

Australian Standard™

**Information technology—Database
languages—SQL**

Part 1: Framework (SQL/Framework)



This Australian Standard was prepared by Committee IT-027, Data Management and Interchange. It was approved on behalf of the Council of Standards Australia on 16 May 2005. This Standard was published on 16 June 2005.

The following are represented on Committee IT-027:

Association of Superannuation Funds of Australia
Australia Post
Australia Bankers Association
Australian Bureau of Statistics
Australian Computer Society
Australian Customs Service
Australian Electoral Commission
Australian Electric and Electronic Manufacturers Association
Australian Industry Group
Australian Information Industry Association
Australian Institute of Health and Welfare
Australian Taxation Office
Centrelink
Data Management Association Australia
Department of Defence
Department of Immigration, Multicultural and Indigenous Affairs
Health Insurance Commission

Keeping Standards up-to-date

Standards are living documents which reflect progress in science, technology and systems. To maintain their currency, all Standards are periodically reviewed, and new editions are published. Between editions, amendments may be issued. Standards may also be withdrawn. It is important that readers assure themselves they are using a current Standard, which should include any amendments which may have been published since the Standard was purchased.

Detailed information about Standards can be found by visiting the Standards Web Shop at www.standards.com.au and looking up the relevant Standard in the on-line catalogue.

Alternatively, the printed Catalogue provides information current at 1 January each year, and the monthly magazine, *The Global Standard*, has a full listing of revisions and amendments published each month.

Australian Standards™ and other products and services developed by Standards Australia are published and distributed under contract by SAI Global, which operates the Standards Web Shop.

We also welcome suggestions for improvement in our Standards, and especially encourage readers to notify us immediately of any apparent inaccuracies or ambiguities. Contact us via email at mail@standards.org.au, or write to the Chief Executive, Standards Australia, GPO Box 5420, Sydney, NSW 2001.

This Standard was issued in draft form for comment as DR 05142.

Australian Standard™

Information technology—Database languages—SQL

Part 1: Framework (SQL/Framework)

Originated as AS/NZS 3968.0:1994.
Revised and redesignated as AS ISO/IEC 9075.1—2005.

COPYRIGHT

© Standards Australia

All rights are reserved. No part of this work may be reproduced or copied in any form or by any means, electronic or mechanical, including photocopying, without the written permission of the publisher.

Published by Standards Australia GPO Box 5420, Sydney, NSW 2001, Australia

ISBN 0 7337 6763 X

PREFACE

This Standard was prepared by the Standards Australia Committee IT-027, Data Management and Interchange to supersede AS/NZS 3968.0:1994, *Information technology—Database languages—SQL, Part 0: Definition of data structures and basic operations*.

This Standard is identical with, and has been reproduced from, ISO/IEC 9075-1:2003, *Information technology—Database languages—SQL—Part 1: Framework (SQL/Framework)*.

The objective of this Standard is to provide database designers, administrators and developers with the conceptual framework used in other parts of the 9075 series of SQL Standards.

This Standard is Part 1 of AS 9075, *Information technology—Database languages—SQL*, which is published in parts as follows:

- Part 1: Framework (SQL/Framework) (this Standard)
- Part 2: Foundation (SQL/Foundation)
- Part 3: Call-Level Interface (SQL/CLI)
- Part 4: Persistent Stored Modules (SQL/PSM)
- Part 5: Host Language Bindings (SQL/Bindings)
- Part 9: Management of External Data (SQL/MED)
- Part 10: Object Language Bindings (SQL/OLB)
- Part 11: Information and Definition Schemas (SQL/Schemata)
- Part 13: SQL Routines and Types Using the Java™ Programming Language (SQL/JRT)
- Part 14: XML-Related Specifications (SQL/XML)

As this Standard is reproduced from an international standard, the following applies:

- (a) Its number appears on the cover and title page while the international standard number appears only on the cover.
- (b) In the source text ‘this part of ISO/IEC 9075’ should read ‘this Australian Standard’.
- (c) A full point substitutes for a comma when referring to a decimal marker.

References to International Standards should be replaced by references to Australian or Australian/New Zealand Standards, as follows:

<i>Reference to International Standard</i>		<i>Australian Standard</i>	
ISO/IEC		AS	
9075	Information technology—Database languages—SQL	9075	Information technology—Database languages—SQL
9075-2	Part 2: Foundation (SQL/Foundation)	9075.2	Part 2: Foundation (SQL/Foundation)
9075-3	Part 3: Call-Level Interface (SQL/CLI)	9075.3	Part 3: Call-Level Interface (SQL/CLI)
9075-4	Part 4: Persistent Stored Modules (SQL/PSM)	9075.4	Part 4: Persistent Stored Modules (SQL/PSM)
9075-9	Part 9: Management of External Data (SQL/MED)	9075.9	Part 9: Management of External Data (SQL/MED)

9075-10	Part 10: Object Language Bindings (SQL/OLB)	9075.10	Part 10: Object Language Bindings (SQL/OLB)
9075-11	Part 11: Information and Definition Schemas (SQL/Schemas)	9075.11	Part 11: Information and Definition Schemas (SQL/Schemas)
9075-13	Part 13: SQL Routines and Types Using the Java™ Programming Language (SQL/JRT)	9075.13	Part 13: SQL Routines and Types Using the Java™ Programming Language (SQL/JRT)
9075-14	Part 14: XML-Related Specifications (SQL/XML)	9075.14	Part 14: XML-Related Specifications (SQL/XML)

Only referenced documents that have adopted as Australian or Australian/New Zealand Standards have been listed.

CONTENTS

Page

1	Scope.....	1
2	Normative references.....	3
2.1	JTC1 standards.....	3
2.2	Other international standards.....	4
3	Definitions and use of terms.....	5
3.1	Definitions.....	5
3.1.1	Definitions provided in this standard.....	5
3.2	Use of terms.....	6
3.3	Informative elements.....	7
4	Concepts.....	9
4.1	Caveat.....	9
4.2	SQL-environments and their components.....	9
4.2.1	SQL-environments.....	9
4.2.2	SQL-agents.....	9
4.2.3	SQL-implementations.....	10
4.2.3.1	SQL-clients.....	10
4.2.3.2	SQL-servers.....	10
4.2.4	SQL-client modules.....	10
4.2.5	User identifiers.....	11
4.2.6	User mapping concepts.....	11
4.2.7	Routine mapping concepts.....	11
4.2.8	Catalogs and schemas.....	12
4.2.8.1	Catalogs.....	12
4.2.8.2	SQL-schemas.....	12
4.2.8.3	The Information Schema.....	12
4.2.8.4	The Definition Schema.....	12
4.2.9	Foreign servers and descriptors.....	13
4.2.10	Foreign-data wrappers and descriptors.....	13
4.2.11	SQL-data.....	13
4.3	Tables.....	13
4.4	SQL data types.....	14
4.4.1	General data type information.....	14
4.4.2	The null value.....	15
4.4.3	Predefined types.....	15

4.4.3.1	Numeric types.	15
4.4.3.2	Character string types.	15
4.4.3.3	Boolean type.	16
4.4.3.4	Datetime types.	16
4.4.3.5	Interval types.	16
4.4.3.6	XML type.	16
4.4.4	Constructed atomic types.	17
4.4.4.1	Reference types.	17
4.4.5	Constructed composite types.	17
4.4.5.1	Collection types.	17
4.4.5.2	Row types.	17
4.4.5.3	Fields.	17
4.5	Sites and operations on sites.	18
4.5.1	Sites.	18
4.5.2	Assignment.	18
4.5.3	Nullability.	18
4.6	SQL-schema objects.	18
4.6.1	General SQL-schema object information.	18
4.6.2	Descriptors relating to character sets.	19
4.6.2.1	Character sets.	19
4.6.2.2	Collations.	20
4.6.2.3	Transliterations.	20
4.6.3	Domains and their components.	20
4.6.3.1	Domains.	20
4.6.3.2	Domain constraints.	20
4.6.4	User-defined types.	21
4.6.4.1	Structured types.	21
4.6.4.2	Attributes.	21
4.6.5	Distinct types.	21
4.6.6	Base tables and their components.	22
4.6.6.1	Base tables.	22
4.6.6.2	Columns.	22
4.6.6.3	Table constraints.	22
4.6.6.4	Triggers.	23
4.6.7	View definitions.	23
4.6.8	Assertions.	23
4.6.9	SQL-server modules (defined in ISO/IEC 9075-4, SQL/PSM).	23
4.6.10	Schema routines.	23
4.6.11	Sequence generators.	23
4.6.12	Privileges.	24
4.6.13	Roles.	24
4.7	Integrity constraints and constraint checking.	24

4.7.1	Constraint checking	24
4.7.2	Determinism and constraints.	25
4.8	Communication between an SQL-agent and an SQL-server.	25
4.8.1	Host languages.	25
4.8.2	Parameter passing and data type correspondences.	26
4.8.2.1	General parameter passing and data type correspondence information.	26
4.8.2.2	Data type correspondences.	26
4.8.2.3	Locators.	26
4.8.2.4	Status parameters.	27
4.8.2.5	Indicator parameters.	27
4.8.3	Descriptor areas.	27
4.8.4	Diagnostic information.	27
4.8.5	SQL-transactions.	28
4.9	Modules.	28
4.10	Routines.	29
4.10.1	General routine information.	29
4.10.2	Type preserving functions.	29
4.11	SQL-statements.	30
4.11.1	Classes of SQL-statements.	30
4.11.2	SQL-statements classified by function.	30
5	The parts of ISO/IEC 9075.	33
5.1	Overview.	33
5.2	ISO/IEC 9075-1: Framework (SQL/Framework).	33
5.3	ISO/IEC 9075-2: Foundation (SQL/Foundation).	34
5.3.1	Data types specified in ISO/IEC 9075-2.	34
5.3.2	Tables.	34
5.3.3	Bindings methods.	34
5.3.3.1	Embedded SQL.	34
5.3.3.2	Dynamic SQL.	35
5.3.3.3	Direct invocation of SQL.	35
5.3.4	SQL-statements specified in ISO/IEC 9075-2.	35
5.4	ISO/IEC 9075-3: Call Level Interface (SQL/CLI).	36
5.5	ISO/IEC 9075-4: Persistent Stored Modules (SQL/PSM).	37
5.5.1	SQL-statements specified in ISO/IEC 9075-4.	37
5.6	ISO/IEC 9075-9: Management of External Data (SQL/MED).	37
5.7	ISO/IEC 9075-10: Object Language Bindings (SQL/OLB).	38
5.8	ISO/IEC 9075-11: Information and Definition Schemas (SQL/Schemata).	38
5.9	ISO/IEC 9075-13: Java Routines and Types Using the Java Programming Language (SQL/JRT).	38
5.10	ISO/IEC 9075-14: XML-Related Specifications (SQL/XML).	38
6	Notation and conventions used in other parts of ISO/IEC 9075.	39
6.1	Notation taken from ISO/IEC 10646.	39
6.2	Notation provided in this International Standard.	39

6.3	Conventions.	41
6.3.1	Specification of syntactic elements.	41
6.3.2	Specification of the Information and Definition Schemata.	41
6.3.3	Use of terms.	42
6.3.3.1	Syntactic containment.	42
6.3.3.2	Terms denoting rule requirements.	43
6.3.3.3	Rule evaluation order.	43
6.3.3.4	Conditional rules.	44
6.3.3.5	Syntactic substitution.	44
6.3.3.6	Other terms.	45
6.3.3.7	Exceptions.	45
6.3.3.8	General Rules not terminated on exception conditions.	46
6.3.4	Descriptors.	46
6.3.5	Relationships of parts within ISO/IEC 9075.	47
6.3.5.1	New and modified Clauses, Subclauses, and Annexes.	48
6.3.5.2	Functions.	49
6.3.5.3	New and modified Format items.	49
6.3.5.4	New and modified paragraphs and rules.	50
6.3.5.5	New and modified tables.	51
6.3.5.6	Modified annexes.	51
6.3.6	Index typography.	51
6.3.7	Feature ID and Feature Name.	52
6.4	Object identifier for Database Language SQL.	52
7	Annexes to the parts of ISO/IEC 9075.	55
7.1	Implementation-defined elements.	55
7.2	Implementation-dependent elements.	55
7.3	Deprecated features.	55
7.4	Incompatibilities with previous versions.	55
8	Conformance.	57
8.1	Minimum conformance.	57
8.2	Conformance to parts.	57
8.3	Conformance to features.	57
8.4	Conformance to SQL packages.	58
8.4.1	Enhanced datetime facilities.	59
8.4.2	Enhanced integrity management.	59
8.4.3	PSM.	60
8.4.4	Basic object support.	60
8.4.5	Enhanced object support.	60
8.4.6	Active database.	61
8.4.7	OLAP.	61
8.4.8	Extensions and options.	61
8.5	SQL flagger.	62

8.6	Claims of conformance.....	63
8.6.1	Requirements for SQL applications.....	63
8.6.2	Requirements for SQL-implementations.....	64
Annex A	Maintenance and interpretation of SQL.....	65
Annex B	Implementation-defined elements.....	67
Annex C	Implementation-dependent elements.....	69
Index.		71

TABLES

Page

1	Relationships between externally-invoked and SQL-invoked routines.	29
2	Symbols used in BNF.	39
3	SQL Packages.	58

INTRODUCTION

The organization of this part of ISO/IEC 9075 is as follows:

- 1) [Clause 1, “Scope”](#), specifies the scope of this part of ISO/IEC 9075.
- 2) [Clause 2, “Normative references”](#), identifies additional standards that, through reference in this part of International Standard, constitute provisions of ISO/IEC 9075.
- 3) [Clause 3, “Definitions and use of terms”](#), defines terms used in this and other parts of ISO/IEC 9075.
- 4) [Clause 4, “Concepts”](#), describes the concepts used in ISO/IEC 9075.
- 5) [Clause 5, “The parts of ISO/IEC 9075”](#), summarises the content of each of the parts of ISO/IEC 9075, in terms of the concepts described in [Clause 4, “Concepts”](#).
- 6) [Clause 6, “Notation and conventions used in other parts of ISO/IEC 9075”](#), defines notation and conventions used in other parts of ISO/IEC 9075.
- 7) [Clause 7, “Annexes to the parts of ISO/IEC 9075”](#), describes the content of annexes of other parts of ISO/IEC 9075.
- 8) [Clause 8, “Conformance”](#), specifies requirements that apply to claims of conformance to all or some of the parts of ISO/IEC 9075.
- 9) [Annex A, “Maintenance and interpretation of SQL”](#), is an informative Annex. It describes the formal procedures for maintenance and interpretation of ISO/IEC 9075.
- 10) [Annex B, “Implementation-defined elements”](#), is an informative Annex. It lists those features for which the body of this part of ISO/IEC 9075 states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or any other behavior is partly or wholly implementation-defined.
- 11) [Annex C, “Implementation-dependent elements”](#), is an informative Annex. It lists those features for which the body of this part of ISO/IEC 9075 states that the syntax, the meaning, the returned results, the effect on SQL-data and/or schemas, or any other behavior is partly or wholly implementation-dependent.

In the text of this part of ISO/IEC 9075, Clauses begin a new odd-numbered page. Any resulting blank space is not significant.

AUSTRALIAN STANDARD

Information technology — Database languages — SQL —

Part 1: Framework (SQL/Framework)

1 Scope

This part of ISO/IEC 9075 describes the conceptual framework used in other parts of ISO/IEC 9075 to specify the grammar of SQL and the result of processing statements in that language by an SQL-implementation.

This part of ISO/IEC 9075 also defines terms and notation used in the other parts of ISO/IEC 9075.

This page intentionally left blank.

2 Normative references

The following referenced documents are indispensable for the application 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.

2.1 JTC1 standards

[ASN.1] ISO/IEC 8824-1:1998, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 8824-1:1998/Cor 1:1999

ISO/IEC 8824-1:1998/Amd 1:2000 *Relative object identifiers*

ISO/IEC 8824-1:1998/Amd 2:2000 *ASN.1 Semantic Model*

[Foundation] ISO/IEC 9075-2:2003, *Information technology — Database languages — SQL — Part 2: Foundation (SQL/Foundation)*

[CLI] ISO/IEC 9075-3:2003, *Information technology — Database languages — SQL — Part 3: Call-Level Interface (SQL/CLI)*

[PSM] ISO/IEC 9075-4:2003, *Information technology — Database languages — SQL — Part 4: Persistent Stored Modules (SQL/PSM)*

[MED] ISO/IEC 9075-9:2003, *Information technology — Database languages — SQL — Part 9: Management of External Data (SQL/MED)*

[OLB] ISO/IEC 9075-10:2003, *Information technology — Database languages — SQL — Part 10: Object Language Bindings (SQL/OLB)*

[Schemata] ISO/IEC 9075-11:2003, *Information technology — Database languages — SQL — Part 11: Information and Definition Schemas (SQL/Schemata)*

[JRT] ISO/IEC 9075-13:2003, *Information technology — Database languages — SQL — Part 13: Java Routines and Types (SQL/JRT)*

[XML] ISO/IEC 9075-14:2003, *Information technology — Database languages — SQL — Part 14: XML-Related Specifications (SQL/XML)*

[UCS] ISO/IEC 10646-1:2000, *Information technology — Universal Multi-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane*

[UCS Supp] ISO/IEC 10646-2:2001, *Information technology — Universal Multi-Octet Coded Character Set (UCS) — Part 2: Supplementary Planes*

[ISO14651] ISO/IEC 14651:2001, *Information technology — International string ordering and comparison — Method for comparing character strings and description of the common template tailorable ordering*

2.2 Other international standards

[Unicode 3.0] The Unicode Consortium, *The Unicode Standard, Version 3.0*, Reading, MA, Addison-Wesley Developers Press, 2000. ISBN 0-201-61633-5.

[Unicode 3.1] The Unicode Consortium, *The Unicode Standard, Version 3.1.0, Unicode Standard Annex #27: Unicode 3.1 (which amends The Unicode Standard, Version 3.0)*. 2001-03-23.
<http://www.unicode.org/unicode/reports/tr27/>

[Unicode10] Davis, Mark and Whistler, Ken. *Unicode Technical Standard #10, Unicode Collation Algorithm, Version 8.0*, 2001-03-23. The Unicode Consortium.
<http://www.unicode.org/unicode/reports/tr10/tr10-8.html>

[Unicode15] Davis, Mark and Dürst, Martin. *Unicode Standard Annex #15, Unicode Normalization Forms, Version 21.0*, 2001-03-23. The Unicode Consortium.
<http://www.unicode.org/unicode/reports/tr15/tr15-21.html>

[Unicode19] Davis, Mark. *Unicode Standard Annex #19, UTF-32, Version 8.0*, 2001-03-23. The Unicode Consortium.
<http://www.unicode.org/unicode/reports/tr19/tr19-8.html>

3 Definitions and use of terms

3.1 Definitions

3.1.1 Definitions provided in this standard

In this part of ISO/IEC 9075, the definition of a verb defines every voice, mood, and tense of that verb.

For the purposes of this and other parts of ISO/IEC 9075, the following definitions apply:

3.1.1.1 atomic: Incapable of being subdivided.

3.1.1.2 compilation unit: A segment of executable code, possibly consisting of one or more subprograms.

3.1.1.3 data type: A set of representable values.

3.1.1.4 descriptor: A coded description of an SQL object. It includes all of the information about the object that a conforming SQL-implementation requires.

3.1.1.5 fully qualified of a name of some SQL object: with all optional components specified explicitly.

NOTE 1 — A fully qualified name does not necessarily identify an object uniquely. For example, although a fully qualified specific name, consisting of a catalog name, a schema name and a specific name, uniquely identifies a routine, a fully qualified routine name doesn't necessarily do so.

3.1.1.6 identifier: A means by which something is identified.

3.1.1.7 identify: To reference something without ambiguity.

3.1.1.8 implementation-defined: Possibly differing between SQL-implementations, but specified by the implementor for each particular SQL-implementation.

3.1.1.9 implementation-dependent: Possibly differing between SQL-implementations, but not specified by ISO/IEC 9075, and not required to be specified by the implementor for any particular SQL-implementation.

3.1.1.10 instance (of a value): A physical representation of a value. Each instance is at exactly one site. An instance has a data type that is the data type of its value.

3.1.1.11 null value: A special value that is used to indicate the absence of any data value.

3.1.1.12 object (as in 'x object'): Any *thing*. An *X* object is a component of, or is otherwise associated with, some *X*, and cannot exist independently of that *X*. For example, an SQL object is an object that exists only in the context of SQL; an SQL-schema object is an object that exists in some SQL-schema.

3.1.1.13 persistent: Continuing to exist indefinitely, until destroyed deliberately. Referential and cascaded actions are regarded as deliberate. Actions incidental to the termination of an SQL-transaction or an SQL-session are not regarded as deliberate.

- 3.1.1.14 property (of an object):** An attribute, quality, or characteristic of the object.
- 3.1.1.15 row:** A sequence of (field name, value) pairs, the data type of each value being specified by the row type.
- 3.1.1.16 scope (of a standard):** The clause in the standard that defines the subject of the standard and the aspects covered, thereby indicating the limits of applicability of the standard or of particular parts of it.
- 3.1.1.17 scope (of a name or declaration):** The one or more BNF non-terminal symbols within which the name or declaration is effective.
- 3.1.1.18 scope (of a local dynamic name):** The module containing the object named.
- 3.1.1.19 scope (of a reference type):** The table that a value of that reference type references.
- 3.1.1.20 sequence:** An ordered collection of objects that are not necessarily distinct.
- 3.1.1.21 site:** A place occupied by an instance of a value of some specified data type (or subtype of that data type).
- 3.1.1.22 sizing item:** sizing item: The value of an implementation-defined item for the SQL-implementation or for a profile.
- 3.1.1.23 SQL-connection:** An association between an SQL-client and an SQL-server.
- 3.1.1.24 SQL-environment:** The context in which SQL-data exists and SQL-statements are executed.
- 3.1.1.25 SQL-implementation:** A processor that processes SQL-statements. A *conforming SQL-implementation* is an SQL-implementation that satisfies the requirements for SQL-implementations as defined in Clause 8, “Conformance”.
- 3.1.1.26 SQL-session:** The context within which a single user, from a single SQL-agent, executes a sequence of consecutive SQL-statements over a single SQL-connection.
- 3.1.1.27 SQL-statement:** A string of characters that conforms to the Format and Syntax Rules specified in the parts of ISO/IEC 9075.
- 3.1.1.28 table:** An unordered collection of rows having an ordered collection of one or more columns. Each column has a name and a data type. Each row has, for each column, exactly one value in the data type of that column.

3.2 Use of terms

The concepts on which ISO/IEC 9075 is based are described in terms of objects, in the usual sense of the word.

An object might be considered to be a component of the object on which it depends. If an objects ceases to exist, then every object dependent on that objects also ceases to exist. The representation of an object is known as a descriptor. The descriptor of an object represents everything that needs to be known about the object. See also Subclause 6.3.4, “Descriptors”.

3.3 Informative elements

In several places in the body of ISO/IEC 9075, informative notes appear. For example:

NOTE 2 — This is an example of a note.

Those notes do not belong to the normative part ISO/IEC 9075. Because claims of conformance to non-normative material is not meaningful, conformance to material specified in those notes shall not be claimed.

This page intentionally left blank.

4 Concepts

4.1 Caveat

This Clause describes concepts that are, for the most part, specified precisely in other parts of ISO/IEC 9075. In any case of discrepancy, the specification in the other part is to be presumed correct.

4.2 SQL-environments and their components

4.2.1 SQL-environments

An SQL-environment comprises:

- One SQL-agent.
- One SQL-implementation.
- Zero or more SQL-client modules, containing externally-invoked procedures available to the SQL-agent.
- Zero or more authorization identifiers.
- Zero or more user mappings.
- Zero or more routine mappings.
- Zero or more catalogs, each of which contains one or more SQL-schemas.
- The sites, principally base tables, that contain SQL-data, as described by the contents of the schemas. This data may be thought of as “the database”, but the term is not used in ISO/IEC 9075, because it has different meanings in the general context.

4.2.2 SQL-agents

An *SQL-agent* is that which causes the execution of SQL-statements. In the case of the direct invocation of SQL (see [Subclause 5.3.3.3, “Direct invocation of SQL”](#)), it is implementation-defined. Alternatively, it may consist of one or more compilation units that, when executed, invoke externally-invoked procedures in an SQL-client module.

4.2.3 SQL-implementations

An *SQL-implementation* is a processor that executes SQL-statements, as required by the SQL-agent. An SQL-implementation, as perceived by the SQL-agent, includes one SQL-client, to which that SQL-agent is bound, and one or more SQL-servers. An SQL-implementation can conform to ISO/IEC 9075 without allowing more than one SQL-server to exist in an SQL-environment.

Because an SQL-implementation can be specified only in terms of how it executes SQL-statements, the concept denotes an installed instance of some software (database management system). ISO/IEC 9075 does not distinguish between features of the SQL-implementation that are determined by the software vendor and those determined by the installer.

ISO/IEC 9075 recognizes that SQL-client and SQL-server software may have been obtained from different vendors; it does not specify the method of communication between SQL-client and SQL-server.

4.2.3.1 SQL-clients

An *SQL-client* is a processor, perceived by the SQL-agent as part of the SQL-implementation, that establishes SQL-connections between itself and SQL-servers and maintains a diagnostics area and other state data relating to interactions between itself, the SQL-agent, and the SQL-servers.

When one or more SQL-connections have been established for an SQL-client, the diagnostics area maintained by that SQL-client is a copy of the first diagnostics area in the diagnostics area stack (see [Subclause 4.8.4, “Diagnostic information”](#)) maintained by the SQL-server of the current SQL-connection. When no SQL-connection exists, the diagnostics area is either empty or contains diagnostics information pertaining to some failed SQL-connection.

4.2.3.2 SQL-servers

Each *SQL-server* is a processor, perceived by the SQL-agent as part of the SQL-implementation, that manages SQL-data.

Each SQL-server:

- Manages the SQL-session taking place over the SQL-connection between itself and the SQL-client.
- Executes SQL-statements received from the SQL-client, receiving and sending data as required.
- Maintains the state of the SQL-session, including the authorization identifier and certain session defaults.

4.2.4 SQL-client modules

An *SQL-client module* is a module that is explicitly created and dropped by implementation-defined mechanisms.

An SQL-client module does not necessarily have a name; if it does, the permitted names are implementation-defined.

An SQL-client module contains zero or more *externally-invoked procedures*. An externally-invoked procedure consists of a single SQL-statement. An externally-invoked procedure is invoked from a compilation unit of a host language.

Exactly one SQL-client module is associated with an SQL-agent at any time. However, in the case of either direct binding style or SQL/CLI, this may be a default SQL-client module whose existence is not apparent to the user.

SQL-session modules are implicitly-created modules for prepared SQL-statements (see Subclause 4.37, “SQL-sessions”, in ISO/IEC 9075-2).

Each <SQL-client module definition> *M* that is associated with an SQL-session, other than an SQL-session module, may have associated with it a *shadow module M1* that has an implementation-dependent name not equivalent to the <module name> of any other <SQL-client module definition> in the same SQL-session and whose <module authorization clause> specifies “SCHEMA *SN*”, where *SN* is the explicit or implicit <schema name> of the <module authorization clause> of *M*. The <language clause>, the SQL-path, if specified, and the <module character set specification> of *M1* are identical to the corresponding characteristic of *M*. When *M* contains a <module authorization clause> that specifies “FOR STATIC ONLY”, this shadow module effectively contains one <externally-invoked procedure> for each SQL-statement prepared by <prepared statement>s or <execute immediate statement>s contained in *M*.

4.2.5 User identifiers

A *user identifier* represents a user. The means of creating and destroying user identifiers, and their mapping to real users, is not specified by ISO/IEC 9075.

4.2.6 User mapping concepts

A user mapping pairs an authorization identifier with a foreign server descriptor.

4.2.7 Routine mapping concepts

A routine mapping pairs an SQL-invoked routine with a foreign server descriptor.

NOTE 3 — In this revision of ISO/IEC 9075, the SQL-invoked routine that is paired with a foreign server descriptor is restricted to be an SQL-invoked regular function. Possible future revisions may drop this restriction.

4.2.8 Catalogs and schemas

4.2.8.1 Catalogs

A *catalog* is a named collection of SQL-schemas, foreign server descriptors, and foreign data wrapper descriptors in an SQL-environment. The mechanisms for creating and destroying catalogs are implementation-defined.

The default catalog name for <preparable statement>s that are dynamically prepared in the current SQL-session through the execution of <prepare statement>s and <execute immediate statement>s is initially implementation-defined but may be changed by the use of <set catalog statement>s.

4.2.8.2 SQL-schemas

An *SQL-schema*, often referred to simply as a *schema*, is a persistent, named collection of descriptors. Any object whose descriptor is in some SQL-schema is known as an *SQL-schema object*.

A schema, the schema objects in it, and the SQL-data described by them are said to be owned by the authorization identifier associated with the schema.

SQL-schemas are created and destroyed by execution of SQL-schema statements (or by implementation-defined mechanisms).

4.2.8.3 The Information Schema

Every catalog contains an SQL-schema with the name INFORMATION_SCHEMA that includes the descriptors of a number of schema objects, mostly view definitions, that together allow every descriptor in that catalog to be accessed, but not changed, as though it was SQL-data.

The data available through the views in an Information Schema includes the descriptors of the Information Schema itself. It does not include the schema objects or base tables of the Definition Schema (see [Subclause 4.2.8.4, “The Definition Schema”](#)).

Each Information Schema view is so specified that a given user can access only those rows of the view that represent descriptors on which that user has privileges.

4.2.8.4 The Definition Schema

The *definition schema* is a fictitious schema with the name DEFINITION_SCHEMA; if it were to exist, the SQL-data in its base tables would describe all the SQL-schemas available to an SQL-server. ISO/IEC 9075 defines it only in order to use it as the basis for the views of the Information Schema.

The structure of the Definition Schema is a representation of the data model of SQL.

4.2.9 Foreign servers and descriptors

A foreign server is a processor that is not part of the SQL-implementation. A foreign server is described by a foreign server descriptor. A foreign server manages data that is not part of the SQL-environment. An SQL-server and an SQL-client can use a foreign server descriptor, which is a catalog element, to communicate with a foreign server. The data managed by a foreign server can be accessed by an SQL-server or an SQL-client through foreign tables, which are SQL-schema elements.

4.2.10 Foreign-data wrappers and descriptors

A foreign-data wrapper provides the mechanism by which an SQL-server can access the data that is managed by a foreign server. A foreign-data wrapper is described by a foreign-data wrapper descriptor.

4.2.11 SQL-data

SQL-data is data described by SQL-schemas — data that is under the control of an SQL-implementation in an SQL-environment.

4.3 Tables

A *table* has an ordered collection of one or more columns and an unordered collection of zero or more rows. Each column has a name and a data type. Each row has, for each column, exactly one value in the data type of that column.

SQL-data consists entirely of table variables, called *base tables*. An operation that references zero or more base tables and returns a table is called a *query*. The result of a query is called a *derived table*.

The rows of a table have a type, called “the row type”; every row of a table has the same row type, which is also the row type of the table. A table that is declared to be based on some structured type is called a “typed table”; its columns correspond in name and declared type to the attributes of the structured type. Typed tables have one additional column, called the “self-referencing column” whose type is a reference type associated with the structured type of the table.

If a typed table *TB1* has an associated structured type *TP1* that is a subtype of some other structured type *TP2*, then *TB1* can be defined to be a “subtable” of a typed table *TB2* whose associated type is *TP2*; *TB2* is, in this case, a “supertable” of *TB1*.

A *view* is a named query, which can be invoked by use of this name. The result of such an invocation is called a *viewed table*.

Some queries, and hence some views, are *updatable*, meaning they can appear as targets of statements that change SQL-data. The results of changes expressed in this way are defined in terms of corresponding changes to base tables.

No two columns of a base table or a viewed table can have the same name. Derived tables, other than viewed tables, may contain more than one column with the same name.

A base table is either a schema object (its descriptor is in a schema; see Subclause 4.6.6, “Base tables and their components”) or a module object (its descriptor is in a module; see Subclause 4.9, “Modules”). A base table whose descriptor is in a schema is called a *created base table*, and may be either persistent or temporary (though its descriptor is persistent in either case). A *persistent base table* contains 0 (zero) or more rows of persistent SQL-data. A base table declared in a module may only be temporary, and is called a *declared temporary table*.

A *temporary table* is an SQL-session object that cannot be accessed from any other SQL-session. A *global temporary table* can be accessed from any associated SQL-client module. A *local temporary table* can be accessed only from the module to which it is local.

A temporary table is empty when an SQL-session is initiated and it is emptied (that is, all its rows are deleted) either when an SQL-transaction is terminated or when an SQL-session is terminated, depending on its descriptor.

4.4 SQL data types

4.4.1 General data type information

Every data value belongs to some data type.

Every data type is either *predefined*, *constructed*, or *user-defined*. Every data type has a name. The name of a predefined or constructed data type is a reserved word specified by that part of ISO/IEC 9075 that specifies the data type. The name of a user-defined type is provided in its definition. A user-defined data type is a schema object; see Subclause 4.6.4, “User-defined types”.

A predefined data type is a data type specified by ISO/IEC 9075, and is therefore provided by the SQL-implementation. A data type is predefined even though the user is required (or allowed) to provide certain parameters when specifying it (for example the precision of a number).

A predefined data type is atomic. An atomic type is a data type whose values are not composed of values of other data types. The existence of an operation (SUBSTRING, EXTRACT) that is capable of selecting part of a string or datetime value does not imply that a string or datetime is not atomic.

A constructed type is either atomic or composite. A composite type is a data type each of whose values is composed of zero or more values, each of a declared data type.

4.4.2 The null value

Every data type includes a special value, called the *null value*, sometimes denoted by the keyword NULL. This value differs from other values in the following respects:

- Since the null value is in every data type, the data type of the null value implied by the keyword NULL cannot be inferred; hence NULL can be used to denote the null value only in certain contexts, rather than everywhere that a literal is permitted.
- Although the null value is neither equal to any other value nor not equal to any other value — it is *unknown* whether or not it is equal to any given value — in some contexts, multiple null values are treated together; for example, the <group by clause> treats all null values together.

4.4.3 Predefined types

4.4.3.1 Numeric types

There are two classes of numeric type: *exact numeric*, which includes integer types and types with specified precision and scale; and *approximate numeric*, which is essentially floating point, and for which a precision may optionally be specified.

Every number has a *precision* (number of digits), and exact numeric types also have a scale (digits after the radix point). Arithmetic operations may be performed on operands of different or the same numeric type, and the result is of a numeric type that depends only on the numeric type of the operands. If the result cannot be represented exactly in the result type, then whether it is rounded or truncated is implementation-defined. An exception condition is raised if the result is outside the range of numeric values of the result type, or if the arithmetic operation is not defined for the operands.

4.4.3.2 Character string types

A value of *character string type* is a string (sequence) of characters drawn from some character repertoire. The characters in a character string *S* are all drawn from the same character set *CS*. If *S* is the value of some expression *E*, then *CS* is the character set specified for the declared type of *E*. A character string type is either of fixed length, or of variable length up to some implementation-defined maximum. A value of *character large object type* is a string of characters from some character repertoire and is always associated with exactly one character set. A *large object character string* is of variable length, up to some implementation-defined maximum that is probably greater than that of other character strings.

A character string may be specified as being based on a specific character set by specifying CHARACTER SET in the data type; a particular character set chosen by the implementation to be the *national character set* may be specified by specifying NATIONAL CHARACTER, NATIONAL CHARACTER VARYING, or NATIONAL CHARACTER LARGE OBJECT (or one of several syntactic equivalents) as the data type.