



BSI Standards Publication

Information technology — Database languages SQL

Part 4: Persistent stored modules (SQL/PSM)

National foreword

This British Standard is the UK implementation of ISO/IEC 9075-4:2023.

The UK participation in its preparation was entrusted to Technical Committee IST/40, Data management and interchange.

A list of organizations represented on this committee can be obtained on request to its committee manager.

Contractual and legal considerations

This publication has been prepared in good faith, however no representation, warranty, assurance or undertaking (express or implied) is or will be made, and no responsibility or liability is or will be accepted by BSI in relation to the adequacy, accuracy, completeness or reasonableness of this publication. All and any such responsibility and liability is expressly disclaimed to the full extent permitted by the law.

This publication is provided as is, and is to be used at the recipient's own risk.

The recipient is advised to consider seeking professional guidance with respect to its use of this publication.

This publication is not intended to constitute a contract. Users are responsible for its correct application.

© The British Standards Institution 2023
Published by BSI Standards Limited 2023

ISBN 978 0 539 16120 5

ICS 35.060

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 June 2023.

Amendments/corrigenda issued since publication

Date	Text affected

Information technology — Database languages SQL —

Part 4: Persistent stored modules (SQL/PSM)

*Technologies de l'information — Langages de base de données SQL —
Partie 4: Modules stockés persistants (SQL/PSM)*



Reference number
ISO/IEC 9075-4:2023(E)

AVA Documents
Document Sharing Hub
© ISO/IEC 2023



COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2023

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword.....	viii
Introduction.....	x
1 Scope.....	1
2 Normative references.....	2
3 Terms and definitions.....	3
4 Concepts.....	4
4.1 Notations and conventions.....	4
4.1.1 Notations.....	4
4.1.2 Use of terms.....	4
4.2 SQL-server modules.....	4
4.3 Tables.....	5
4.3.1 Base tables.....	5
4.3.1.1 Temporary tables.....	5
4.3.2 Unique identification of tables.....	5
4.4 SQL-schemas.....	5
4.5 SQL-invoked routines.....	6
4.5.1 Overview of SQL-invoked routines.....	6
4.5.2 Execution of conventional SQL-invoked routines.....	6
4.5.3 Routine descriptors.....	6
4.6 SQL-paths.....	6
4.7 Host parameters.....	6
4.7.1 Status parameters.....	6
4.8 Diagnostics area.....	6
4.9 Condition handling.....	7
4.10 Cursors.....	8
4.10.1 General description of cursors.....	8
4.11 SQL-statements.....	9
4.11.1 Classes of SQL-statements.....	9
4.11.2 SQL-statements classified by function.....	9
4.11.2.1 SQL-schema statements.....	9
4.11.2.2 SQL-control statements.....	9
4.11.2.3 SQL-control declarations.....	9
4.11.2.4 SQL-diagnostics statements.....	10
4.11.3 SQL-statements and transaction states.....	10
4.11.4 SQL-statement atomicity and statement execution contexts.....	10
4.11.5 Embeddable SQL-statements.....	10
4.11.6 Preparable and immediately executable SQL-statements.....	11
4.11.7 Directly executable SQL-statements.....	11

4.11.8	Iterated SQL-statements.	11
4.11.9	Compound statements.	12
4.12	Basic security model.	12
4.12.1	Privileges.	12
4.13	SQL-sessions.	12
4.13.1	General description of SQL-sessions.	12
5	Lexical elements.	13
5.1	<token> and <separator>.	13
5.2	Names and identifiers.	15
6	Scalar expressions.	17
6.1	<value specification> and <target specification>.	17
6.2	<identifier chain>.	19
6.3	<next value expression>.	21
6.4	<SQL variable reference>.	22
7	Query expressions.	23
7.1	<table reference>.	23
7.2	<query specification>.	24
8	Additional common rules.	25
8.1	Potential sources of non-determinism.	25
8.2	Executing an <SQL procedure statement>.	26
8.3	Executing a <compound statement>.	27
8.4	Invoking an SQL-invoked routine.	29
8.5	Transformation of query specifications.	30
8.6	Handler execution.	31
9	Additional common elements.	34
9.1	<sqlstate value>.	34
10	Schema definition and manipulation.	36
10.1	<schema definition>.	36
10.2	<drop schema statement>.	37
10.3	<table definition>.	38
10.4	<column definition>.	39
10.5	<default clause>.	40
10.6	<check constraint definition>.	41
10.7	<drop column scope clause>.	42
10.8	<drop column definition>.	43
10.9	<drop table constraint definition>.	44
10.10	<drop table statement>.	45
10.11	<view definition>.	46
10.12	<drop view statement>.	47
10.13	<drop domain statement>.	48
10.14	<drop character set statement>.	49
10.15	<drop collation statement>.	50
10.16	<drop transliteration statement>.	51
10.17	<assertion definition>.	52
10.18	<drop assertion statement>.	53

10.19	<trigger definition>.....	54
10.20	<drop user-defined ordering statement>.....	55
10.21	<SQL-server module definition>.....	56
10.22	<drop module statement>.....	59
10.23	<drop data type statement>.....	60
10.24	<SQL-invoked routine>.....	61
10.25	<drop routine statement>.....	63
10.26	<drop user-defined cast statement>.....	64
11	Access control.....	65
11.1	<grant statement>.....	65
11.2	<privileges>.....	66
11.3	<revoke statement>.....	67
12	SQL-client modules.....	71
12.1	<externally-invoked procedure>.....	71
12.2	<SQL procedure statement>.....	73
13	Data manipulation.....	74
13.1	<declare cursor>.....	74
13.2	<open statement>.....	75
13.3	<fetch statement>.....	76
13.4	<close statement>.....	77
13.5	<select statement: single row>.....	78
13.6	<delete statement: positioned>.....	79
13.7	<update statement: positioned>.....	80
13.8	<temporary table declaration>.....	81
14	Additional data manipulation rules.....	82
14.1	Effect of opening a cursor.....	82
15	Control statements.....	83
15.1	<compound statement>.....	83
15.2	<handler declaration>.....	86
15.3	<condition declaration>.....	88
15.4	<SQL variable declaration>.....	89
15.5	<assignment statement>.....	90
15.6	<case statement>.....	94
15.7	<if statement>.....	97
15.8	<iterate statement>.....	99
15.9	<leave statement>.....	100
15.10	<loop statement>.....	101
15.11	<while statement>.....	102
15.12	<repeat statement>.....	104
15.13	<for statement>.....	105
16	Dynamic SQL.....	108
16.1	<prepare statement>.....	108
16.2	<input using clause>.....	110
16.3	<output using clause>.....	111
17	Embedded SQL.....	112

17.1	<embedded SQL host program>.....	112
18	Direct invocation of SQL.....	114
18.1	<direct SQL statement>.....	114
19	Diagnostics management.....	115
19.1	<get diagnostics statement>.....	115
19.2	<signal statement>.....	117
19.3	<resignal statement>.....	119
20	Information Schema.....	121
20.1	Information Schema digital artifact.....	121
20.2	MODULE_COLUMN_USAGE view.....	121
20.3	MODULE_PRIVILEGES view.....	123
20.4	MODULE_TABLE_USAGE view.....	124
20.5	MODULES view.....	125
20.6	PARAMETERS view.....	126
20.7	ROLE_MODULE_GRANTS view.....	127
20.8	ROUTINES view.....	128
20.9	Short name views.....	129
21	Definition Schema.....	130
21.1	Definition Schema digital artifact.....	130
21.2	MODULE_COLUMN_USAGE base table.....	130
21.3	MODULE_PRIVILEGES base table.....	132
21.4	MODULE_TABLE_USAGE base table.....	134
21.5	MODULES base table.....	135
21.6	ROUTINES base table.....	137
22	Status codes.....	138
22.1	SQLSTATE.....	138
23	Conformance.....	139
23.1	Claims of conformance to SQL/PSM.....	139
23.2	Additional conformance requirements for SQL/PSM.....	139
23.3	Implied feature relationships of SQL/PSM.....	139
Annex A (informative) SQL conformance summary.....		141
Annex B (informative) Implementation-defined elements.....		146
Annex C (informative) Implementation-dependent elements.....		148
Annex D (informative) SQL optional feature taxonomy.....		150
Annex E (informative) Deprecated features.....		152
Annex F (informative) Incompatibilities with ISO/IEC 9075:2016.....		153
Annex G (informative) Defect Reports not addressed in this edition of this document.....		154
Index.....		155

Tables

Table	Page
1 Data types of <condition information item name>s.	115
2 SQL-statement codes.	116
3 SQLSTATE class and subclass codes.	138
4 Implied feature relationships of SQL/PSM.	139
A.1 Feature definitions outside of Conformance Rules.	141
D.1 Feature taxonomy for optional features.	150

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC have not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and <https://patents.iec.ch>. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange*.

This seventh edition cancels and replaces the sixth edition (ISO/IEC 9075-4:2016), which has been technically revised. It also incorporates the Technical Corrigenda ISO/IEC 9075-4:2016/Cor.1:2019 and ISO/IEC 9075-4:2016/Cor.2:2022.

The main changes are as follows:

- improve the presentation and accuracy of the summaries of implementation-defined and implementation-dependent aspects of this document;
- introduction of several digital artifacts;
- alignment with updated ISO house style and other guidelines for creating standards.

This seventh edition of ISO/IEC 9075-4 is designed to be used in conjunction with the following editions of other parts of the ISO/IEC 9075 series, all published 2023:

- ISO/IEC 9075-1, sixth edition;
- ISO/IEC 9075-2, sixth edition;
- ISO/IEC 9075-3, sixth edition;
- ISO/IEC 9075-9, fifth edition;
- ISO/IEC 9075-10, fifth edition;
- ISO/IEC 9075-11, fifth edition;
- ISO/IEC 9075-13, fifth edition;
- ISO/IEC 9075-14, sixth edition;
- ISO/IEC 9075-15, second edition;
- ISO/IEC 9075-16, first edition.

A list of all parts in the ISO/IEC 9075 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

The organization of this document is as follows:

- 1) [Clause 1, “Scope”](#), specifies the scope of this document.
- 2) [Clause 2, “Normative references”](#), identifies additional standards that, through reference in this document, constitute provisions of this document.
- 3) [Clause 3, “Terms and definitions”](#), defines the terms and definitions used in this document.
- 4) [Clause 4, “Concepts”](#), presents concepts used in the definition of persistent stored modules.
- 5) [Clause 5, “Lexical elements”](#), defines a number of lexical elements used in the definition of persistent stored modules.
- 6) [Clause 6, “Scalar expressions”](#), defines a number of scalar expressions used in the definition of persistent stored modules.
- 7) [Clause 7, “Query expressions”](#), defines the elements of the language that produce rows and tables of data as used in persistent stored modules.
- 8) [Clause 8, “Additional common rules”](#), specifies the rules for assignments that retrieve data from or store data into SQL-data, and formation rules for set operations.
- 9) [Clause 9, “Additional common elements”](#), defines additional common elements used in the definition of persistent stored modules.
- 10) [Clause 10, “Schema definition and manipulation”](#), defines the schema definition and manipulation statements associated with the definition of persistent stored modules.
- 11) [Clause 11, “Access control”](#), defines facilities for controlling access to SQL-data.
- 12) [Clause 12, “SQL-client modules”](#), defines the facilities for using persistent stored modules.
- 13) [Clause 13, “Data manipulation”](#), defines data manipulation operations associated with persistent stored modules.
- 14) [Clause 14, “Additional data manipulation rules”](#), defines additional rules for data manipulation.
- 15) [Clause 15, “Control statements”](#), defines the control statements used with persistent stored modules.
- 16) [Clause 16, “Dynamic SQL”](#), defines the facilities for executing SQL-statements dynamically in the context of persistent stored modules.
- 17) [Clause 17, “Embedded SQL”](#), defines the host language embeddings.
- 18) [Clause 18, “Direct invocation of SQL”](#), defines direct invocation of SQL language.
- 19) [Clause 19, “Diagnostics management”](#), defines enhancements to the facilities used with persistent stored modules.
- 20) [Clause 20, “Information Schema”](#), defines the Information and Definition Schema objects associated with persistent stored modules.
- 21) [Clause 21, “Definition Schema”](#), defines base tables on which the viewed tables containing schema information depend.
- 22) [Clause 22, “Status codes”](#), defines SQLSTATE values related to persistent stored modules.
- 23) [Clause 23, “Conformance”](#), defines the criteria for conformance to this document.

- 24) [Annex A, “SQL conformance summary”](#), is an informative Annex. It summarizes the conformance requirements of the SQL language.
- 25) [Annex B, “Implementation-defined elements”](#), is an informative Annex. It lists those features for which the body of this document states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or other aspect is partly or wholly implementation-defined.
- 26) [Annex C, “Implementation-dependent elements”](#), is an informative Annex. It lists those features for which the body of this document states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or other aspect is partly or wholly implementation-dependent.
- 27) [Annex D, “SQL optional feature taxonomy”](#), is an informative Annex. It identifies the optional features of the SQL language specified in this document by an identifier and a short descriptive name. This taxonomy is used to specify conformance.
- 28) [Annex E, “Deprecated features”](#), is an informative Annex. It lists features that the responsible Technical Committee intends not to include in a future edition of this document.
- 29) [Annex F, “Incompatibilities with ISO/IEC 9075:2016”](#), is an informative Annex. It lists incompatibilities with the previous edition of this document.
- 30) [Annex G, “Defect Reports not addressed in this edition of this document”](#), is an informative Annex. It describes the Defect Reports that were known at the time of publication of this document. Each of these problems is a problem carried forward from the previous edition of the ISO/IEC 9075 series. No new problems have been created in the drafting of this edition of this document.

In the text of this document, in [Clause 5, “Lexical elements”](#), through [Clause 23, “Conformance”](#), Subclauses begin new pages. Any resulting blank space is not significant.

Information technology — Database language SQL —

Part 4:

Persistent Stored Modules (SQL/PSM)

1 Scope

This document specifies the syntax and semantics of a database language for declaring and maintaining persistent database language routines in SQL-server modules.

The database language for <externally-invoked procedure>s and <SQL-invoked routine>s includes the following capabilities.

- The specification of statements to direct the flow of control.
- The assignment of the result of expressions to variables and parameters.
- The specification of condition handlers that allow SQL-invoked routines to deal with various conditions that arise during their execution.
- The specification of statements to signal and resignal conditions.
- The declaration of standing SQL-server cursors.
- The declaration of local variables.

It also includes the definition of the Information Schema tables that contain schema information pertaining to SQL-server modules and SQL-invoked routines.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 9075-1, *Information technology — Database languages — SQL — Part 1: Framework (SQL/Framework)*

ISO/IEC 9075-2, *Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation)*

ISO/IEC 9075-11, *Information technology — Database languages — SQL — Part 11: Information and Definition Schemas (SQL/Schemata)*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 9075-1 and ISO/IEC 9075-2 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <https://www.electropedia.org/>

4 Concepts

This Clause modifies *Clause 4, “Concepts”, in ISO/IEC 9075-2.*

4.1 Notations and conventions

This Subclause modifies *Subclause 4.1, “Notations and conventions”, in ISO/IEC 9075-2.*

4.1.1 Notations

This Subclause modifies *Subclause 4.1.1, “Notations”, in ISO/IEC 9075-2.*

The notations used in this document are defined in *ISO/IEC 9075-1.*

The syntax defined in this document is available from the ISO website as a “digital artifact”. See <https://standards.iso.org/iso-iec/9075/-4/ed-7/en/> to download digital artifacts for this document. To download the syntax defined in a plain-text format, select the file named *ISO_IEC_9075-4(E)_PSM.bnf.txt*. To download the syntax defined in an XML format, select the file named *ISO_IEC_9075-4(E)_PSM.bnf.xml*.

4.1.2 Use of terms

This Subclause modifies *Subclause 4.1.2, “Use of terms”, in ISO/IEC 9075-2.*

Insert after the 2nd paragraph: An SQL-statement *S1* is said to be executed as a *direct result of executing* an *<SQL control statement> S2* if *S2* contains *S1*.

Insert after the 2nd paragraph: The phrase “The scope of a *<handler declaration>* contained in a *Y* is that *Y*, excluding every *<SQL schema statement>* contained in that *Y*” means that the scope of the *<handler declaration>* does not extend to SQL-statements contained in such an *<SQL schema statement>*; it does, however, extend to the *<SQL schema statement>* itself.

4.2 SQL-server modules

An *SQL-server module* is a persistent object defined in a schema and identified by an *<SQL-server module name>*. SQL-server modules are created with *<SQL-server module definition>*s and destroyed with *<drop module statement>*s and by *<drop schema statement>*s that destroy the schemas that contain them.

An *<SQL-server module definition>* contains an *<SQL-server module name>*, an optional *<SQL-server module character set specification>*, an optional *<SQL-server module schema clause>*, an optional *<SQL-server module path specification>*, zero or more SQL-server module declared local temporary tables specified by *<temporary table declaration>*s, and one or more *<SQL-invoked routine>*s.

The *<SQL-server module name>* of an SQL-server module is a *<schema qualified name>*. The character set specified by the *<SQL-server module character set specification>* identifies the character repertoire used for expressing the names of schema objects used in the *<SQL-server module definition>*. The *<default schema name>* specified by the *<SQL-server module schema clause>* identifies the schema name used for implicit qualification of unqualified names appearing in the *<SQL-server module definition>*. The SQL-invoked routines of an SQL-server module are invoked only from SQL-statements.

An SQL-server module has an *SQL-server module authorization identifier*, which is set to the authorization identifier of the owner of the schema that contains the SQL-server module at the time the SQL-server module is created. The SQL-server module authorization identifier acts as the current authorization identifier for privilege determination for the SQL objects, if any, contained in the SQL-server module.

An SQL-server module is described by an SQL-server module descriptor. An SQL-server module descriptor includes the following components.

- The SQL-server module name of the SQL-server module.
- The descriptor of the character set in which the SQL-server module is represented.
- The default schema name used for implicit qualification of unqualified names in the SQL-server module.
- The SQL-server module authorization identifier of the SQL-server module.
- The list of schema names contained in the <SQL-server module path specification>.
- The table descriptor of every local temporary table declared in the SQL-server module.
- The descriptor of every SQL-invoked routine contained in the SQL-server module.
- The text of the <SQL-server module definition>.
- The creation timestamp, of an implementation-defined (IV115) timestamp type.

4.3 Tables

This Subclause modifies Subclause 4.17, “Tables”, in ISO/IEC 9075-2.

4.3.1 Base tables

This Subclause modifies Subclause 4.17.2, “Base tables”, in ISO/IEC 9075-2.

4.3.1.1 Temporary tables

This Subclause modifies Subclause 4.17.2.4, “Temporary tables”, in ISO/IEC 9075-2.

Insert after the 8th paragraph: An SQL-server module declared local temporary table is a declared local temporary table defined in an <SQL-server module definition>. An SQL-server module declared local temporary table is effectively materialized the first time any <module routine> in the <SQL-server module definition> that contains the <temporary table declaration> is executed, and it persists for that SQL-session. Every SQL-server module in every SQL-session that references an SQL-server module declared local temporary table causes a distinct instance of that declared local temporary table (i.e., a multiset of rows that is visible only to that SQL-server module during that SQL-session) to be materialized. That is, the multiset of rows that is referenced by the <table name> of an SQL-server module declared local temporary table cannot be shared between SQL-sessions, nor between SQL-server modules that execute during an SQL-session.

4.3.2 Unique identification of tables

This Subclause modifies Subclause 4.17.5, “Unique identification of tables”, in ISO/IEC 9075-2.

Insert into the 1st paragraph, after the 4th list item:

- The <table name> of an SQL-server module declared local temporary table, together with an SQL-session identifier and an SQL-server module name, uniquely identifies a multiset of rows.

4.4 SQL-schemas

This Subclause modifies Subclause 4.28, “SQL-schemas”, in ISO/IEC 9075-2.

Insert into the 2nd paragraph, after the last list item:

- An SQL-server module descriptor.

4.5 SQL-invoked routines

This Subclause modifies Subclause 4.35, “SQL-invoked routines”, in ISO/IEC 9075-2.

4.5.1 Overview of SQL-invoked routines

This Subclause modifies Subclause 4.35.1, “Overview of SQL-invoked routines”, in ISO/IEC 9075-2.

Insert after the 3rd paragraph: An SQL-invoked routine that is defined in an <SQL-server module definition> is called an *SQL-module-level routine*.

4.5.2 Execution of conventional SQL-invoked routines

This Subclause modifies Subclause 4.35.3, “Execution of conventional SQL-invoked routines”, in ISO/IEC 9075-2.

Augment the 2nd paragraph by adding “SQL-server module” to the list of the containing objects.

4.5.3 Routine descriptors

This Subclause modifies Subclause 4.35.5, “Routine descriptors”, in ISO/IEC 9075-2.

Insert into the 1st paragraph, after the 18th list item:

- If the SQL-invoked routine is not a schema-level routine, then the <SQL-server module name> of the SQL-server module that includes the SQL-invoked routine and the <schema name> of the schema that includes the SQL-server module.

4.6 SQL-paths

This Subclause modifies Subclause 4.36, “SQL-paths”, in ISO/IEC 9075-2.

Insert after the 3rd paragraph: The definition of SQL-server modules specify an SQL-path that is used to search for the subject routine of a <routine invocation> whose <routine name>s do not contain a <schema name> when the <routine invocation> is contained in the <SQL-server module definition>.

4.7 Host parameters

This Subclause modifies Subclause 4.37, “Host parameters”, in ISO/IEC 9075-2.

4.7.1 Status parameters

This Subclause modifies Subclause 4.37.2, “Status parameters”, in ISO/IEC 9075-2.

Insert after the 6th paragraph: If the <SQL procedure statement> is a <compound statement>, then the active condition may result from the action of some exception handler specified in the <compound statement>.

4.8 Diagnostics area

This Subclause modifies Subclause 4.38, “Diagnostics area”, in ISO/IEC 9075-2.

Insert after the 5th paragraph: Information about a completion or exception condition is placed into one or more condition areas of the first diagnostics area before any handler is activated. The diagnostics area stack is then pushed so that the handler can access that information even while its own execution is causing the first diagnostics area to be modified.

Insert after the 5th paragraph: The first diagnostics area is emptied during the execution of a <signal statement>. Information is added to the first diagnostics area during the execution of a <resignal statement>.

4.9 Condition handling

Condition handling is the method of handling exception and completion conditions in SQL/PSM. Condition handling provides a <handler declaration> to define a handler, specifying its type, the exception and completion conditions it can resolve, and the action it takes to do so. Condition handling also provides the ability to explicitly signal exception and completion conditions.

<handler declaration>s specify the handling of exception and completion conditions. <handler declaration>s are optionally specified in <compound statement>s. The scope of a <handler declaration> specified in <compound statement> *CS* is *CS*, excluding every <SQL schema statement> contained in *CS*.

A <handler declaration> associates one or more conditions with a handler action. The handler action is an <SQL procedure statement>.

A *general <handler declaration>* is one that is associated with the <condition value>s SQLEXCEPTION, SQLWARNING, or NOT FOUND. All other <handler declaration>s are *specific <handler declaration>s*.

A condition represents an error or informational state caused by execution of an <SQL procedure statement>. Conditions are raised to provide information in a diagnostics area about the execution of an <SQL procedure statement>.

A <condition declaration> is used to declare a <condition name>, and to optionally associate it with an SQLSTATE value. If a <condition declaration> does not specify an SQLSTATE value, it declares a *user-defined exception condition*. <condition name>s can be used in <handler declaration>s, <signal statement>s, and <resignal statement>s.

When the <compound statement> containing a <handler declaration> is executed, a handler is created for the conditions associated with that <handler declaration>. A created handler is *activated* when it is the most appropriate handler for an exception or completion condition that has been raised by an SQL-statement. Such a handler is an *active* handler.

The *most appropriate* handler is determined during execution of an implicit or explicit <resignal statement> or a handler execution. A handler cannot be the most appropriate handler for an exception condition that occurs during the execution of that handler. An implicit <resignal statement> is executed when a <compound statement> or <handler action> completes with a condition other than *successful completion (00000)*.

If there is no most appropriate handler and the condition is an exception condition, then the SQL-statement raising the exception condition is terminated with that exception condition. This type of exception condition is called an *unhandled exception condition*. Unhandled exception conditions are examined at the next visible scope for handling. If an exception condition remains unhandled at the outermost <externally-invoked procedure> or <direct SQL statement>, it is seen by the SQL-client. Even if the SQL-client resolves the exception condition, execution is not resumed in the SQL-server where the exception condition was raised.

If there is no most appropriate handler and the condition is a completion condition, then execution is resumed as specified in [Subclause 6.3.3.7, “Exceptions”](#), in ISO/IEC 9075-1. This type of completion condition is called an *unhandled completion condition*.

A handler type specifies CONTINUE, EXIT, or UNDO.

If a handler type specifies CONTINUE, then, when the handler is activated, it will perform the following actions.

- Push the diagnostics area stack.

- Execute the handler action.
- Pop the diagnostics area stack.
- Cause the SQL-session to continue as it would have done if execution of the innermost executing statement that raised the condition had completed.

If a handler type specifies EXIT, then, when the handler is activated, it will perform the following actions.

- Push the diagnostics area stack.
- Execute the handler action.
- Pop the diagnostics area stack.
- Implicitly LEAVE the <compound statement> for which the handler was created, with no active exception condition.

If a handler type specifies UNDO, then, when the handler is activated, it will perform the following actions.

- Push the diagnostics area stack.
- Roll back all of the changes to SQL-data or to schemas by the execution of every SQL-statement contained in the SQL-statement list of the <compound statement> at the scope of the handler and cancel all <SQL procedure statement>s triggered by the execution of such statements.
- Execute the handler action.
- Pop the diagnostics area stack.
- Cause the SQL-session to continue as it would have done if execution of the <compound statement> for which the handler was created had completed.

If a <handler action> completes with a completion condition: *successful completion (00000)*, then it was able to resolve the condition, and execution resumes as specified in [Subclause 15.2, “<handler declaration>”](#).

If a <handler action> completes with an exception or completion condition other than *successful completion (00000)*, then an implicit <resignal statement> is executed. The <resignal statement> determines whether there is another <handler declaration> that can resolve the condition.

4.10 Cursors

This Subclause modifies [Subclause 4.40, “Cursors”](#), in ISO/IEC 9075-2.

4.10.1 General description of cursors

This Subclause modifies [Subclause 4.40.1, “General description of cursors”](#), in ISO/IEC 9075-2.

Insert after the [2nd paragraph](#): A cursor specified by a <declare cursor> contained in an <SQL-client module definition> without an intervening <SQL schema statement> is a *standing SQL-client cursor*. A cursor specified by a <declare cursor> contained in a <compound statement> is a *standing SQL-server cursor*.

Augment the [3rd paragraph](#) by adding “contained in the <local cursor declaration list> of a <compound statement>.” as another option for the containment of the cursor.

Augment the [5th paragraph](#), the [2nd list item](#), the [1st list item](#) by adding “or an indication of a <compound statement>.” immediately after “then an indication of an SQL-client module”.

Insert after the 10th paragraph: For every <declare cursor> in a <compound statement>, a standing SQL-server cursor is effectively created each time the <compound statement> is executed and destroyed when that execution completes.

NOTE 1 — Destroying an open with-return cursor does not simultaneously destroy that cursor's result set.

4.11 SQL-statements

This Subclause modifies Subclause 4.41, "SQL-statements", in ISO/IEC 9075-2.

4.11.1 Classes of SQL-statements

This Subclause modifies Subclause 4.41.1, "Classes of SQL-statements", in ISO/IEC 9075-2.

Insert into the 5th paragraph, after the last list item:

- SQL-control declarations

4.11.2 SQL-statements classified by function

This Subclause modifies Subclause 4.41.2, "SQL-statements classified by function", in ISO/IEC 9075-2.

4.11.2.1 SQL-schema statements

This Subclause modifies Subclause 4.41.2.1, "SQL-schema statements", in ISO/IEC 9075-2.

Insert into the 1st paragraph, after the last list item:

- <SQL-server module definition>
- <drop module statement>

4.11.2.2 SQL-control statements

This Subclause modifies Subclause 4.41.2.6, "SQL-control statements", in ISO/IEC 9075-2.

Insert into the 1st paragraph, after the last list item:

- <compound statement>
- <case statement>
- <if statement>
- <iterate statement>
- <leave statement>
- <loop statement>
- <while statement>
- <repeat statement>
- <for statement>
- <assignment statement>

4.11.2.3 SQL-control declarations

The following are the SQL-control declarations:

- <condition declaration>

4.11 SQL-statements

- <handler declaration>
- <SQL variable declaration>

4.11.2.4 SQL-diagnostics statements

This Subclause modifies Subclause 4.41.2.8, “SQL-diagnostics statements”, in ISO/IEC 9075-2.

Insert into the [1st paragraph](#), after the last list item:

- <signal statement>
- <resignal statement>

4.11.3 SQL-statements and transaction states

This Subclause modifies Subclause 4.41.4, “SQL-statements and transaction states”, in ISO/IEC 9075-2.

Insert into the [1st paragraph](#), after the last list item:

- <for statement>

Insert into the [4th paragraph](#), after the last list item:

- <iterate statement>
- <leave statement>

Insert into the [5th paragraph](#), after the last list item:

- SQL-control statements other than:
 - <for statement>
 - <iterate statement>
 - <leave statement>

4.11.4 SQL-statement atomicity and statement execution contexts

This Subclause modifies Subclause 4.41.5, “SQL-statement atomicity and statement execution contexts”, in ISO/IEC 9075-2.

Insert into the [2nd paragraph](#), after the last list item:

- <assignment statement>.
- <case statement>.
- <compound statement>, unless BEGIN ATOMIC is specified.
- <for statement>.
- <if statement>.
- <loop statement>.
- <repeat statement>.
- <while statement>.

4.11.5 Embeddable SQL-statements

This Subclause modifies Subclause 4.41.6, “Embeddable SQL-statements”, in ISO/IEC 9075-2.

Insert into the [1st paragraph](#), in the [8th list item](#), after the last list item:

- All other SQL-control statements

NOTE 2 — SQL-control declarations contained in (for example) <compound statement>s are permitted, even when the containing SQL-statement is embedded in an <embedded SQL host program>.

4.11.6 Preparable and immediately executable SQL-statements

This Subclause modifies Subclause 4.41.7, “Preparable and immediately executable SQL-statements”, in ISO/IEC 9075-2.

Insert into the [2nd paragraph](#), in the [6th list item](#), after the last list item:

- <compound statement>
- <case statement>
- <if statement>
- <iterate statement>
- <leave statement>
- <loop statement>
- <while statement>
- <repeat statement>
- <for statement>
- <assignment statement>

Insert into the [2nd paragraph](#), after the last list item: Consequently, the following SQL-control declarations are not preparable:

- <condition declaration>
- <handler declaration>
- <SQL variable declaration>

4.11.7 Directly executable SQL-statements

This Subclause modifies Subclause 4.41.8, “Directly executable SQL-statements”, in ISO/IEC 9075-2.

Insert into the [1st paragraph](#), in the [6th list item](#), after the last list item:

- All other SQL-control statements

4.11.8 Iterated SQL-statements

The following are the iterated SQL-statements:

- <loop statement>
- <while statement>
- <repeat statement>
- <for statement>

4.11.9 Compound statements

A compound statement allows a sequence of SQL-statements to be considered as a single SQL-statement. A compound statement also defines a local scope in which SQL-variables, condition handlers, and standing SQL-server cursors can be declared. See [Subclause 15.1, “<compound statement>”](#).

4.12 Basic security model

This Subclause modifies [Subclause 4.42, “Basic security model”](#), in ISO/IEC 9075-2.

4.12.1 Privileges

This Subclause modifies [Subclause 4.42.2, “Privileges”](#), in ISO/IEC 9075-2.

Insert into the [1st paragraph](#), after the last list item:

— SQL-server module

Insert after the [17th paragraph](#): An execute privilege descriptor may also identify the existence of a privilege on the SQL-server module identified by the privilege descriptor.

Insert after the [17th paragraph](#): The object identification included in an execute privilege descriptor may also identify the SQL-server module described by the descriptor.

Insert after the [17th paragraph](#): If the object identified by an execute privilege descriptor is an SQL-invoked routine *R*, then *R* shall be a schema-level routine.

NOTE 3 — “schema-level routine” is defined in [Subclause 11.60, “<SQL-invoked routine>”](#), in ISO/IEC 9075-2.

4.13 SQL-sessions

This Subclause modifies [Subclause 4.45, “SQL-sessions”](#), in ISO/IEC 9075-2.

4.13.1 General description of SQL-sessions

This Subclause modifies [Subclause 4.45.1, “General description of SQL-sessions”](#), in ISO/IEC 9075-2.

Insert after the [4th paragraph](#): Certain operations during an SQL-session *SS* are possible only when *SS* is in *condition handling mode*. This mode becomes in effect when execution of an SQL-statement has completed to the extent that all diagnostics information pertaining to that execution is recorded in the first diagnostics area. Condition handling mode ceases to be in effect when execution of the next SQL-statement begins.

5 Lexical elements

This Clause modifies Clause 5, “Lexical elements”, in ISO/IEC 9075-2.

5.1 <token> and <separator>

This Subclause modifies Subclause 5.2, “<token> and <separator>”, in ISO/IEC 9075-2.

Function

Specify lexical units (tokens and separators) that participate in SQL language.

Format

```
<reserved word> ::=
    !! All alternatives from ISO/IEC 9075-2

    | DO
    | ELSEIF
    | HANDLER
    | IF | ITERATE
    | LEAVE | LOOP
    | REPEAT | RESIGNAL
    | SIGNAL
    | UNTIL
    | WHILE

<non-reserved word> ::=
    !! All alternatives from ISO/IEC 9075-2

    | CONDITION_IDENTIFIER
    | EXIT
    | STACKED
    | UNDO
```

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.

5.2 Names and identifiers

This Subclause modifies Subclause 5.4, “Names and identifiers”, in ISO/IEC 9075-2.

Function

Specify names.

Format

<SQL-server module name> ::=
 <schema qualified name>

<SQL variable name> ::=
 <identifier>

<condition name> ::=
 <identifier>

Syntax Rules

- 1) [Augment SR 4\)a](#)] by adding “<SQL-server module definition> ” as an additional not allowed intervening definition.
- 2) [Insert before SR 4\)b](#)]: If the <local or schema qualified name> is contained in an <SQL-server module definition> without an intervening <schema definition>, then the <default schema name> that is specified or implicit in the <SQL-server module definition> is implicit.
- 3) [Augment SR 5\)a](#)] by adding “Alternatively *TN* shall be contained in an <SQL-server module definition> that contains a <temporary table declaration> *TT* whose <table name> has a <qualified identifier> equivalent to *QI*. ” at the end of the Rule.
- 4) [Augment SR 7](#)] by adding “Alternatively, *CN* shall be contained, without an intervening <SQL schema statement>, in a <compound statement> whose <local cursor declaration list> contains a <declare cursor> whose <cursor name> is *CN*. ” at the end of the Rule.
- 5) [Augment SR 8\)a\)ii\)1\)A](#)] by adding “<SQL-server module definition> ” as an additional not allowed intervening definition for *UDTN*. .
- 6) [Insert after SR 8\)a\)ii\)1\)A](#)]: If *UDTN* is contained in an <SQL-server module definition> without an intervening <schema definition>, then the implicit <schema name> of *UDTN* is the <schema name> that is specified or implicit in <SQL-server module definition>.
- 7) [Augment SR 8\)a\)ii\)1\)B](#)] by adding “is not contained in an <SQL-client module definition> ” as an additional predicate for *UDTN*.
- 8) [Augment SR 8\)b\)i](#)] by adding “<SQL-server module definition> ” as an additional not allowed intervening definition for *UDTN*. .
- 9) [Insert after SR 8\)b\)i](#)]: If *UDTN* is contained in an <SQL-server module definition> without an intervening <schema definition>, then the implicit <schema name> of *UDTN* is the <schema name> that is specified or implicit in <SQL-server module definition>.
- 10) [Augment SR 8\)b\)ii](#)] by adding “is not contained in an <SQL-client module definition> ” as an additional predicate for *UDTN*.
- 11) [Augment SR 13\)d\)i](#)] by adding “<SQL-server module definition> ” as an additional not allowed intervening definition for *SQN*.

- 12) Insert before [SR 13](#)d)ii): If *SQLN* is contained in an <SQL-server module definition> without an intervening <schema definition>, then the <default schema name> that is specified or implicit in the <SQL-server module definition> is implicit.
- 13) Insert after [SR 28](#)): If a <descriptor name> contains an <SQL variable reference> and does not specify a <scope option>, then that <SQL variable reference> is used to supply the value for the <descriptor name>.

NOTE 4 — The previous rule disambiguates between an <SQL variable reference> and a <non-extended descriptor name> that is an <identifier> and gives precedence to the <SQL variable reference>.

Access Rules

No additional Access Rules.

General Rules

- 1) Insert after [the last GR](#): An <SQL-server module name> identifies an SQL-server module.
- 2) Insert after [the last GR](#): An <SQL variable name> identifies an SQL variable.
- 3) Insert after [the last GR](#): A <condition name> identifies an exception condition or a completion condition and optionally a corresponding SQLSTATE value.

Conformance Rules

No additional Conformance Rules.

6 Scalar expressions

This Clause modifies [Clause 6](#), “Scalar expressions”, in ISO/IEC 9075-2.

6.1 <value specification> and <target specification>

This Subclause modifies [Subclause 6.4](#), “<value specification> and <target specification>”, in ISO/IEC 9075-2.

Function

Specify one or more values, host parameters, SQL parameters, dynamic parameters, host variables, or SQL variables.

Format

```
<general value specification> ::=
    !! All alternatives from ISO/IEC 9075-2
    | <SQL variable reference>

<simple value specification> ::=
    !! All alternatives from ISO/IEC 9075-2
    | <SQL variable reference>

<target specification> ::=
    !! All alternatives from ISO/IEC 9075-2
    | <SQL variable reference>

<simple target specification> ::=
    !! All alternatives from ISO/IEC 9075-2
    | <SQL variable reference>

<target array reference> ::=
    !! All alternatives from ISO/IEC 9075-2
    | <SQL variable reference>
```

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

- 1) [Augment GR 3](#)] by adding “SQL variable” to the list of targets.
- 2) [Augment GR 3](#)] by adding “<SQL variable reference>” to the list of <target specification>s.
- 3) [Augment GR 15](#)] by adding “SQL variable” to the list of targets.

- 4) Augment GR 15 by adding “<SQL variable reference>” to the list of <simple target specification>s.

Conformance Rules

No additional Conformance Rules.

6.2 <identifier chain>

This Subclause modifies Subclause 6.6, “<identifier chain>”, in ISO/IEC 9075-2.

Function

Disambiguate a period-separated chain of identifiers.

Format

No additional Format items.

Syntax Rules

- 1) **Insert after SR 7):** An SQL variable V is said to be *refinable* if the declared type of V is a row type, a JSON type, or a structured type.
- 2) **Augment SR 8)** by adding “an SQL variable SV ” to the list of possible referents of a basis.
- 3) **Insert after SR 9)a)ii)1)C):** The scope of one or more <beginning label>s whose associated <local declaration list> includes an SQL variable whose <SQL variable name> is equivalent to I_1 .
- 4) **Insert after SR 9)a)ii)2)B):** If $IPST$ is a <beginning label>, then let SV be the SQL variable whose <SQL variable name> is equivalent to I_1 . PIC_1 is the basis of IC , the basis length is 1 (one), the basis scope is the scope of SV , and the basis referent is SV .
- 5) **Insert after SR 9)b)iii):** If IC is contained in the scope of a <beginning label> whose associated <local declaration list> includes an SQL variable SV whose <SQL variable name> is equivalent to I_1 , then PIC_1 is a candidate basis of IC , the scope of PIC_1 is the scope of SV , and the referent of PIC_1 is SV .
- 6) **Insert after SR 9)b)iii):** If $N = 2$ and PIC_1 is equivalent to a <beginning label> BL whose scope contains IC and whose associated <local declaration list> includes an SQL variable SV whose <SQL variable name> is equivalent to I_2 , then PIC_2 is a candidate basis of IC , the scope of PIC_2 is the scope of SV , and the referent of PIC_2 is SV .
- 7) **Insert after SR 9)b)iii):** If $N > 2$ and PIC_1 is equivalent to a <beginning label> BL whose scope contains IC and whose associated <local declaration list> includes a refinable SQL variable SV whose <SQL variable name> is equivalent to I_2 , then PIC_2 is a candidate basis of IC , the scope of PIC_2 is the scope of SV , and the referent of PIC_2 is SV .
- 8) **Augment NOTE 163** by adding “an <SQL variable reference> ” as an alternative for an <SQL parameter reference>.
- 9) **Augment NOTE 163** by adding “Alternatively, on the left-hand side of an <assignment statement>, (PIC_{BL}) is interpreted as “<left paren> <target specification> <right paren>”, and the identifiers $I_{BL+1}, \dots, I_N$ are parsed using the Format and Syntax Rules of <modified field reference> and <mutator reference> in Subclause 15.5, “<assignment statement>”. ” at the end of the note.
- 10) **Insert after SR 16):** A <basic identifier chain> whose basis referent is an SQL variable is an SQL variable reference.

Access Rules

No additional Access Rules.

General Rules

- 1) Insert after [GR 4](#): If *BIC* is an SQL variable reference, then *BIC* references the SQL variable *SV* of a given execution of the <compound statement> whose <local declaration list> contains the <SQL variable declaration> that declares *SV*.

Conformance Rules

- 1) Insert after [the last CR](#): Without Feature P005, “Qualified SQL variable references”, conforming SQL language shall not contain an SQL variable reference whose first <identifier> is the <beginning label> of a <compound statement>.

6.3 <next value expression>

This Subclause modifies Subclause 6.14, “<next value expression>”, in ISO/IEC 9075-2.

Function

Return the next value of a sequence generator.

Format

No additional Format items.

Syntax Rules

- 1) Insert after SR 1)e): An <assignment statement>.
- 2) Insert after the last SR: <next value expression> shall not be contained without an intervening <SQL statement list> in a <case statement>, an <if statement>, a <loop statement>, a <while statement>, or a <repeat statement>.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.

6.4 <SQL variable reference>

Function

Reference an SQL variable.

Format

<SQL variable reference> ::=
 <basic identifier chain>

Syntax Rules

- 1) An <SQL variable reference> shall be a <basic identifier chain> that is an SQL variable reference.

Access Rules

None.

General Rules

None.

Conformance Rules

None.

7 Query expressions

This Clause modifies Clause 7, “Query expressions”, in ISO/IEC 9075-2.

7.1 <table reference>

This Subclause modifies Subclause 7.6, “<table reference>”, in ISO/IEC 9075-2.

Function

Reference a table.

Format

No additional Format items.

Syntax Rules

- 1) Augment SR 9)d)iii) by adding “<SQL variable reference>” to the list of BNF non-terminals.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.

7.2 <query specification>

This Subclause modifies Subclause 7.16, “<query specification>”, in ISO/IEC 9075-2.

Function

Specify a table derived from the result of a <table expression>.

Format

No additional Format items.

Syntax Rules

- 1) Insert after SR 10)f)i): If IC is contained in the scope of a <beginning label> whose associated <local declaration list> includes an SQL variable SV whose <SQL variable name> is equivalent to I_1 , then PIC_1 is a candidate basis of IC and the scope of PIC_1 is the scope of SV .
- 2) Insert after SR 10)f)i): If $N = 2$ and PIC_1 is equivalent to a <beginning label> BL whose scope contains IC and whose associated <local declaration list> includes an SQL variable SV whose <SQL variable name> is equivalent to I_2 , then PIC_2 is a candidate basis of IC , the scope of PIC_2 is the scope of SV , and the referent of PIC_2 is SV .
- 3) Insert after SR 10)f)i): If $N > 2$ and PIC_1 is equivalent to a <beginning label> BL whose scope contains IC and whose associated <local declaration list> includes a refinable SQL variable SV whose <SQL variable name> is equivalent to I_2 , then PIC_2 is a candidate basis of IC , the scope of PIC_2 is the scope of SV , and the referent of PIC_2 is SV .
- 4) Insert after SR 17)c): If the i -th <derived column> in the <select list> does not specify an <as clause> and the <value expression> of that <derived column> is a single SQL variable reference, then the <column name> of the i -th column of the result is the <SQL variable name> of the SQL variable designated by the SQL variable reference.
- 5) Insert after SR 19)b)iv): An SQL variable.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

- 1) Insert after the last CR: Without Feature P005, “Qualified SQL variable references”, conforming SQL language shall not contain an <asterisked identifier chain> whose first <identifier> is the <beginning label> of a <compound statement>.

8 Additional common rules

This Clause modifies Clause 9, “Additional common rules”, in ISO/IEC 9075-2.

8.1 Potential sources of non-determinism

This Subclause modifies Subclause 9.16, “Potential sources of non-determinism”, in ISO/IEC 9075-2.

Function

Define the potential sources of non-determinism.

Syntax Rules

- 1) Insert after SR 1)as): A <compound statement> that contains an <SQL variable declaration> that specifies a <default option> that contains CURRENT_USER, CURRENT_ROLE, SESSION_USER, SYSTEM_USER, CURRENT_CATALOG, CURRENT_SCHEMA, or CURRENT_PATH.
- 2) Insert after SR 1)as): A <compound statement> that contains an <SQL variable declaration> that specifies a <domain name> and the domain descriptor identified by the <domain name> has a default value that contains a <datetime value function>, CURRENT_USER, CURRENT_ROLE, SESSION_USER, SYSTEM_USER, CURRENT_CATALOG, CURRENT_SCHEMA, or CURRENT_PATH.

Access Rules

None.

General Rules

None.

Conformance Rules

None.

8.2 Executing an <SQL procedure statement>

This Subclause modifies [Subclause 9.17](#), “Executing an <SQL procedure statement>”, in ISO/IEC 9075-2.

Function

Execute an <SQL procedure statement>.

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

NOTE 5 — The General Rules of this Subclause as defined in [GR 2\)](#) of [Subclause 9.17](#), “Executing an <SQL procedure statement>”, in ISO/IEC 9075-2 are not always terminated when an exception condition is raised.

- 1) [Augment GR 7\)](#) by adding “<compound statement>” as an additional restriction for what *S* should not be.
- 2) [Insert before GR 23\)](#): Condition handling mode becomes in effect in the SQL-session.
- 3) [Insert before GR 23\)](#): If *S* did not execute successfully, then:
 - a) Let *SST* be the value of the SQLSTATE associated with the unsuccessful execution of *S*.
 - b) If a handler *MAH* is the most appropriate handler for *SST*, then the General Rules of [Subclause 8.6](#), “Handler execution”, are applied with *MAH* as *MOST APPROPRIATE HANDLER*.
 - c) If no handler is activated and *SST* identifies an SQLSTATE value associated with an exception condition *EC*, then this is an *unhandled exception condition* and the <SQL procedure statement> that resulted in execution of *S* is terminated with the exception condition *EC*.

NOTE 6 — The General Rules of [Subclause 15.2](#), “<handler declaration>”, determine which handler will be invoked for the current condition.

- 4) [Insert after GR 25\)](#): Condition handling mode ceases to be in effect in the SQL-session.

Conformance Rules

No additional Conformance Rules.

8.3 Executing a <compound statement>

Function

Specify the rules for executing a statement that groups other statements together.

Subclause Signature

"Executing a <compound statement>" [General Rules] (
Parameter: "COMPOUND STATEMENT"
)

COMPOUND STATEMENT — a <compound statement> to be executed.

Syntax Rules

None.

Access Rules

None.

General Rules

- 1) Let *CS* be the *COMPOUND STATEMENT* in an application of the General Rules of this Subclause.
- 2) Except where explicitly specified, the General Rules of this Subclause are not terminated if an exception condition is raised.
- 3) If *CS* specifies *ATOMIC*, then a new savepoint level is established.
- 4) The SQL variables, standing SQL-server cursors, and handlers specified in the <local declaration list>, <local cursor declaration list>, and the <local handler declaration list> of *CS* are created in an implementation-dependent (US009) order.
- 5) Let *N* be the number of <SQL procedure statement>s contained in the <SQL statement list> that is immediately contained in *CS* without an intervening <SQL control statement>. For *i* ranging from 1 (one) to *N*:
 - a) Let *S_i* be the *i*-th such <SQL procedure statement>.
 - b) The General Rules of Subclause 9.17, "Executing an <SQL procedure statement>", in ISO/IEC 9075-2, are applied with *S_i* as *EXECUTING STATEMENT*.
 - c) If the execution of *S_i* terminates with exception conditions or completion conditions other than *successful completion (00000)*, then:
 - i) The following <resignal statement> is effectively executed:

RESIGNAL
 - ii) If there are unhandled exception conditions at the completion of the execution of a handler (if any), then the execution of GR 5) is terminated immediately and evaluation continues with GR 6).

8.3 Executing a <compound statement>

- 6) For every open cursor *CR* that is declared in the <local cursor declaration list> of *CS*, the General Rules of Subclause 15.4, “Effect of closing a cursor”, in ISO/IEC 9075-2, are applied with *CR* as *CURSOR* and SAVE as *DISPOSITION*.
- 7) The SQL variables, standing SQL-server cursors, and handlers specified in <local declaration list>, the <local cursor declaration list>, and the <local handler declaration list> of *CS* are destroyed.
- 8) If *CS* specifies ATOMIC, then the current savepoint level is destroyed.
NOTE 7 — Destroying a savepoint level destroys all existing savepoints that are established at that level.
- 9) The <condition name> of every <condition declaration> contained in <local declaration list> ceases to be considered to be defined.
- 10) Evaluation of the General Rules is terminated and control is returned to the invoking Subclause.

Conformance

None.

8.4 Invoking an SQL-invoked routine

This Subclause modifies *Subclause 9.18, “Invoking an SQL-invoked routine”*, in ISO/IEC 9075-2.

Function

Execute the invocation of an SQL-invoked routine.

Syntax Rules

- 1) Convert SR 6) to be: Case:
 - a) If *RI* is contained in an <SQL-server module definition> *M*,
Case:
 - a) If *RI* is contained in an <SQL schema statement>, then the applicable privileges for the <authorization identifier> that owns the containing schema include EXECUTE on *M*.
 - b) Otherwise, the current privileges include EXECUTE on *M*.
 - b) Otherwise, the original SR 6)
- 2) Convert SR 7)c) to be: Case:
 - a) If *RI* is contained in an <SQL-server module definition>, then let *DP* be the SQL-path of that <SQL-server module definition>.
 - b) Otherwise, the original SR 7)c)
- 3) Augment SR 9)h)iii)5) by adding “<SQL variable reference>” to the list of BNF non-terminals.

Access Rules

No additional Access Rules.

General Rules

NOTE 8 — The General Rules of this Subclause as defined in GR 2) of Subclause 9.18, “Invoking an SQL-invoked routine”, in ISO/IEC 9075-2 are not always terminated when an exception condition is raised.

- 1) Insert after GR 6)f)i)4): The identities of declared local temporary tables that are defined in <SQL-server module definition>s.
- 2) Insert after GR 9)a): If *R* is included in an SQL-server module *S*, then the method and time of binding of *P* to *S* is implementation-defined (IW001).
- 3) Augment GR 10)b)ii) by adding “<SQL variable reference>” to the list of BNF non-terminals.

Conformance Rules

No additional Conformance Rules.

8.5 Transformation of query specifications

This Subclause modifies Subclause 9.20, “Transformation of query specifications”, in ISO/IEC 9075-2.

Function

Transform <query specification>s to a normalized form before further processing.

Syntax Rules

- 1) Insert after [SR 16\)b\)](#): If $DCOL_k$ is a single SQL variable reference, then $COLN_k$ is the <SQL variable name> of the SQL variable designated by the SQL variable reference.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.

8.6 Handler execution

Function

Execute the most appropriate handler for an exception condition.

Subclause Signature

"Handler execution" [General Rules] (
Parameter: "MOST APPROPRIATE HANDLER"
)

MOST APPROPRIATE HANDLER — a condition handler that is the most appropriate handler for a particular condition.

Syntax Rules

None.

Access Rules

None.

General Rules

- 1) Let *MAH* be the *MOST APPROPRIATE HANDLER* in an application of the General Rules of this Subclause.
- 2) Except where explicitly specified, the General Rules of this Subclause are not terminated if an exception condition is raised.
- 3) Let *HD* be the <handler declaration> that effectively created *MAH*.
- 4) Let *CS* be the <compound statement> simply containing *HD*.
- 5) Let *HT* be the <handler type> specified in *MAH*.
- 6) Let *HA* be the <handler action> simply contained in *MAH*. If *HA* is not a <compound statement>, then *HA* is effectively replaced by:

BEGIN *HA* ; END
- 7) *MAH* is effectively removed from the list of potential most appropriate handlers.

NOTE 9 — This prevents the handler currently being executed from being chosen as the most appropriate handler for an exception that is raised during the execution of this handler.
- 8) If *MAH* is activated in an atomic execution context and the condition raised is a *transaction rollback (40000)* with any subcondition, then the following <resignal statement> is effectively executed:

RESIGNAL

NOTE 10 — If a condition results in an implicit rollback (See Subclause 4.43.5, "Implicit rollbacks", in ISO/IEC 9075-2) in an atomic execution context, the transaction has been effectively rolled back by the time the Handler body is executed. If a transaction initiating statement is executed following this, it would require a transaction to be initiated in an atomic execution context, which is not valid. Therefore, the condition is effectively signaled to an outer non-atomic execution context.

- 9) If *HT* specifies CONTINUE, then:
 - a) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - b) The General Rules of Subclause 8.3, “Executing a <compound statement>”, are applied with *HA* as *COMPOUND STATEMENT*.
 - c) Case:
 - i) If there is an unhandled condition other than *successful completion (00000)* at the completion of *HA*, then:
 - 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) The following <resignal statement> is effectively executed:

RESIGNAL
 - ii) Otherwise:
 - 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with POP as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) *MAH* is effectively added back to the list of potential most appropriate handlers.
 - 3) *HA* completes with completion condition *successful completion (00000)* and the SQL-session continues as it would have done if execution of the innermost executing statement that raised the condition had completed.
- 10) If *HT* specifies EXIT, then:
 - a) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - b) The General Rules of Subclause 8.3, “Executing a <compound statement>”, are applied with *HA* as *COMPOUND STATEMENT*.
 - c) For every open standing SQL-server cursor *CR* declared in the <local cursor declaration list> of *CS*, the General Rules of Subclause 15.4, “Effect of closing a cursor”, in ISO/IEC 9075-2, are applied with *CR* as *CURSOR* and SAVE as *DISPOSITION*.
 - d) Case:
 - i) If there is an unhandled condition other than *successful completion (00000)* at the completion of *HA*, then:
 - 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) The following <resignal statement> is effectively executed:

RESIGNAL
 - ii) Otherwise:

- 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with POP as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) *MAH* is effectively added back to the list of potential most appropriate handlers.
 - 3) *HA* completes with completion condition *successful completion (00000)* and the SQL-session continues as it would have done if execution of *CS* had completed.
- 11) If *HT* specifies UNDO, then:
- a) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - b) All changes made to SQL-data or schemas by the execution of SQL-statements contained in the <SQL statement list> of *CS* and all <SQL procedure statement>s triggered by the execution of such statements are canceled.
 - c) For every open standing SQL-server cursor *CR* declared in the <local cursor declaration list> of *CS*, the General Rules of Subclause 15.4, “Effect of closing a cursor”, in ISO/IEC 9075-2, are applied with *CR* as *CURSOR* and SAVE as *DISPOSITION*.
 - d) The General Rules of Subclause 8.3, “Executing a <compound statement>”, are applied with *HA* as *COMPOUND STATEMENT*.
 - e) Case:
 - i) If there is an unhandled condition other than *successful completion (00000)* at the completion of *HA*, then:
 - 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with PUSH as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) The following <resignal statement> is effectively executed:

RESIGNAL
 - ii) Otherwise:
 - 1) The General Rules of Subclause 23.2, “Pushing and popping the diagnostics area stack”, in ISO/IEC 9075-2, are applied with POP as *OPERATION* and the diagnostics area stack as *STACK*.
 - 2) *MAH* is effectively added back to the list of potential most appropriate handlers.
 - 3) *HA* completes with completion condition *successful completion (00000)* and the SQL-session continues as it would have done if execution of *CS* had completed.
- 12) Evaluation of the General Rules is terminated and control is returned to the invoking Subclause.

Conformance Rules

None.

9 Additional common elements

This Clause modifies *Clause 10, “Additional common elements”*, in ISO/IEC 9075-2.

9.1 <sqlstate value>

Function

Specify an SQLSTATE value.

Format

```
<sqlstate value> ::=
  SQLSTATE [ VALUE ] <character string literal>
```

Syntax Rules

- 1) Let *L* be the <character string literal> contained in <sqlstate value>.
- 2) The implicit or explicit character set of *L* shall be the implementation-defined (IC019) character set in which SQLSTATE parameter values are returned.
- 3) Let *V* be the character string that is the value of
 TRIM (BOTH ' ' FROM *L*)
- 4) *V* shall comprise either:
 - a) Five characters of which the first two have the form of a standard-defined class code and the last three have the form of a standard-defined subclass code.
 - b) Five characters of which the first two have the form of a standard-defined class code and the last three have the form of an implementation-defined (IC017) subclass code.
 - c) Five characters of which the first two have the form of an implementation-defined (IC016) class code and the last three have the form of either a standard-defined subclass code or an implementation-defined (IC017) subclass code.
- 5) *V* shall not be the SQLSTATE value for the condition *successful completion* (00000).
- 6) The SQLSTATE value defined by the <sqlstate value> is *V*.

Access Rules

None.

General Rules

None.

Conformance Rules

None.

10 Schema definition and manipulation

This Clause modifies [Clause 11](#), “Schema definition and manipulation”, in ISO/IEC 9075-2.

10.1 <schema definition>

This Subclause modifies [Subclause 11.1](#), “<schema definition>”, in ISO/IEC 9075-2.

Function

Define a schema.

Format

```
<schema element> ::=  
    !! All alternatives from ISO/IEC 9075-2  
    | <SQL-server module definition>
```

Syntax Rules

No additional Syntax Rules.

Access Rules

No additional Access Rules.

General Rules

No additional General Rules.

Conformance Rules

No additional Conformance Rules.