TO *G* WITH GRANT OPTION

In General Rule 10), replace "USAGE privilege of *G* for every character set identified by a <character set specification> contained in the <translation definition> of *T1* is grantable" with "user privileges of *G* contain a grantable USAGE privilege for every character set identified by a <character set specification> contained in the <translation definition> of *T1*".

5. *Rationale: Editorial.*

In General Rule 10), replace "<grantor>" with "grantor" and replace "<object>" with "object".

6. *Rationale: Editorial.*

Replace General Rule 11) with the following:

11) If <table name> is specified, then for each view *V* owned by some <grantee> *G* such that *T* or some column *CT* of *T* is referenced in the <query expression> of *V*, let RT_i, for *i* ranging from 1 to the number of tables identified by the <table reference>s contained in the <query expression> of *V*, be the <table name>s of those tables. For every column *CV* of *V*:

a) Let CRT_{ij}, for *j* ranging from 1 to the number of columns of RT_i that are underlying columns of *CV*, be the <column name>s of those columns.

b) If WITH GRANT OPTION was specified, then let *WGO* be "WITH GRANT OPTION"; otherwise, let *WGO* be a zero-length string.

c) If, following successful execution of the <grant statement>, the user privileges of *G* will contain a REFERENCES(CRT_{ij}) privilege for all *i* and for all *j*, and will contain a REFERENCES privilege on some column of RT_i, for all *i*, then the following <grant statement> is effectively executed as though the current <authorization identifier> were "_SYSTEM" and without further Access Rule checking:

GRANT REFERENCES (*CV*) ON *V* TO *G WGO*

11.37 <revoke statement>

1. *Rationale: Editorial.*

In Syntax Rule 6), replace "A privilege descriptor is" with "A privilege descriptor *P* is".

2. *Rationale: The following change address problems in the definition of when a privilege descriptor D is allowed to be created by a grant permitted by P, so as to properly deal with REFERENCES privilege against views.*

Replace Syntax Rule 7) b) iv) 2) with the following:

2) *P* and *D* are both column privilege descriptors, the privilege descriptor *D* identifies a <column name> *CVN* explicitly or implicitly contained in the <view column list> of a <view definition> *V* and either:

A) *V* is an updatable view. For each column *CV* identified by a <column name> *CVN*, there is a corresponding column in the underlying table of the <query expression> *TN*. Let *CTN* be the <column name> of the column of the <query expression> from which *CV* is derived. The action for *P* is UPDATE or INSERT and the identified column of *P* is *TN.CTN*; or

B) For every table *T* identified by a <table reference> contained in the <query expression> of *V* and for every column *CT* that is a column of *T* and an underlying column of *CV*, the action for *P* is REFERENCES and either the identified column of *P* is *CT* or the identified table of *P* is *T*.

3. *Rationale: The following change modifies the definition of "allowed to be created by" to reflect a collations dependency upon contained translations.*

Replace Syntax Rule 7) b) iv) 4) with the following:

4) The privilege descriptor *D* identifies the <collation name> of a <collation definition> *CO* and the identified character set name of *P* is included in the collation descriptor for *CO* or the identified translation name of *P* is included in the collation descriptor for *CO*.

4. *Rationale: The following change fixes the definition of when a privilege descriptor D is allowed to be created by a grant permitted by P, as that definition doesn't deal with USAGE privilege against domains properly.*

Add the following to Syntax Rule 7):

b.1) The following conditions hold:

i) The privilege descriptor for *D* indicates that its grantor is the special grantor value "_SYSTEM";

ii) The grantee of *P* is the owner of the domain identified by *D*, or the grantee of *P* is PUBLIC, and

iii) The privilege descriptor *D* identifies the <domain name> of a <domain definition> *DO* and either the column privilege descriptor *P* has an action of REFERENCES and identifies a column referenced in the <search condition> included in the domain descriptor for *DO*, or the privilege descriptor *P* has an action of USAGE and identifies a domain, collation, character set, or translation whose <domain name>, <collation name>, <character set name> or <translation name>, respectively, is contained in the <search condition> of the domain descriptor for *DO*.

5. *Rationale: The following changes deal with the need that when determining whether or not the owner of a view has a grantable REFERENCES privilege on a particular column of a view, the owner must have a grantable REFERENCES privilege on every column that is an underlying column of that column.*
This rule updates the definition of a modified privilege descriptor to deal with the case where revocation of REFERENCES privilege on one or more columns of a table RTi impacts one or more columns CVN of view V.

Replace Syntax Rule 10) c) with the following:

c) Case:

i) The following conditions hold:

1) *P* is not a REFERENCES column privilege descriptor that identifies a <column name> *CVN* explicitly or implicitly contained in the <view column list> of a <view definition> *V*, and;

2) Let *XO* and *XA* respectively be the identifier of the object identified by a privilege descriptor *X* and the action of *X*. Within the set of privilege descriptors upon which *P* directly depends, there exists some *XO* and *XA* for which the set of identified privilege descriptors unioned with the set of modified privilege descriptors include all privilege descriptors specifying the grant of *XA* on *XO* WITH GRANT OPTION.

ii) The following conditions hold:

1) *P* is a REFERENCES column privilege descriptor that identifies a column *CV* named by a <column name> *CVN* explicitly or implicitly contained in the <view column list> of a <view definition> *V*, and;

2) Let *SP* be the set of privileges upon which *P* directly depends. For every table *T* identified by a <table reference> contained in the <query expression> of *V*, let *RT* be the <table name> of *T*. There exists a column *CT* whose <column name> is *CRT*, such that:

A) *CT* is a column of *T* and an underlying column of *CV*, and

B) Every privilege descriptor *PD* that is the descriptor of some member of *SP* that specifies REFERENCES on *CRT* WITH GRANT OPTION is either an identified privilege descriptor for *CRT* or a modified privilege descriptor for *CRT*.

and

6. *Rationale: The following rule deals with the case where, due to a <revoke statement>, all REFERENCE privileges are lost against some underlying table of V, because in that instance all REFERENCE privileges should be lost against view V. The change above only deals with loss of grantability of REFERENCE against particular columns of V. To deal with this problem, it is necessary to define a new case where a privilege descriptor may become abandoned. [new item c)]. Also Syntax Rule 11) is structured as a list, which implies that all of the items in the list must be true for the privilege to be abandoned. This is inappropriate in that item b) is mutually exclusive with the new item c) i). Also indicate that the definition of abandoned doesn't depend upon whether or not GRANT OPTION FOR was specified.*

Replace Syntax Rule 11) with the following:

11) A privilege descriptor *P* is *abandoned* if:

Case:

a) It is not an independent node, and *P* is not itself an identified or a modified privilege descriptor, and there exists no path to *P* from any independent node other than paths that include an identified privilege descriptor or a modified privilege descriptor.

b) *P* is a SELECT column privilege descriptor and there exists a SELECT table privilege descriptor *X* with the same grantee, grantor, catalog name, schema name, and table name, and *X* is an abandoned privilege descriptor.

c) The following conditions hold:

i) *P* is a REFERENCES column privilege descriptor that identifies a <column name> *CVN* explicitly or implicitly contained in the <view column list> of a <view definition> *V*; and,

ii) Letting *SP* be the set of privileges upon which *P* directly depends, either:

1) There exists some table name *RT* such that:

A) *RT* is the name of the table identified by some <table reference> contained in the <query expression> of *V*, and

B) For every column privilege descriptor *CPD* that is the descriptor of some member of *SP* that specifies REFERENCES on *RT*, *CPD* is either an identified privilege descriptor for *RT* or an abandoned privilege descriptor for *RT*.

or

2) There exists some column name *CRT* such that:

A) *CRT* is the name of some column of the table of some <table reference> contained in the <query expression> of *V*, and

B) For every column privilege descriptor *CPD* that is the descriptor of some member of *SP* that specifies REFERENCES on *CRT*, *CPD* is either an identified privilege descriptor for *CRT* or an abandoned privilege descriptor for *CRT*.

7. *Rationale: The following changes address an inconsistency in that to have grantable USAGE privilege on a column that specifies <collate clause>, its owner doesn't need grantable USAGE privilege on that overriding collation. But, to do the same for a domain, the owner needs grantable USAGE privilege on the overriding collation. The following changes also provide a more consistent treatment of the <drop behavior> of the <revoke statement>.*

Replace Syntax Rule 17) with the following:

17) For every domain descriptor *DO* contained in *S1*, *DO* is said to be *lost* if the destruction of all abandoned privileged descriptors, and, if GRANT OPTION FOR is not specified, all identified privilege descriptors would result in *A1* no longer having USAGE privilege on any character set included in the data type descriptor included in *DO*.

Replace Syntax Rule 18) with the following:

18) For every table descriptor *TD* contained in *S1*, for every column descriptor *CD* included in *TD*, *CD* is said to be *lost* if either:

a) The destruction of all abandoned privilege descriptors, and, if GRANT OPTION FOR is not specified, all identified privilege descriptors would result in *A1* no longer having USAGE privilege on any character set included in the data type descriptor included in *CD*, or

b) The name of the domain *DN* included in *CD*, if any, identifies a lost domain descriptor and the destruction of all abandoned privilege descriptors, and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *A1* no longer having USAGE privilege on any character set included in the data type descriptor of the domain descriptor of *DN*.

Add the following Syntax Rules:

18.1) For every module *MO*, let *G* be the <module authorization identifier> that owns *MO*. *MO* is said to be *lost* if the destruction of all abandoned privilege descriptors, and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *G* no longer having USAGE privilege on the character set referenced in the <module character set specification> of *MO*.

18.2) Let *SD* be the descriptor of the schema *S1*. *SD* is said to be *lost* if the destruction of all abandoned privilege descriptors, and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *A1* no longer having USAGE privilege on the default character set included in the schema descriptor *SD*.

18.3) For every domain descriptor *DO* contained in *S1*, *DO* is said to be *impacted* if *DO* is not lost, and the destruction of all abandoned privilege descriptors and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *A1* no longer having USAGE privilege on the collation whose <collation name> is contained in *DO*.

18.4) For every column descriptor *CD* contained in a table descriptor contained in *S1*, *CD* is said to be *impacted* if *CD* is not lost and the destruction of all abandoned privilege descriptors and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *A1* no longer having USAGE privilege on the collation whose <collation name> is contained in *CD*.

18.5) For every collation descriptor CN contained in S1, CN is said to be impacted if the destruction of all abandoned privilege descriptors and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in A1 no longer having USAGE privilege on the collation whose <collation name> is contained in CN as the name of the collation's <collation source>.

18.6) For every character set descriptor *CSD* contained in *S1*, *CSD* is said to be *impacted* if the destruction of all abandoned privilege descriptors and, if GRANT OPTION FOR is not specified, all identified privilege descriptors, would result in *A1* no longer having USAGE privilege on the collation whose <collation name> is contained in *CSD*.

18.7) If RESTRICT is specified, then there shall be no abandoned privilege descriptors, abandoned view descriptors, abandoned table constraint descriptors, abandoned assertion descriptors, abandoned domain constraint descriptors, lost domain descriptors, lost column descriptors, lost module descriptors, lost schema descriptors, impacted domain descriptors, impacted column descriptors, impacted collation descriptors, or impacted character set descriptors.

18.8) If CASCADE is specified, then the impact on a module that is determined to be a lost module is implementation-defined.

Delete General Rule 7).

8. *Rationale: The following changes specify the necessary actions against lost objects, given the changes above.*

Add the following General Rules:

7.1) For every lost column descriptor *CD*, let *S1.TN* be the <table name> of the table whose descriptor includes the descriptor *CD* and let *CN* be the <column name> of *CD*. The following <alter table statement> is effectively executed without further Access Rule checking:

ALTER TABLE *S1.TN* DROP COLUMN *CN* CASCADE

7.2) For every lost domain descriptor *DO*, let *S1.DN* be the <domain name> of *DO*. The following <drop domain statement> is effectively executed without further Access Rule checking:

DROP DOMAIN *S1.DN* CASCADE

7.3) For every lost schema descriptor *SD*, the default character set of that schema is modified to include the name of the implementation-defined <character set specification> that would have been this schema's default character set had the <schema definition> not specified a <schema character set specification>.

9. *Rationale: The following changes describe the actions to take against impacted domains, columns, collations, and character sets.*

Add the following General Rules:

7.4) If the object identified by *O* is a collation, let *OCN* be the name of that collation.

7.5) For every impacted domain descriptor *DO*, *DO* is modified such that it does not include *OCN*.

7.6) For every impacted column descriptor *CD*, *CD* is modified such that it does not include *OCN*.

7.7) For every impacted collation descriptor *CN*, *CN* is modified such that it does not include *OCN*. If *CN* does not include any translation name, then *CN* is modified to indicate that it utilizes the DEFAULT collation for its character repertoire.

7.8) For every impacted character set descriptor *CSD*, *CSD* is modified such that it does not include *OCN*. If *CSD* does not include any translation name, then *CSD* is modified to indicate that it utilizes the DEFAULT collation for its character repertoire.

12.3 <procedure>

1. *Rationale: <get diagnostics statement> input parameters must also be established.*

Add the following General Rule 6) b) i.1):

i.1) The values of all input parameters to the <procedure> are established.

2. *Rationale: Remove the incorrect suggestion that the <set transaction statement> directly affects the constraint mode of any constraint.*

Replace General Rule 6) c) iii) 2) with the following:

2) Casc:

A) If a <set transaction statement> has been executed since the termination of the last SQL-transaction in the SQL-session, then the access mode and isolation level of the SQL-transaction are set as specified by the <set transaction statement>. If a <set constraints mode statement> *SCM* has been executed since the termination of the last SQL-transaction in the SQL-session, then the constraint modes of constraints specified in *SCM* are set as specified in *SCM*.

B) Otherwise, the access mode of that SQL-transaction is read-write and the isolation level of that SQL-transaction is SERIALIZABLE, and the constraint modes for all constraints in that SQL-transaction are set to their initial states.

3. *Rationale: Resolve inconsistency regarding "no data" and exception condition.*

Delete General Rules 8) and 9).

4. *Rationale: Clarify with respect to the language added to Subclause 4.18.1; to permit implementation-defined 01 warnings to be returned in SQLSTATE when no exceptions are specified by the Standard.*

Replace General Rule 11).a).ii) with the following:

ii) Either a completion condition is raised: *successful completion*, or a completion condition is raised: *warning*, or a completion condition is raised: *no data*.

12.4 Calls to a <procedure>

1. *Rationale: The newly revised Ada language standard (ISO/IEC-8652:1995, Information technology—Programming languages—Ada) contains support for decimal-encoded numeric data and variable length character strings. The revised interface allows newly written applications in the revised Ada language access to these features of SQL. However, support for programs written against the old Ada interface is preserved.*

Replace Syntax Rule 1) with:

1) If the subject <language clause> specifies ADA then:

   a) The SQL-implementation shall generate the source code of an Ada library unit package the name of which shall be identical to the <module name> of the <module> if <module name> is a valid Ada identifier and is otherwise implementation-defined. For each <procedure> of the <module> there shall appear within the package defined above a subprogram declaration declaring a procedure; the name of that procedure shall be <procedure name> if <procedure name> is a valid Ada identifier and is otherwise implementation-defined; the parameters in the Ada procedure declarations shall appear in the same order as the <parameter declarations> of the <procedure> and shall have the same <parameter names> as the <parameter declarations> if those names are valid Ada identifiers, otherwise the names are implementation-defined; the parameter modes and subtype marks used in the parameter specifications are constrained by the remaining paragraphs of this subclause.

   b) Any <data type> in a <parameter declaration> shall specify CHARACTER, CHARACTER VARYING, NATIONAL CHARACTER, NATIONAL CHARACTER VARYING, BIT, BIT VARYING, NUMERIC, SMALLINT, INTEGER, REAL, DOUBLE PRECISION.

   c) The types of parameter specifications within the Ada subprogram declarations shall be taken from the library unit package, `Interfaces.SQL` and its children: `Numerics` and `Varying` and optional children *`Adacsn`* and *`Adacsn`*`.Varying`.

      i) The declaration of the library unit package, `Interfaces.SQL`, shall conform to the following template:

```
package Interfaces.SQL is
  -- the declarations of CHAR and NCHAR
  -- may be subtype declarations
  type CHAR is see the rules
  type NCHAR is see the rules
  type BIT is array (POSITIVE range <>) of BOOLEAN;
  type SMALLINT is range bs..ts;
  type INT is range bi..ti;
  type REAL is digits dr;
  type DOUBLE_PRECISION is digits dd;
  type SQLCODE_TYPE is range bsc..tsc;
  subtype SQL_ERROR is SQLCODE_TYPE range SQLCODE_TYPE'FIRST .. -1;
  subtype NOT_FOUND is SQLCODE_TYPE range 100..100;
  subtype INDICATOR_TYPE is t;
  type SQLSTATE_TYPE is new CHAR (1 .. 5);
  package SQLSTATE_CODES is
    AMBIGUOUS_CURSOR_NAME_NO_SUBCLASS:
      constant SQLSTATE_TYPE :="3C000";
    CARDINALITY_VIOLATION_NO_SUBCLASS:
      constant SQLSTATE_TYPE :="21000";
    CONNECTION_EXCEPTION_NO_SUBCLASS:
      constant SQLSTATE_TYPE :="08000";
    CONNECTION_EXCEPTION_CONNECTION_DOES_NOT_EXIST:
```

```
    constant SQLSTATE_TYPE :="08003";
CONNECTION_EXCEPTION_CONNECTION_FAILURE:
    constant SQLSTATE_TYPE :="08006";
CONNECTION_EXCEPTION_CONNECTION_NAME_IN_USE:
    constant SQLSTATE_TYPE :="08002";
CONNECTION_EXCEPTION_SQLCLIENT_UNABLE_TO_ESTABLISH_SQLCONNECTION:
    constant SQLSTATE_TYPE :="08001";
CONNECTION_EXCEPTION_SQLSERVER_REJECTED_ESTABLISHMENT_OF_SQLCONNECTION:
    constant SQLSTATE_TYPE :="08004";
CONNECTION_EXCEPTION_TRANSACTION_RESOLUTION_UNKNOWN:
    constant SQLSTATE_TYPE :="08007";
DATA_EXCEPTION_NO_SUBCLASS:
    constant SQLSTATE_TYPE :="22000";
DATA_EXCEPTION_CHARACTER_NOT_IN_REPERTOIRE:
    constant SQLSTATE_TYPE :="22021";
DATA_EXCEPTION_DATETIME_FIELD_OVERFLOW:
    constant SQLSTATE_TYPE :="22008";
DATA_EXCEPTION_DIVISION_BY_ZERO:
    constant SQLSTATE_TYPE :="22012";
DATA_EXCEPTION_ERROR_IN_ASSIGNMENT:
    constant SQLSTATE_TYPE :="22005";
DATA_EXCEPTION_INDICATOR_OVERFLOW:
    constant SQLSTATE_TYPE :="22022";
DATA_EXCEPTION_INTERVAL_FIELD_OVERFLOW:
    constant SQLSTATE_TYPE :="22015";
DATA_EXCEPTION_INVALID_CHARACTER_VALUE_FOR_CAST:
    constant SQLSTATE_TYPE :="22018";
DATA_EXCEPTION_INVALID_DATETIME_FORMAT:
    constant SQLSTATE_TYPE :="22007";
DATA_EXCEPTION_INVALID_ESCAPE_CHARACTER:
    constant SQLSTATE_TYPE :="22019";
DATA_EXCEPTION_INVALID_ESCAPE_SEQUENCE:
    constant SQLSTATE_TYPE :="22025";
DATA_EXCEPTION_INVALID_INTERVAL_FORMAT:
    constant SQLSTATE_TYPE := "22006";
DATA_EXCEPTION_INVALID_PARAMETER_VALUE:
    constant SQLSTATE_TYPE :="22023";
DATA_EXCEPTION_INVALID_TIME_ZONE_DISPLACEMENT_VALUE:
    constant SQLSTATE_TYPE :="22009";
DATA_EXCEPTION_NULL_VALUE_NO_INDICATOR_PARAMETER:
    constant SQLSTATE_TYPE :="22002";
DATA_EXCEPTION_NUMERIC_VALUE_OUT_OF_RANGE:
    constant SQLSTATE_TYPE :="22003";
DATA_EXCEPTION_STRING_DATA_LENGTH_MISMATCH:
    constant SQLSTATE_TYPE :="22026";
DATA_EXCEPTION_STRING_DATA_RIGHT_TRUNCATION:
    constant SQLSTATE_TYPE :="22001";
DATA_EXCEPTION_SUBSTRING_ERROR:
    constant SQLSTATE_TYPE :="22011";
DATA_EXCEPTION_TRIM_ERROR:
    constant SQLSTATE_TYPE :="22027";
DATA_EXCEPTION_UNTERMINATED_C_STRING:
    constant SQLSTATE_TYPE :="22024";
DEPENDENT_PRIVILEGE_DESCRIPTORS_STILL_EXIST_NO_SUBCLASS:
    constant SQLSTATE_TYPE :="2B000";
DYNAMIC_SQL_ERROR_NO_SUBCLASS:
    constant SQLSTATE_TYPE :="07000";
DYNAMIC_SQL_ERROR_CURSOR_SPECIFICATION_CANNOT_BE_EXECUTED:
    constant SQLSTATE_TYPE :="07003";
DYNAMIC_SQL_ERROR_INVALID_DESCRIPTOR_COUNT:
    constant SQLSTATE_TYPE :="07008";
DYNAMIC_SQL_ERROR_INVALID_DESCRIPTOR_INDEX:
    constant SQLSTATE_TYPE :="07009";
DYNAMIC_SQL_ERROR_PREPARED_STATEMENT_NOT_A_CURSOR_SPECIFICATION:
    constant SQLSTATE_TYPE :="07005";
```

```
DYNAMIC_SQL_ERROR_RESTRICTED_DATA_TYPE_ATTRIBUTE_VIOLATION:
  constant SQLSTATE_TYPE :="07006";
DYNAMIC_SQL_ERROR_USING_CLAUSE_DOES_NOT_MATCH_DYNAMIC_PARAMETER_SPEC:
  constant SQLSTATE_TYPE :="07001";
DYNAMIC_SQL_ERROR_USING_CLAUSE_DOES_NOT_MATCH_TARGET_SPEC:
  constant SQLSTATE_TYPE :="07002";
DYNAMIC_SQL_ERROR_USING_CLAUSE_REQUIRED_FOR_DYNAMIC_PARAMETERS:
  constant SQLSTATE_TYPE :="07004";
DYNAMIC_SQL_ERROR_USING_CLAUSE_REQUIRED_FOR_RESULT_FIELDS:
  constant SQLSTATE_TYPE :="07007";
FEATURE_NOT_SUPPORTED_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="0A000";
FEATURE_NOT_SUPPORTED_MULTIPLE_ENVIRONMENT_TRANSACTIONS:
  constant SQLSTATE_TYPE :="0A001";
INTEGRITY_CONSTRAINT_VIOLATION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="23000";
INVALID_AUTHORIZATION_SPECIFICATION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="28000";
INVALID_CATALOG_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="3D000";
INVALID_CHARACTER_SET_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="2C000";
INVALID_CONDITION_NUMBER_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="35000";
INVALID_CONNECTION_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="2E000";
INVALID_CURSOR_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="34000";
INVALID_CURSOR_STATE_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="24000";
INVALID_SCHEMA_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="3F000";
INVALID_SQL_DESCRIPTOR_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="33000";
INVALID_SQL_STATEMENT_NAME_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="26000";
INVALID_TRANSACTION_STATE_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="25000";
INVALID_TRANSACTION_TERMINATION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="2D000";
NO_DATA_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="02000";
REMOTE_DATABASE_ACCESS_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="HZ000";
REMOTE_DATABASE_ACCESS_REMOTE_DATABASE_ACCESS_ACCESS_CONTROL_VIOLATION:
  constant SQLSTATE_TYPE :="HZ010";
REMOTE_DATABASE_ACCESS_REMOTE_DATABASE_ACCESS_BAD_REPETITION_COUNT:
  constant SQLSTATE_TYPE :="HZ020";
REMOTE_DATABASE_ACCESS_REMOTE_DATABASE_ACCESS_COMMAND_HANDLE_UNKNOWN:
  constant SQLSTATE_TYPE :="HZ030";
REMOTE_DATABASE_ACCESS_REMOTE_DATABASE_ACCESS_CONTROL_AUTHENTICATION_FAILURE:
  constant SQLSTATE_TYPE :="HZ040";
REMOTE_DATABASE_ACCESS_CONTROL_SERVICES_NOT_ALLOWED:
  constant SQLSTATE_TYPE :="HZ230";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_HANDLE_NOT_SPECIFIED:
  constant SQLSTATE_TYPE :="HZ050";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_HANDLE_UNKNOWN:
  constant SQLSTATE_TYPE :="HZ060";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_NAME_NOT_SPECIFIED:
  constant SQLSTATE_TYPE :="HZ070";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_NOT_AVAILABLE_PREMANENT:
  constant SQLSTATE_TYPE :="HZ080";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_NOT_AVAILABLE_TRANSIENT:
  constant SQLSTATE_TYPE :="HZ081";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_ALREADY_OPEN:
```

```
    constant SQLSTATE_TYPE :="HZ090";
REMOTE_DATABASE_ACCESS_DATA_RESOURCE_UNKNOWN:
    constant SQLSTATE_TYPE :="HZ100";
REMOTE_DATABASE_ACCESS_DIALOGUE_ID_UNKNOWN:
    constant SQLSTATE_TYPE :="HZ110";
REMOTE_DATABASE_ACCESS_DUPLICATE_COMMAND_HANDLE:
    constant SQLSTATE_TYPE :="HZ120";
REMOTE_DATABASE_ACCESS_DUPLICATE_DATA_RESOURCE_HANDLE:
    constant SQLSTATE_TYPE :="HZ130";
REMOTE_DATABASE_ACCESS_DUPLICATE_DIALOGUE_ID:
    constant SQLSTATE_TYPE :="HZ140";
REMOTE_DATABASE_ACCESS_DUPLICATE_OPERATION_ID:
    constant SQLSTATE_TYPE :="HZ150";
REMOTE_DATABASE_ACCESS_INVALID_SEQUENCE:
    constant SQLSTATE_TYPE :="HZ160";
REMOTE_DATABASE_ACCESS_DIALOGUE_ALREADY_ACTIVE:
    constant SQLSTATE_TYPE :="HZ161";
REMOTE_DATABASE_ACCESS_DIALOGUE_INITIALIZING:
    constant SQLSTATE_TYPE :="HZ162";
REMOTE_DATABASE_ACCESS_DIALOGUE_NOT_ACTIVE:
    constant SQLSTATE_TYPE :="HZ163";
REMOTE_DATABASE_ACCESS_DIALOGUE_TERMINATING:
    constant SQLSTATE_TYPE :="HZ164";
REMOTE_DATABASE_ACCESS_TRANSACTION_NOT_OPEN:
    constant SQLSTATE_TYPE :="HZ165";
REMOTE_DATABASE_ACCESS_TRANSACTION_OPEN:
    constant SQLSTATE_TYPE :="HZ166";
REMOTE_DATABASE_ACCESS_TRANSACTION_TERMINATING:
    constant SQLSTATE_TYPE :="HZ167";
REMOTE_DATABASE_ACCESS_NO_DATA_RESOURCE_AVAILABLE:
    constant SQLSTATE_TYPE :="HZ170";
REMOTE_DATABASE_ACCESS_OPERATION_ABORTED_PERMANENT:
    constant SQLSTATE_TYPE :="HZ180";
REMOTE_DATABASE_ACCESS_OPERATION_ABORTED_TRANSIENT:
    constant SQLSTATE_TYPE :="HZ181";
REMOTE_DATABASE_ACCESS_OPERATION_CANCELLED:
    constant SQLSTATE_TYPE :="HZ190";
REMOTE_DATABASE_ACCESS_SERVICE_NOT_NEGOTIATED:
    constant SQLSTATE_TYPE :="HZ200";
REMOTE_DATABASE_ACCESS_TRANSACTION_ROLLED_BACK:
    constant SQLSTATE_TYPE :="HZ210";
REMOTE_DATABASE_ACCESS_USER_AUTHENTICATION_FAILURE:
    constant SQLSTATE_TYPE :="HZ220";
REMOTE_DATABASE_ACCESS_HOST_IDENTIFIER_ERROR:
    constant SQLSTATE_TYPE :="HZ300";
REMOTE_DATABASE_ACCESS_INVALID_SQL_CONFORMANCE_LEVEL:
    constant SQLSTATE_TYPE :="HZ310";
REMOTE_DATABASE_ACCESS_RDA_TRANSACTION_NOT_OPEN:
    constant SQLSTATE_TYPE :="HZ320";
REMOTE_DATABASE_ACCESS_RDA_TRANSACTION_OPEN:
    constant SQLSTATE_TYPE :="HZ325";
REMOTE_DATABASE_ACCESS_SQL_ACCESS_CONTROL_VIOLATION:
    constant SQLSTATE_TYPE :="HZ330";
REMOTE_DATABASE_ACCESS_SQL_DATABASE_RESOURCE_ALREADY_OPEN:
    constant SQLSTATE_TYPE :="HZ340";
REMOTE_DATABASE_ACCESS_SQL_DBL_ARGUMENT_COUNT_MISMATCH:
    constant SQLSTATE_TYPE :="HZ350";
REMOTE_DATABASE_ACCESS_SQL_DBL_ARGUMENT_TYPE_MISMATCH:
    constant SQLSTATE_TYPE :="HZ360";
REMOTE_DATABASE_ACCESS_SQL_DBL_NO_CHAR_SET:
    constant SQLSTATE_TYPE :="HZ365";
REMOTE_DATABASE_ACCESS_SQL_DBL_TRANSACTION_STATEMENT_NOT_ALLOWED:
    constant SQLSTATE_TYPE :="HZ370";
REMOTE_DATABASE_ACCESS_SQL_USAGE_MODE_VIOLATION:
    constant SQLSTATE_TYPE :="HZ380";
```

```
REMOTE_DATABASE_ACCESS_ABORT_FAILURE_SERVICE_PROVIDER:
  constant SQLSTATE_TYPE :="HZ410";
REMOTE_DATABASE_ACCESS_ABORT_FAILURE_SERVICE_USER:
  constant SQLSTATE_TYPE :="HZ411";
REMOTE_DATABASE_ACCESS_ASSOCIATION_FAILURE_PERMANENT:
  constant SQLSTATE_TYPE :="HZ420";
REMOTE_DATABASE_ACCESS_ASSOCIATION_FAILURE_TRANSIENT:
  constant SQLSTATE_TYPE :="HZ421";
REMOTE_DATABASE_ACCESS_RELEASE_FAILURE:
  constant SQLSTATE_TYPE :="HZ430";
REMOTE_DATABASE_ACCESS_BEGIN_DIALOGUE_REJECTED_PROVIDER:
  constant SQLSTATE_TYPE :="HZ450";
REMOTE_DATABASE_ACCESS_BEGIN_DIALOGUE_REJECTED_USER:
  constant SQLSTATE_TYPE :="HZ451";
REMOTE_DATABASE_ACCESS_HEURISTIC_HAZARD:
  constant SQLSTATE_TYPE :="HZ460";
REMOTE_DATABASE_ACCESS_HEURISTIC_MIX:
  constant SQLSTATE_TYPE :="HZ461";
REMOTE_DATABASE_ACCESS_PABORT_ROLLBACK_FALSE:
  constant SQLSTATE_TYPE :="HZ470";
REMOTE_DATABASE_ACCESS_PABORT_ROLLBACK_TRUE:
  constant SQLSTATE_TYPE :="HZ471";
REMOTE_DATABASE_ACCESS_ROLLBACK:
  constant SQLSTATE_TYPE :="HZ480";
REMOTE_DATABASE_ACCESS_UABORT_ROLLBACK_FALSE:
  constant SQLSTATE_TYPE :="HZ490";
REMOTE_DATABASE_ACCESS_UABORT_ROLLBACK_TRUE:
  constant SQLSTATE_TYPE :="HZ491";
REMOTE_DATABASE_ACCESS_UERROR:
  constant SQLSTATE_TYPE :="HZ4A0";
SUCCESSFUL_COMPLETION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="00000";
SYNTAX_ERROR_OR_ACCESS_RULE_VIOLATION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="42000";
TRANSACTION_ROLLBACK_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="40000";
TRANSACTION_ROLLBACK_INTEGRITY_CONSTRAINT_VIOLATION:
  constant SQLSTATE_TYPE :="40002";
TRANSACTION_ROLLBACK_SERIALIZATION_FAILURE:
  constant SQLSTATE_TYPE :="40001";
TRANSACTION_ROLLBACK_STATEMENT_COMPLETION_UNKNOWN:
  constant SQLSTATE_TYPE :="40003";
TRIGGERED_DATA_CHANGE_VIOLATION_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="27000";
WARNING_NO_SUBCLASS:
  constant SQLSTATE_TYPE :="01000";
WARNING_CURSOR_OPERATION_CONFLICT:
  constant SQLSTATE_TYPE :="01001";
WARNING_DEFAULT_OPTION_TOO_LONG_FOR_INFORMATION_SCHEMA:
  constant SQLSTATE_TYPE := "0100B";
WARNING_DISCONNECT_ERROR:
  constant SQLSTATE_TYPE :="01002";
WARNING_IMPLICIT_ZERO_BIT_PADDING:
  constant SQLSTATE_TYPE :="01008";
WARNING_INSUFFICIENT_ITEM_DESCRIPTOR_AREAS:
  constant SQLSTATE_TYPE :="01005";
WARNING_NULL_VALUE_ELIMINATED_IN_SET_FUNCTION:
  constant SQLSTATE_TYPE :="01003";
WARNING_PRIVILEGE_NOT_GRANTED:
  constant SQLSTATE_TYPE :="01007";
WARNING_PRIVILEGE_NOT_REVOKED:
  constant SQLSTATE_TYPE :="01006";
WARNING_QUERY_EXPRESSION_TOO_LONG_FOR_INFORMATION_SCHEMA:
  constant SQLSTATE_TYPE :="0100A";
WARNING_SEARCH_CONDITION_TOO_LONG_FOR_INFORMATION_SCHEMA:
```

```
      constant SQLSTATE_TYPE :="01009";
    WARNING_STRING_DATA_RIGHT_TRUNCATION_WARNING:
      constant SQLSTATE_TYPE :="01004";
    WITH_CHECK_OPTION_VIOLATION_NO_SUBCLASS:
      constant QLSTATE_TYPE :="44000";
  end SQLSTATE_CODES;
end Interfaces.SQL;
```

where *bs*, *ts*, *bi*, *ti*, *dr*, *dd*, *bsc* and *tsc* are implementation-defined integral values; *t* is INT or SMALLINT corresponding to the implementation-defined <exact numeric type> of indicator parameters.

ii) The library unit package `Interfaces.SQL.Numerics` shall contain a sequence of decimal fixed point type declarations of the following form.

```
type Scale_s is delta 10.0 ** -s digits max_p;
```

where *s* is an integer ranging from 0 to an implementation-defined maximum value and *max_p* is an implementation-defined integer maximum precision.

iii) The library unit package `Interfaces.SQL.Varying` shall contain type or subtype declarations with the defining identifiers CHAR, NCHAR and BIT.

iv) Let *SQLcsn* be a <character set name> and let *Adacsn* be the result of replacing <period>'s in *SQLcsn* with <underscore>. If *Adacsn* is a valid Ada identifier, then the library unit packages `Interfaces.SQL.`*Adacsn* and `Interfaces.SQL.`*Adacsn*`.Varying` shall contain a type or subtype declaration with defining identifier CHAR. If *Adacsn* is not a valid Ada identifier, the names of these packages shall be implementation-defined.

v) `Interfaces.SQL` and its children may contain context clauses and representation items as needed. These packages may also contain declarations of Ada character types as needed to support the declarations of the types CHAR and NCHAR.

Note: If the implementation-defined character set specification used by default with a CHARACTER data type is Latin1, then the declaration

```
subtype CHAR is String;
```

within `Interfaces.SQL` and the declaration

```
subtype CHAR is Ada.Strings.Unbounded.Unbounded_String;
```

within `Interfaces.SQL.Varying` (assuming the appropriate context clause), conform to the requirements of this paragraph of this subclause. If the character set underlying NATIONAL CHARACTER is supported by an Ada package specification *Host_Char_Pkg* that declares a type *String_Type* that stores strings over the given character set, and furthermore the package specification *Host_Char_Pkg_Varying* (not necessary distinct from *Host_Char_Pkg*) declares a type *String_Type_Varying* that reproduces the functionality of Ada.Strings.Unbounded.Unbounded_String over the national character type (rather than Latin1), then the declaration

```
subtype NCHAR is Host_Char_Pkg.String_Type;
```

within Interfaces.SQL and the declaration

```
subtype NCHAR is Host_Char_Pkg_Varying.String_Type_Varying;
```

within `Interfaces.SQL.Varying` conform to the requirements of this paragraph. Similar comments apply to other character sets and the packages `Interfaces.SQL.`*Adacsn* and `Interfaces.SQL.`*Adacsn*`.Varying`.

vi) The library unit package `Interfaces.SQL` shall contain declarations of the following form:

```
package CHARACTER_SET renames Interfaces.SQL.Adacsn;
```

```
subtype CHARACTER_TYPE is CHARACTER_SET.cst;
```

where *cst* is a data type capable of storing a single character from the default character set. The package `Interfaces.SQL.Adacsn` shall contain the necessary declaration for *cst*.

Note: If the default character set is Latin1, then a declaration of the form:

```
package CHARACTER_SET is
  subtype cst is Character;
end CHARACTER_SET;
```

may be substituted for the renaming declaration of CHARACTER_SET.

d) The base type of the SQLCODE parameter shall be `Interfaces.SQL.SQLCODE_TYPE`. The base type of the SQLSTATE parameter shall be `Interfaces.SQL.SQLSTATE_TYPE`.

Note: SQLSTATE is the preferred status parameter. The SQLCODE status parameter is a deprecated feature that is supported for compatibility with earlier versions of this International Standard. See Annex D, "Deprecated Features".

e) The Ada parameter mode of the SQLCODE parameter is **out**. The Ada parameter mode of the SQLSTATE parameter is **out**.

f) If the *i*-th <parameter declaration> specifies a <data type> that is

i) CHARACTER(*L*) for some *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.CHAR`;

ii) CHARACTER VARYING(*L*) for some *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.VARYING.CHAR`;

iii) NATIONAL CHARACTER(*L*) for some *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.NCHAR`;

iv) NATIONAL CHARACTER VARYING(*L*) for some *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.VARYING.NCHAR`;

v) CHARACTER(*L*) CHARACTER SET *csn* for some *L* and some character set name, *csn*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.Adacsn.CHAR`;

vi) CHARACTER VARYING(*L*) CHARACTER SET *csn* for some *L* and some character set name, *csn*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.Adacsn.VARYING.CHAR`

If *P* is an actual parameter associated with the *i*-th parameter in a call to the encompassing procedure, then *P* shall be sufficient to hold a character string of length *L* in the appropriate character set.

Note: If a character set uses fixed length encodings then the definition of the subtype CHAR for fixed length strings may be an array type whose element type is an Ada character type. If that character type is defined so as to use the number of bits per character used by the SQL encoding, then the restriction on *P* is precisely P'LENGTH = *L*. For variable length strings using fixed length encodings, if the definition of CHAR in the appropriate VARYING package is based on the type Ada.Strings.Unbounded.Unbounded_String, there is no restriction on *P*. Otherwise, a precise statement of the restriction on *P* is implementation-defined.

g) If the *i*-th <parameter declaration> specifies a <data type> that is

i) BIT(*L*) for some length *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.BIT`. If *P* is an actual parameter associated with the *i*-th parameter in a call to the encompassing procedure, then P'LENGTH shall be equal to *L*.

ii) BIT VARYING(*L*) for some length *L*, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.Varying.BIT`. If *P* is an actual parameter associatcd with the *i*-th parameter in a call to the encompassing procedure, then *P* shall bc sufficient to hold a bit string of length *L*.

h) If the *i*-th <parameter declaration> specifies a <data type> that is NUMERIC(*P*,*S*) for some <precision> and <scale> *P* and *S*, then the Ada library unit package generated for the encompassing module shall contain a declaration equivalent to

```
subtype Numeric_p_s is Interfaces.SQL.Numerics.Scale_s
    digits p;
```

The subtype mark in the *i*-th parameter specification shall specify this subtype.

i) If the *i*-th <parameter declaration> specifies a <data type> that is SMALLINT, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.SMALLINT`.

j) If the *i*-th <parameter declaration> specifies a <data type> that is INTEGER, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.INT`.

k) If the *i*-th <parameter declaration> specifies a <data type> that is REAL, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.REAL`.

l) If the *i*-th <parameter declaration> specifies a <data type> that is DOUBLE_PRECISION, then the subtype mark in the *i*-th parameter declaration shall specify `Interfaces.SQL.DOUBLE_PRECISION`.

m) For every parameter,

Case:

i) If the parameter is an input parameter but not an output parameter, then the Ada parameter mode is **in**.

ii) If the parameter is an output parameter but not an input parameter, then the Ada parameter mode is **out**.

iii) If the parameter is both an input parameter and an output parameter, then the Ada parameter mode is **in out**.

iv) Otherwise, the Ada parameter mode is **in**, **out**, or **in out**.

n) The following Ada library unit renaming declaration exists:

```
with Interfaces.SQL;
package SQL_Standard renames Interfaces.SQL.
```

2. *Rationale: When CHARACTER or CHARACTER VARYING data is exchanged between host programs written in the C language and an SQL database system, the C convention of terminating strings with a "null character" is used on the C side of the interface. The binding rules incorrectly assume that the "null character" is always equivalent to a single "char" in C (i.e., a single byte). However, so-called wide character sets like Unicode sometimes encode characters with one of the two bytes equal to 0; this requires that the C convention be modified so that a number of bytes equivalent to the size of the "widest" character all be 0. That, in turn, requires that the SQL/C binding provide sufficient storage for those additional bytes.*

Replace Syntax Rule 2) c) with the following:

c) If the i-th <parameter declaration> specifies a <data type> *PDT* that is CHARACTER(*L*) or CHARACTER VARYING(*L*) for some <length> *L*, then the type of the *i*-th parameter *P* shall be C char with length *k* greater than the maximum possible length in octets of *PDT*, where *k* is the size in octets of the largest character in the character set of *PDT*.

3. *Rationale: Editorial.*

In Syntax Rule 2) d), replace "(*L*/(*B*+1))" with "(*L*/*B*) plus one".

4. *Rationale: Fix an incorrect PL/I specification.*

In Syntax Rule 7) f), delete the word "REAL".

In Syntax Rule 7) g), delete the word "REAL".

5. *Rationale: The following rule defines a number of the symbols that are used later in the Subclause. Its deletion (as 12.3 General Rule 8)) in Technical Corrigendum 1 without replacement was in error.*

Insert the following General Rule:

0.1) When a <procedure> is called by an SQL-agent, let PD_i be the <parameter declaration> of the *i*-th parameter and let DT_i and PN_i be the <data type> and the <parameter name> specified in PD_i, respectively. Let PI_i be the *i*-th parameter in the procedure call.

6. *Rationale: A <select statement: single row> or a <fetch statement> may return a null value. General Rule 1) of 9.1 "Retrieval assignment" assigns a value to an indicator parameter, but not its corresponding data parameter. Ada requires that all output parameters be assigned values by a subprogram.*

Replace General Rule 1) with the following:

1) If the subject <language clause> specifies ADA, then:

a) Where P_i is used as an input parameter whose value is evaluated, a reference to PN_i in a <general value specification> has the value PI_i.

b) Where P_i is used as an output parameter, a reference to PN_i that assigns a value SV_i to PN_i implicitly assigns the value SV_i to PI_i.

c) If P_i is used as an output parameter and no value has been assigned to PL_i, then an implementation-dependent value is assigned PI_i.

7. *Rationale: When CHARACTER or CHARACTER VARYING data is exchanged between host programs written in the C language and an SQL database system, the C convention of terminating strings with a "null character" is used on the C side of the interface. The binding rules incorrectly assume that the "null character" is always equivalent to a single "char" in C (i.e., a single byte). However, so-called wide character sets like Unicode sometimes encode characters with one of the two bytes equal to 0; this requires that the C convention be modified so that a number of bytes equivalent to the size of the "widest" character all be 0. That, in turn, requires that the SQL/C binding provide sufficient storage for those additional bytes.*

Replace the first sentence of General Rule 2) b) ii) with the following:

ii) Let CL_i be k greater than the maximum possible length in octets of PN_i, where k is the size in octets of the largest character in the character set of DT_i.

Add the following note at the end of General Rule 2) b):

Note: In the preceding Rule, the phrase "implementation-defined null character that terminates a C character string" implies one or more octets all of whose bits are zero and whose number is equal to the number of octets in the largest character of the character set of DT_i.

13.1 <declare cursor>

1. *Rationale: Editorial.*

Replace Syntax Rule 8) with the following:

8) The *simply underlying table* of the <cursor specification> is *T*.

2. *Rationale: There is a contradiction between Syntax Rules 13) and 14) of Subclause 13.1 and Syntax Rule 8) and Access Rule 2) of Subclause 13.9.*

Replace Syntax Rule 13) with the following:

13) If an <updatability clause> of FOR UPDATE without a <column name list> is specified or implicit, then a <column name list> that includes the <column name> of every column of the simply underlying table of the simply underlying table of *T* is implicit.

Replace Syntax Rule 14) with the following:

14) If an <updatability clause> of FOR UPDATE with a <column name list> is specified, then each <column name> in the <column name list> shall be the <column name> of a column of the simply underlying table of the simply underlying table of *T*.

Delete Syntax Rule 9) and insert it as General Rule 0.1).

13.2 <open statement>

1. *Rationale: CR was used ambiguously to reference the syntactic construct <declare cursor> and the cursor itself. This change disambiguates the references. The change also accommodates the use of the rules of this Subclause for dynamic cursors.*

Replace Syntax Rule 1) with the following:

1) The containing <module> shall contain a <declare cursor> *DC* whose <cursor name> is the same as the <cursor name> in the <open statement>. Let *CR* be the cursor specified by *DC*. *CR* is *associated* with the <cursor specification> contained in *DC*.

Replace General Rule 2) with the following:

2) Let *S* be the <cursor specification> associated with *CR*.

13.3 <fetch statement>

1. *Rationale: CR was used ambiguously to reference the syntactic construct <declare cursor> and the cursor itself. This change disambiguates the references.*

Replace Syntax Rule 2) with the following:

2) The containing <module> shall contain a <declare cursor> *DC* whose <cursor name> is the same as the <cursor name> in the <fetch statement>. Let *T* be the table defined by the <cursor specification> of *DC*. Let *CR* be the cursor specified by *DC*.

Replace Syntax Rule 3) with the following:

3) If the implicit or explicit <fetch orientation> is not NEXT, then the <declare cursor> *DC* shall specify SCROLL.

2. *Rationale: Resolve inconsistency regarding "no data" and exception condition.*

Replace General Rule 9) with the following:

9) If an exception condition occurs during the assignment of a value to a target, then the values of all targets are implementation-dependent and *CR* remains positioned on the current row.

13.4 <close statement>

1. *Rationale: CR was used ambiguously to reference the syntactic construct <declare cursor> and the cursor itself. This change disambiguates the references.*

Replace Syntax Rule 1) with the following:

1) The containing <module> shall contain a <declare cursor> *DC* whose <cursor name> is the same as the <cursor name> in the <close statement>. Let *CR* be the cursor specified by *DC*.

Replace General Rule 2) with the following:

2) Cursor *CR* is placed in the closed state and the copy of the <cursor specification> of *DC* is destroyed.

13.5 <select statement: single row>

1. Rationale: Resolve inconsistency regarding "no data" and exception condition.

Delete General Rule 5)

Add the following General Rule:

8.1) If an exception condition is raised during the assignment of a value to a target, then the values of all targets are implementation-dependent.

13.6 <delete statement: positioned>

1. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

3.1) The schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*.

13.7 <delete statement: searched>

1. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

2.1) The schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*.

13.8 <insert statement>

1. Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.

Add the following Syntax Rule:

6.1) The schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*.

2. Rationale: Editorial.

In Access Rule 1) a), replace "<privileges>" with "privileges".

In Leveling Rule 2) a), replace "<insert statement>" with "<insert columns and source>".

3. Rationale: Editorial.

In Leveling Rule 2) a), replace "<value specification>" with "<value specification> or a <null specification>".

13.9 <update statement: positioned>

1. *Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.*

Add the following Syntax Rule:

10.1) The schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*.

2. *Rationale: The following change is needed for the same reason that the changes to Subclause 11.5, "<default clause>" were necessary.*

Replace General Rule 5) with the following:

5) The value associated with DEFAULT is the default value for the <object column> in the containing <set clause>, as indicated in the General Rules of Subclause 11.5, "<default clause>".

3. *Rationale: There is a contradiction between Syntax Rules 13) and 14) of Subclause 13.1 and Syntax Rule 8) and Access Rule 2) of Subclause 13.9.*

Delete Access Rule 2) and insert it as Syntax Rule 7.1).

13.10 <update statement: searched>

1. *Rationale: Clarify that references to non-existing objects is only allowed within the same <schema definition>.*

Add the following Syntax Rule:

5.1) The schema identified by the explicit or implicit qualifier of the <table name> shall include the descriptor of *T*.

14.1 <set transaction statement>

1. *Rationale: Clarify the scope of the <set transaction statement>.*

Add the following sentence to the end of the paragraph under Function:

Note: This statement has no effect on any SQL-transaction subsequent to the next SQL-transaction.

14.2 <set constraints mode statement>

1. *Rationale: Clarify the scope of the <set transaction statement>.*

Add the following sentence to the end of the paragraph under Function:

This statement has no effect on any SQL-transaction subsequent to this SQL-transaction.

14.3 <commit statement>

1. Rationale: Clarification.

In General Rule 4), replace "were executed" with "were executed for each active SQL-connection".

15.1 <connect statement>

1. Rationale: Editorial.

In General Rule 9) a), replace "SQL-server" with "SQL-session".

2. Rationale: Editorial.

Replace General Rule 2) with the following:

2) If <user name> is specified, then let *S* be the character string that is the value of <user name> and let *V* be the character string that is the value of

TRIM (BOTH ' ' FROM *S*)

15.2 <set connection statement>

1. Rationale: Editorial.

In General Rule 4), replace "SQL-server" with "SQL-session".

15.3 <disconnect statement>

1. Rationale: Editorial.

In General Rule 3), replace "SQL-connection, if any" with "SQL-connection".

16.5 <set local time zone statement>

1. Rationale: Editorial.

In Leveling Rule 1), replace "SQL;" with "SQL:".

In Leveling Rule 2), replace "SQL;" with "SQL, in addition to any Intermediate SQL restrictions:".

17.1 Description of SQL item descriptor areas

1. Rationale: Editorial.

In Syntax Rule 3) e), change "date" to "data".

2. *Rationale: Syntax Rule 3) applies only to data types that are not implementation-defined.*

Add the following to Syntax Rule 2):

l) TYPE indicates an implementation-defined data type and *T* satisfies the implementation-defined rules for matching that data type.

Add the following to Syntax Rule 3):

i) TYPE indicates an implementation-defined data type.

17.2 <allocate descriptor statement>

1. *Rationale: Editorial.*

In General Rule 2), replace "<scope clause>" with "<scope option>".

17.3 <deallocate descriptor statement>

1. *Rationale: Editorial.*

In General Rule 1), replace "<scope clause>" with "<scope option>".

2. *Rationale: Editorial.*

In Leveling Rule 2) a), replace "shall contain no" with "shall not contain any".

17.4 <get descriptor statement>

1. *Rationale: Clarification of wording.*

In General Rule 7) b) replace "datetime data type, interval qualifier" with "datetime data type or interval qualifier as appropriate, interval leading field precision" and replace "character set of" with "character set, respectively, of".

2. *Rationale: The behavior of <get descriptor statement> should be consistent with the behavior of <using clause>.*

Replace the last sentence of General Rule 7) c) with the following:

For a <dynamic parameter specification>, the value of UNNAMED is 1 and the value of NAME is implementation-dependent.

17.5 <set descriptor statement>

1. *Rationale: Disallow the use of literals to specify a value for the DATA field of the SQL descriptor due to the serious problems that arise if one attempts to enforce the existing match rules against literals.*

Add the following Syntax Rule:

3.1) If the <descriptor item name> is DATA, then <simple value specification 2> shall not be a <literal>.

17.6 <prepare statement>

1. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

In General Rule 4), replace "*syntax error or access rule violation in dynamic SQL statement*" with "*syntax error or access rule violation*".

2. *Rationale: Clarify that the operands referred to collectively as E1 may have different syntactic forms.*

Replace General Rule 4) c) with the following:

c) *P* contains *E1* and *E2* as operands of a single dyadic operator.

3. *Rationale: Clarify the use of <dynamic parameter specification> in COALESCE.*

Replace General Rule 4) f) with the following:

f) *P* contains a <null predicate> whose <row value constructor> simply contains *E1* unless *E1* is directly contained in a <value expression> that is an operand of COALESCE.

4. *Rationale: Editorial.*

In General Rule 4) g), replace "where *E1* is the" with "where the" and replace "of the <overlaps predicate>" with "of the <overlaps predicate> is *E1*".

5. *Rationale: Clarify the use of a <dynamic parameter specification> in COALESCE and in a <case expression>.*

Replace General Rule 4) h) with the following:

h) *P* contains a <simple case> where <case operand> and each <when operand> meets the criteria for *E1*, or *P* contains a <case abbreviation> where each operand meets the criteria for *E1*.

6. *Rationale: Clarify the use of a <dynamic parameter specification> in a <between predicate>.*

Replace General Rule 4) i) with the following:

i) *P* contains a <comparison predicate> or <between predicate> where, for some *i*, the *i*-th <value expression> in all of the immediately contained <row value constructor>s meets the criteria for *E1*.

7. *Rationale: Clarify that the operands referred to collectively as E1 may have different syntactic forms.*

Replace General Rule 4) j) with the following:

j) *P* contains a <table value constructor> in which the i-th <value expression> in each <row value constructor> meets the criteria for *E1* and either:

   i) *P* is not an <insert statement>, or

   ii) *P* is an insert statement> and the <table value expression> is not the <query expression> simply contained in the <insert statement>.

8. *Rationale: Clarify the use of <dynamic parameter specification>s in <in value list>.*

Replace General Rule 4) k) with the following:

k) *P* contains an <in predicate> with an <in value list> in which the <row value constructor> and each <value expression> of the <in value list> meet the criteria for *E1*.

9. *Rationale: Define the use of <dynamic parameter specification>s in <result expression>s of <case specification>s.*

Add the following to General Rule 4):

r) *P* contains a <case specification> where at least one <result> is *E1* and each other <result> is either NULL or meets the criteria for *E1*.

10. *Rationale: Clarify the use of <dynamic parameter specification> in COALESCE.*

Replace General Rule 5) j) with the following:

j) If one or more operands of COALESCE are *E1*, then the data type of *E1* is the data type determined by applying Subclause 9.3, "Set operation result data types", to the operands that do not meet the criteria for *E1*.

11. *Rationale: Clarify the use of a <dynamic parameter specification> in a <case expression>.*

Replace General Rule 5) k) with the following:

k) If one or more of the operands of the <case operand> and the <when operand>s in a <case specification> are *E1*, then the data type of *E1* is the data type determined by applying Subclause 9.3, "Set operation result data types", to the <case operand> and all the <when operand>s that do not meet the criteria for *E1*.

12. *Rationale: Clarify the use of a <dynamic parameter specification> in a <between predicate>.*

Replace General Rule 5) m) with the following:

m) In the first and second operands of a <comparison predicate>, if the *i*-th value of one operand is *E1*, then the data type of *E1* is the data type of the *i*-th value of the other operand.

13. *Rationale: Define the type of <dynamic parameter specification>s in <table value constructor>s.*

Replace General Rule 5) o) with the following:

o) In a <table value constructor> that is not a <query expression> simply contained in an <insert statement>, in which the i-th <value expression> of some <row value constructor> is *E1*, the data type determined by applying Subclause 9.3, "Set operation result data types", to the i-th <value expression>s of the other <row value constructor>s that are not *E1*.

In General Rule 5) p), replace "each" with "some".

14. Rationale: Clarify the use of <dynamic parameter specification>s in <in value list>.

Replace General Rule 5) r) with the following:

r) In an <in predicate> that specifies an <in value list>, if the <row value constructor> is not *E1*, then let *D* be its date type. Otherwise, let *D* be the data type determined by applying Subclause 9.3, "Set operation result data types", to the <value specification>s of the <in value list> that are not *E1*. The data type of any *E1* in the <in predicate> is assumed to be *D*.

15. Rationale: Define the use of <dynamic parameter specification>s in <result expression>s of <case specification>s.

Add the following to General Rule 5):

w.1) If any <result expression> in a <case specification> is *E1*, then the data type of *E1* is the data type that is determined by applying Subclause 9.3, "Set operation result data types", to all the <result expression>s that are not *E1*.

16. Rationale: Clarify the use of a <dynamic parameter specification> in a <between predicate>.

Add the following to General Rule 5):

w.2) In an operand of a <between predicate>, if the *i*-th value of one operand is *E1*, then the data type of *E1* is the data type determined by applying Subclause 9.3, "Set operation result data types" using the *i*-th value of the other operands.

17. Rationale: Editorial.

In General Rule 8), replace "<simple target specification>" with "<simple value specification>".

18. Rationale: The following corrects an incorrectly structured General Rule.

Replace General Rule 9) b) with the following:

b) If <statement name> is specified for the <SQL statement name>, then

Case:

i) If *P* is not a <cursor specification> and <statement name> is associated with a cursor *C* through a <dynamic declare cursor>, then an exception is raised: *dynamic SQL error—prepared statement is not a cursor specification.*

ii) Otherwise:

1) If <statement name> is not associated with a cursor and either *P* is not a <cursor specification> or *P* is a <cursor specification> that conforms to the Format and Syntax Rules of a <dynamic single row select statement>, then the same <statement name> shall be specified for each <execute statement> that is to be associated with this prepared statement.

2) If *P* is a <cursor specification> and <statement name> is associated with a cursor *C* through a <dynamic declare cursor>, then an association is made between *C* and *P*. The association is preserved until the prepared statement is destroyed.

17.9 <using clause>

1. *Rationale: Set the LENGTH in the descriptor area for <interval type>s.*

In General Rule 3) e) iv) 6), insert "LENGTH is set to the length in positions of the interval type, " before "DATETIME_INTERVAL_CODE".

17.10 <execute statement>

1. *Rationale: Rule improperly fails to address <parameter using clause> option.*

In General Rule 6), delete the phrase "that is <using descriptor>".

17.11 <execute immediate statement>

1. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

In General Rule 3), replace "*syntax error or access rule violation in dynamic SQL statement*" with "*syntax error or access rule violation*".

17.15 <dynamic fetch statement>

1. *Rationale: Editorial.*

Replace Leveling Rule 1) with the following:

1) The following restrictions apply for Intermediate SQL:

None.

17.18 <dynamic update statement: positioned>

1. *Rationale: Editorial.*

In Format, replace "<set clause> [{ <comma> <set clause> } ...]" with "<set clause list>".

Delete Syntax Rules 7), 8), and 11).

18.1 <get diagnostics statement>

1. *Rationale: DYNAMIC_FUNCTION failed to provide information about errors discovered during execution of a <prepare statement> about the statement being prepared. The values of COMMAND_FUNCTION and DYNAMIC_FUNCTION fail to account for implementation-defined statements and statements that are completely unrecognized by the implementation.*

In General Rule 1) d), in the first sentence, replace "the prepared statement executed" with "the statement being prepared or executed dynamically".

In Table 22, "SQL-statement character codes for use in the diagnostics area", add the following two entries at the end of the table:

Implementation-defined statements	An implementation-defined character string value different from the value associated with any other SQL-statement
Unrecognized statements	A zero-length string

2. *Rationale: CATALOG_NAME, SCHEMA_NAME, and TABLE_NAME do not always have applicable values in GET DIAGNOSTICS for a syntax error or access rule violation.*

In General Rule 3) f) ii), in the last sentence, replace "<space>s" with "a zero-length string".

In General Rule 3) f) ii) 3), replace "<space>s" with "zero-length strings".

3. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

Replace the first paragraph of General Rule 3) g) with the following:

g) If the value of RETURNED_SQLSTATE corresponds to *syntax error or access rule violation*, then:

4. *Rationale: CATALOG_NAME, SCHEMA_NAME, and TABLE_NAME do not always have applicable values in GET DIAGNOSTICS for a syntax error or access rule violation. The rule is also restructured to remove an internal ambiguity.*

Replace General Rule 3) g) i) with the following:

i) Case:

1) If the syntax error or access rule violation was caused by reference to a specific table, then the values of CATALOG_NAME, SCHEMA_NAME, and TABLE_NAME are:

Case:

i) If the specific table referred to was not a declared local temporary table, then <catalog name>, the <unqualified schema name> of the <schema name> of the schema that contains the table that caused the syntax error or access rule violation, and the <qualified identifier>, respectively.

ii) Otherwise, a zero length string, "MODULE" and the <local table name>, respectively.

2) Otherwise, CATALOG_NAME, SCHEMA_NAME and TABLE_NAME contain a zero-length string.

In General Rule 3) g) ii), replace "<space>s" with "a zero-length string".

5. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

Replace the first paragraph of General Rule 3) j) with the following:

j) If the value of RETURNED_SQLSTATE does not correspond to *syntax error or access rule violation*, then:

6. *Rationale: In addition to <SQL connection statement>s, both <commit statement> and <rollback statement> can affect multiple servers and potentially give rise to condition information for more than one server.*

Replace General Rule 3) n) with the following:

n) The values of CONNECTION_NAME and SERVER_NAME are respectively:

Case:

i) If COMMAND_FUNCTION or DYNAMIC_FUNCTION identifies an <SQL-connection statement>, then the <connection name> and <SQL-server name> specified or implied by the <SQL connection statement>.

ii) Otherwise, the <connection name> and <SQL-server name> of the SQL-session in which the condition was raised.

Note: If COMMAND_FUNCTION or DYNAMIC_FUNCTION identifies an SQL-statement that can affect more than one SQL-session (e.g., <commit statement>, <rollback statement>, DISCONNECT ALL), then there may be one or more conditions for each of the SQL-servers involved.

7. *Rationale: Editorial.*

In Leveling Rule 1), replace "SQL;" with "SQL:".

In Leveling Rule 2), replace "SQL;" with "SQL in addition to any Intermediate SQL restrictions:".

19.1 <embedded SQL host program>

1. *Rationale: Prohibit multiple <embedded character set declaration>s, the effects of which are undefined.*

Replace Syntax Rule 6) with the following:

6) An <embedded SQL host program> shall contain no more than one <embedded character set declaration>. If an <embedded character set declaration> is not specified, then an <embedded character set declaration> that specifies an implementation-defined character set that contains at least every character that is in <SQL language character> is implicit.

19.5 <embedded SQL COBOL program>

1. *Rationale: Incorrect use of PICTURE B in COBOL syntax.*

In Format, replace the BNF for <COBOL bit type> with the following:

```
<COBOL bit type> ::=
        { PIC | PICTURE } [ IS ]
        { X [ <left paren> <length> <right paren> ] } ...
        USAGE IS BIT
```

Replace Syntax Rule 5) b) with the following:

b) The syntax "USAGE IS BIT" shall be deleted in any <COBOL bit type>.

20.1 <direct SQL statement>

1. *Rationale: The following unifies the SQLSTATE returned for the different ways of invoking an SQL statement.*

In General Rule 5), replace "*syntax error or access rule violation in direct SQL statement*" with "*syntax error or access rule violation* ".

2. *Rationale: Clarify with respect to the language added to Subclause 4.18.1; to permit implementation-defined 01 warnings to be returned in Direct SQL; to correct typos.*

Replace General Rule 8) with the following:

8) Case:

a) If *S* executed successfully, then either a completion condition is raised: *successful completion*, or a completion condition is raised: *warning*, or a completion condition is raised: *no data*.

b) If *S* did not execute successfully, then all changes made to SQL-data or schemas by the execution of *S* are cancelled and an exception condition is raised.

Note: The method of raising a condition is implementation-defined.

21.1 Introduction

1. *Rationale: Clarification of representation of <identifier>s in the Information Schema base tables and views.*

Replace the last paragraph with the following:

The representation of an <identifier> in the base tables and views of the Information Schema is by a character string corresponding to its <identifier body> (in the case of a <regular identifier>) or its <delimited identifier body> (in the case of a <delimited identifier>). Within this character string, any lower case letter appearing in a <regular identifier> is replaced by the corresponding upper case letter, and any <doublequote symbol> appearing in a <delimited identifier body> is replaced by a <double quote>. Where an <actual identifier> has multiple forms that are equal according to the rules of Subclause 8.2, "<comparison predicate>", the form stored is that encountered at definition time.

21.2.2 INFORMATION_SCHEMA_CATALOG_NAME base table

1. *Rationale: This base table is supposed to contain a single row; however, its primary key does not properly reflect this constraint.*

Replace Definition with the following:

```
CREATE TABLE INFORMATION_SCHEMA_CATALOG_NAME
    ( CATALOG_NAME SQL_IDENTIFIER,
    CONSTRAINT INFORMATION_SCHEMA_CATALOG_NAME_PRIMARY_KEY
     PRIMARY KEY ( CATALOG_NAME ),
    CONSTRAINT INFORMATION_SCHEMA_CATALOG_NAME_CHECK
     CHECK ( ( SELECT COUNT (*) FROM INFORMATION_SCHEMA_CATALOG_NAME ) = 1 ) )
```

2. *Rationale: Editorial.*

Replace the second occurrence of "Definition" with "Description".

21.2.3 INFORMATION_SCHEMA_CATALOG_NAME_CARDINALITY assertion

1. *Rationale: Editorial.*

Delete the Description section.

Add the following at the end of the Subclause:

Leveling Rules

1) The following restrictions apply for Intermediate SQL:

None.

2) The following restrictions apply for Entry SQL in addition to any Intermediate SQL restrictions:

a) Conforming Entry SQL language shall not reference the Information Schema.

21.2.5 DOMAINS view

1. *Rationale: In support of changes to 21.3.5.*

In the Definition, replace "DATETIME_PRECISION, DOMAIN_DEFAULT" with "DATETIME_PRECISION, INTERVAL_TYPE, INTERVAL_PRECISION, DOMAIN_DEFAULT".

21.2.6 DOMAIN_CONSTRAINTS view

1. *Rationale: Editorial.*

In Definition, replace "(S.CATALOG_NAME, SCHEMA_NAME S)" with "(S.CATALOG_NAME, S.SCHEMA_NAME)".

21.2.9 COLUMNS view

1. *Rationale: In support of changes to 21.3.5.*

In the Definition, insert the following lines immediately after the line that reads "COALESCE (D1.DATETIME_PRECISION, D2.DATETIME_PRECISION) AS DATETIME_PRECISION,":

```
COALESCE ( D1.INTERVAL_TYPE, D2.INTERVAL_TYPE ) AS INTERVAL_TYPE,
COALESCE ( D1.INTERVAL_PRECISION, D2.INTERVAL_PRECISION ) AS INTERVAL_PRECISION,
```

21.2.17 ASSERTIONS view

1. *Rationale: The ASSERTIONS view should appear in Full SQL along with <assertion definition>.*

Replace Leveling Rules 1) and 2) with:

1) The following restrictions apply for Intermediate SQL:

   a) Conforming Intermediate SQL language shall not reference the ASSERTIONS view.

2) The following restrictions apply for Entry SQL in addition to any Intermediate SQL restrictions:

   None.

21.2.27 SQL_IDENTIFIER domain

1. *Rationale: Clarification of the representation of <identifier>s in the Information Schema base tables and views.*

Replace the Function with the following:

Define a domain that contains all valid <identifier body>s and <delimited identifier body>s.

Replace Description 1) with the following:

1) This domain specifies all variable-length character values that conform to the rules for formation and representation of an <identifier body> or an <delimited identifier body>.
   Note: There is no way in SQL to specify a <domain constraint> that would be true for the body of any valid <regular identifier> or <delimited identifier> and false for all other character string values.

Replace Description 2) with the following:

2) *L* is the implementation-defined maximum length of <identifier body> and <delimited identifier body>.

21.3.5 DATA_TYPE_DESCRIPTOR base table

1. *Rationale: The descriptions of the existing columns do not allow for the fact that the interval type has two precisions: the fractional seconds precision and the interval leading field precision. In addition, the interval type also specifies the interval qualifier which must be encoded into the DATA_TYPE_DESCRIPTOR. Finally, numeric precision should not be allowed to be null for INTEGER, SMALLINT, NUMERIC, DECIMAL, DATE, and INTERVAL data types.*
   Also because DECIMAL and NUMERIC always have a radix of 10, even when the scale is 0 and to permit implementation-defined data types to be represented in the Information Schema changes are made to the TABLE_OR_DOMAIN_CHECK_COMBINATIONS.

Replace Definition with the following:

```
CREATE TABLE DATA_TYPE_DESCRIPTOR
   (
   TABLE_OR_DOMAIN_CATALOG                INFORMATION_SCHEMA.SQL_IDENTIFIER,
   TABLE_OR_DOMAIN_SCHEMA                 INFORMATION_SCHEMA.SQL_IDENTIFIER,
   TABLE_OR_DOMAIN_NAME                   INFORMATION_SCHEMA.SQL_IDENTIFIER,
   COLUMN_NAME                            INFORMATION_SCHEMA.SQL_IDENTIFIER,
   DATA_TYPE                              INFORMATION_SCHEMA.SQL_IDENTIFIER
```

```
  CONSTRAINT TABLE_OR_DOMAIN_DATA_TYPE_NOT_NULL NOT NULL,
CHARACTER_MAXIMUM_LENGTH                INFORMATION_SCHEMA.CARDINAL_NUMBER,
CHARACTER_OCTET_LENGTH                  INFORMATION_SCHEMA.CARDINAL_NUMBER,
COLLATION_CATALOG                       INFORMATION_SCHEMA.SQL_IDENTIFIER,
COLLATION_SCHEMA                        INFORMATION_SCHEMA.SQL_IDENTIFIER,
COLLATION_NAME                          INFORMATION_SCHEMA.SQL_IDENTIFIER,
NUMERIC_PRECISION                       INFORMATION_SCHEMA.CARDINAL_NUMBER,
NUMERIC_PRECISION_RADIX                 INFORMATION_SCHEMA.CARDINAL_NUMBER,
NUMERIC_SCALE                           INFORMATION_SCHEMA.CARDINAL_NUMBER,
DATETIME_PRECISION                      INFORMATION_SCHEMA.CARDINAL_NUMBER,
INTERVAL_TYPE                           INFORMATION_SCHEMA.CHARACTER_DATA,
INTERVAL_PRECISION                      INFORMATION_SCHEMA.CARDINAL_NUMBER,
CONSTRAINT TABLE_OR_DOMAIN_CHECK_COMBINATIONS
  CHECK ( DATA_TYPE NOT IN ('CHARACTER', 'CHARACTER VARYING', 'BIT',
                            'BIT VARYING', 'REAL', 'DOUBLE PRECISION',
                            'FLOAT', 'INTEGER', 'SMALLINT', 'NUMERIC',
                            'DECIMAL', 'DATE', 'TIME', 'TIMESTAMP',
                            'TIME WITH TIME ZONE',
                            'TIMESTAMP WITH TIME ZONE', 'INTERVAL')
      OR (
           DATA_TYPE IN ( 'CHARACTER', 'CHARACTER VARYING', 'BIT',
                          'BIT VARYING' )
           AND
           ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
             COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME )
             IS NOT NULL
           AND
           ( NUMERIC_PRECISION, NUMERIC_PRECISION_RADIX, NUMERIC_SCALE,
             DATETIME_PRECISION ) IS NULL
           AND
           ( INTERVAL_TYPE, INTERVAL_PRECISION ) IS NULL
         OR
           DATA_TYPE IN ( 'REAL', 'DOUBLE PRECISION', 'FLOAT' )
           AND
           ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
             COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IS NULL
           AND
           NUMERIC_PRECISION IS NOT NULL
           AND
           NUMERIC_PRECISION_RADIX = 2
           AND
           NUMERIC_SCALE IS NULL
           AND
           DATETIME_PRECISION IS NULL
           AND
           ( INTERVAL_TYPE, INTERVAL_PRECISION ) IS NULL
         OR
           DATA_TYPE IN ( 'INTEGER', 'SMALLINT')
           AND
           ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
             COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IS NULL
           AND
           NUMERIC_PRECISION_RADIX IN ( 2, 10 ) )
           AND
           NUMERIC_PRECISION IS NOT NULL
           AND
           NUMERIC_SCALE = 0
           AND
           DATETIME_PRECISION IS NULL
           AND
           ( INTERVAL_TYPE, INTERVAL_PRECISION ) IS NULL
         OR
           DATA_TYPE IN ( 'NUMERIC', 'DECIMAL' )
           AND
           ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
             COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IS NULL
           AND
```

```
                NUMERIC_PRECISION_RADIX = 10
                AND
                ( NUMERIC_PRECISION, NUMERIC_SCALE ) IS NOT NULL
                AND
                DATETIME_PRECISION IS NULL
                AND
                ( INTERVAL_TYPE, INTERVAL_PRECISION ) IS NULL
              OR
                DATA_TYPE IN ( 'DATE', 'TIME', 'TIMESTAMP',
                               'TIME WITH TIME ZONE',
                               'TIMESTAMP WITH TIME ZONE' )
                AND
                ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
                  COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IS NULL
                AND
                (NUMERIC_PRECISION, NUMERIC_PRECISION_RADIX ) IS NULL
                AND
                NUMERIC_SCALE IS NULL
                AND
                DATETIME_PRECISION IS NOT NULL
                AND
                ( INTERVAL_TYPE, INTERVAL_PRECISION ) IS NULL
              OR
                DATA_TYPE = 'INTERVAL'
                AND
                ( CHARACTER_MAXIMUM_LENGTH, CHARACTER_OCTET_LENGTH,
                  COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IS NULL
                AND
                (NUMERIC_PRECISION, NUMERIC_PRECISION_RADIX ) IS NULL
                AND
                NUMERIC_SCALE IS NULL
                AND
                DATETIME_PRECISION IS NOT NULL
                AND
                INTERVAL_TYPE IN ( 'YEAR', 'MONTH', 'DAY', 'HOUR', 'MINUTE',
                                   'SECOND', 'YEAR TO MONTH', 'DAY TO HOUR',
                                   'DAY TO MINUTE', 'DAY TO SECOND',
                                   'HOUR TO MINUTE', HOUR TO SECOND',
                                   'MINUTE TO SECOND' )
                AND
                INTERVAL_PRECISION IS NOT NULL
            ) ),

    CONSTRAINT DATA_TYPE_DESCRIPTOR_PRIMARY_KEY
      PRIMARY KEY ( TABLE_OR_DOMAIN_CATALOG, TABLE_OR_DOMAIN_SCHEMA,
                    TABLE_OR_DOMAIN_NAME, COLUMN_NAME ),

    CONSTRAINT DATA_TYPE_CHECK_REFERENCES_COLLATION
      CHECK ( COLLATION_CATALOG <> ANY ( SELECT CATALOG_NAME FROM SCHEMATA )
              OR
              ( COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME ) IN
              ( SELECT COLLATION_CATALOG, COLLATION_SCHEMA, COLLATION_NAME
                  FROM COLLATIONS ) ),

    CONSTRAINT DATA_TYPE_DESCRIPTOR_CHECK_USED
      CHECK ( ( TABLE_OR_DOMAIN_CATALOG, TABLE_OR_DOMAIN_SCHEMA,
                TABLE_OR_DOMAIN_NAME, COLUMN_NAME )
              IN ( SELECT TABLE_CATALOG, TABLE_SCHEMA, TABLE_NAME, COLUMN_NAME
                     FROM COLUMNS
                   UNION
                   SELECT DOMAIN_CATALOG, DOMAIN_SCHEMA, DOMAIN_NAME, ' '
                     FROM DOMAINS )
            )
)
```

2. *Rationale: Editorial.*

In Description 3), replace "the precision if it is numeric type, and, and the precision if it is a datetime or interval type" with "the precision if it is numeric type, the scale if it is a numeric type, and the fractional seconds precision if it is a datetime or interval type".

3. *Rationale: The descriptions of the existing columns do not allow for the fact that the interval type has two precisions: the fractional seconds precision and the interval leading field precision. In addition, the interval type also specifies the interval qualifier which must be encoded into the DATA_TYPE_DESCRIPTOR. Finally, numeric precision should not be allowed to be null for INTEGER, SMALLINT, NUMERIC, DECIMAL, DATE, and INTERVAL data types.*

Add the following Descriptions:

3.1) If DATA_TYPE is 'INTERVAL', then the values of INTERVAL_TYPE are the value for <interval qualifier> for the data type being described, excluding any <interval leading field precision> and an <interval fractional seconds precision>; otherwise, INTERVAL_TYPE is the null value.

3.2) If DATA_TYPE is 'INTERVAL', then the values of INTERVAL_PRECISION are the interval leading field precision of the data type being described; otherwise, INTERVAL_PRECISION is the null value.

21.3.6 DOMAINS base table

1. *Rationale: The following helps define the action taken when a default value cannot be represented in the Information Schema.*

In Description, replace the first sentence of Item 3) with the following:

The value of DOMAIN_DEFAULT is null if the domain being described has no explicit default value; the value of DOMAIN_DEFAULT is TRUNCATED if the character representation of the <default option> cannot be represented without truncation.

In Description, add the following at the end of Item 6):

Note: TRUNCATED is different from other values like CURRENT_USER, SYSTEM_USER, or CURRENT_DATE in that it is not an SQL keyword and does not correspond to a defined value in SQL.

21.3.8 TABLES base table

1. *Rationale: Editorial.*

In Definition, in the last constraint, change "EXCEPTIONEXISTS" to "EXISTS".

21.3.10 COLUMNS base table

1. *Rationale: Provide General Rules for maintaining the nullability characteristic in a column definition.*

In Definition, insert the following immediately after the column "IS_NULLABLE":

```
CONSTRAINT IS_NULLABLE_NOT_NULL NOT NULL
CONSTRAINT IS_NULLABLE_CHECK
      CHECK ( IS_NULLABLE IN ( 'YES', 'NO' ) ),
```

2. *Rationale: The following helps define the action taken when a default value cannot be represented in the Information Schema.*

In Description, replace the first sentence of Item 6) with the following:

The value of COLUMN_DEFAULT is null if the column being described has no explicit default value or if its default value comes only from a domain; the value of COLUMN_DEFAULT is TRUNCATED if the character representation of the <default option> cannot be represented without truncation.
Note: TRUNCATED is different from other values like CURRENT_USER, SYSTEM_USER, or CURRENT_DATE in that it is not an SQL keyword and does not correspond to a defined value in SQL.

21.3.11 VIEW_TABLE_USAGE base table

1. *Rationale: Clearly define which referenced tables belong in VIEW_TABLE_USAGE.*

In Function, replace "referenced in" with "identified by a <table name> simply contained in a <table reference> that is contained in".

In Description 2), replace "view requires" with "identified by a <table name> simply contained in a <table reference> that is contained in the <query expression> of the view being described".

21.3.12 VIEW_COLUMN_USAGE base table

1. *Rationale: Clearly define which referenced columns belong in VIEW_COLUMN_USAGE.*

In Function, replace "referenced by a view" with "identified by a <table name> simply contained in a <table reference> that is contained in the <query expression> of the view being described".

In Description 2), replace "identifier, respectively, of a column" with "column name, respectively, of a column of a table identified by a <table name> simply contained in a <table reference> that is contained in the <query expression> of the view".

21.3.13 TABLE_CONSTRAINTS base table

1. *Rationale: In the TABLE_CONSTRAINTS base table, the table constraint TABLE_CONSTRAINTS_CHECK_REFERENCES_TABLES specifies that if TABLE_CATALOG is referenced in the SCHEMATA base table, then the table specified by TABLE_CATALOG, TABLE_SCHEMA, and TABLE_NAME must be defined in the TABLES base table. The table constraint TABLE_CONSTRAINTS_CHECK_VIEWS specifies that if TABLE_CATALOG is referenced in the SCHEMATA base table, then not only must the table specified by TABLE_CATALOG, TABLE_SCHEMA, and TABLE_NAME be defined in the TABLES base table, it must also not be a view. Since any row that satisfies the table constraint TABLE_CONSTRAINT_CHECK_VIEWS will also satisfy the table constraint TABLE_CONSTRAINT_CHECK_TABLES, the table constraint TABLE_CONSTRAINT_CHECK_TABLES is redundant.*

In Definition, delete the constraint `TABLE_CONSTRAINTS_CHECK_REFERENCES_TABLES`.