INTERNATIONAL STANDARD

ISO/IEC 9075-9

Fifth edition
2023-06

Information technology — Database languages SQL —

Part 9: Management of External Data (SQL/MED)

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

Partie 9: Gestion des données externes (SQL/MED)

Reference number
ISO/IEC 9075-9:2023(E)

Documents
Document Sharing Hub
© ISO/IEC 2023

Full document details: ISO IEC 09075-9-2023

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

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO IEC 09075-9-2023

Contents

Page

Foreword . . . xiii
Introduction . . . xv
1 Scope . . . 1
2 Normative references . . . 2
3 Terms and definitions . . . 3
4 Concepts . . . 6
4.1 Notations and conventions . . . 6
4.1.1 Notations . . . 6
4.2 Data types . . . 6
4.2.1 Naming of predefined types . . . 6
4.2.2 Data type terminology . . . 6
4.3 Columns, fields, and attributes . . . 6
4.4 Tables . . . 7
4.4.1 Introduction to tables . . . 7
4.4.2 Base tables . . . 7
4.4.2.1 Foreign tables . . . 7
4.4.3 Unique identification of tables . . . 7
4.4.4 Table descriptors . . . 7
4.4.5 Syntactic analysis of derived tables and cursors . . . 7
4.5 Functional dependencies . . . 8
4.5.1 Known functional dependencies in a foreign table . . . 8
4.6 SQL-schemas . . . 8
4.7 SQL-statements . . . 8
4.7.1 SQL-statements classified by function . . . 8
4.7.1.1 SQL-schema statements . . . 8
4.7.1.2 SQL-session statements . . . 9
4.8 Basic security model . . . 9
4.8.1 Privileges . . . 9
4.9 SQL-transactions . . . 9
4.9.1 Properties of SQL-transactions . . . 9
4.10 SQL-sessions . . . 9
4.10.1 SQL-session properties . . . 9
4.11 Introduction to SQL/CLI . . . 10
4.12 Foreign servers . . . 10
4.13 Foreign-data wrappers . . . 11
4.14 User mappings . . . 12
4.15 Routine mappings . . . 12
4.16 Generic options . . . 13

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

4.17 Capabilities and options information. . . . 13
4.18 Datalinks. . . . 14
4.18.1 Introduction to datalinks. . . . 14
4.18.2 Operations involving datalinks. . . . 17
4.18.2.1 Operators that operate on datalinks. . . . 17
4.18.2.2 Other operators involving datalinks. . . . 18
4.19 Foreign-data wrapper interface. . . . 18
4.19.1 Introduction to foreign-data wrapper interface. . . . 18
4.19.2 Handles. . . . 18
4.19.3 Foreign server sessions. . . . 20
4.19.4 Foreign-data wrapper interface routines. . . . 20
4.19.4.1 Introduction to foreign-data wrapper interface routines. . . . 20
4.19.4.2 Handle routines. . . . 20
4.19.4.3 Initialization routines. . . . 24
4.19.4.4 Access routines. . . . 24
4.19.4.5 Termination routines. . . . 25
4.19.4.6 Decomposition and pass-through modes. . . . 26
4.19.4.7 Sequence of actions during the execution of foreign server requests. . . . 26
4.19.5 Return codes. . . . 38
4.19.6 Foreign-data wrapper diagnostics areas. . . . 38
4.19.7 Null pointers. . . . 40
4.19.8 Foreign-data wrapper descriptor areas. . . . 40
5 Lexical elements. . . . 44
5.1 <token> and <separator>. . . . 44
5.2 Names and identifiers. . . . 46
6 Scalar expressions. . . . 48
6.1 <data type>. . . . 48
6.2 <cast specification>. . . . 51
6.3 <value expression>. . . . 52
6.4 <string value function>. . . . 53
6.5 <datalink value expression>. . . . 57
6.6 <datalink value function>. . . . 58
7 Query expressions. . . . 61
7.1 <table reference>. . . . 61
8 Additional common rules. . . . 68
8.1 Retrieval assignment. . . . 68
8.2 Store assignment. . . . 69
8.3 Result of data type combinations. . . . 70
8.4 Type precedence list determination. . . . 71
8.5 Determination of identical values. . . . 72
8.6 Equality operations. . . . 73
8.7 Grouping operations. . . . 74
8.8 Multiset element grouping operations. . . . 75
8.9 Ordering operations. . . . 76
9 Additional common elements. . . . 77
9.1 <generic options>. . . . 77

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

9.2 <alter generic options>. 78
10 Schema definition and manipulation. 80
10.1 <schema definition>. 80
10.2 <drop schema statement>. 81
10.3 <table definition>. 82
10.4 <unique constraint definition>. 83
10.5 <check constraint definition>. 84
10.6 <alter column data type clause>. 85
10.7 <drop column definition>. 86
10.8 <domain definition>. 87
10.9 <assertion definition>. 88
10.10 <user-defined type definition>. 89
10.11 <SQL-invoked routine>. 90
10.12 <drop routine statement>. 91
10.13 <user-defined cast definition>. 92
10.14 <user-defined ordering definition>. 93
10.15 <foreign table definition>. 94
10.16 <alter foreign table statement>. 97
10.17 <add basic column definition>. 98
10.18 <alter basic column definition>. 99
10.19 <drop basic column definition>. 100
10.20 <drop foreign table statement>. 102
11 Access control. 104
11.1 <privileges>. 104
11.2 <revoke statement>. 105
11.3 <user mapping definition>. 106
11.4 <alter user mapping statement>. 108
11.5 <drop user mapping statement>. 109
12 SQL-client modules. 110
12.1 <SQL-client module definition>. 110
12.2 <externally-invoked procedure>. 111
12.3 <SQL procedure statement>. 114
12.4 Data type correspondences. 115
13 Additional data manipulation rules. 118
13.1 Effect of deleting rows from base tables. 118
13.2 Effect of inserting tables into base tables. 119
13.3 Effect of replacing rows in base tables. 121
14 Session management. 123
14.1 <set passthrough statement>. 123
15 Dynamic SQL. 125
15.1 Description of SQL descriptor areas. 125
15.2 <prepare statement>. 126
15.3 <deallocate prepared statement>. 128
15.4 <describe statement>. 129
15.5 <input using clause>. 131

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

15.6 <output using clause>. 135
15.7 <execute statement>. 138
15.8 <dynamic declare cursor>. 139
15.9 <allocate extended dynamic cursor statement>. 140
15.10 <allocate received cursor statement>. 141
15.11 <dynamic open statement>. 142
15.12 <dynamic fetch statement>. 143
15.13 <dynamic close statement>. 144

16 Embedded SQL. 145
16.1 <embedded SQL Ada program>. 145
16.2 <embedded SQL C program>. 147
16.3 <embedded SQL COBOL program>. 148
16.4 <embedded SQL Fortran program>. 149
16.5 <embedded SQL MUMPS program>. 150
16.6 <embedded SQL Pascal program>. 151
16.7 <embedded SQL PL/I program>. 152

17 Diagnostics management. 153
17.1 <get diagnostics statement>. 153

18 Call-Level Interface specifications. 155
18.1 <CLI routine>. 155
18.2 Implicit DESCRIBE USING clause. 156
18.3 Description of CLI item descriptor areas. 157
18.4 Other tables associated with CLI. 158
18.5 SQL/CLI data type correspondences. 160

19 SQL/CLI routines. 162
19.1 BuildDataLink(). 162
19.2 GetDataLinkAttr(). 163
19.3 GetInfo(). 165

20 URLs. 166
20.1 URL format. 166

21 Catalog manipulation. 169
21.1 <foreign server definition>. 169
21.2 <alter foreign server statement>. 171
21.3 <drop foreign server statement>. 172
21.4 <foreign-data wrapper definition>. 174
21.5 <alter foreign-data wrapper statement>. 176
21.6 <drop foreign-data wrapper statement>. 177
21.7 <import foreign schema statement>. 178
21.8 <routine mapping definition>. 180
21.9 <alter routine mapping statement>. 181
21.10 <drop routine mapping statement>. 182

22 SQL/MED common specifications. 183
22.1 Description of foreign-data wrapper item descriptor areas. 183
22.2 Implicit foreign-data wrapper cursor. 187
22.3 Implicit DESCRIBE INPUT USING clause for external data. 189

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

22.4 Implicit DESCRIBE OUTPUT USING clause for external data. 192
22.5 Implicit EXECUTE USING and OPEN USING clauses for external data. 196
22.6 Implicit FETCH USING clause for external data. 199
22.7 Character string retrieval for external data. 203
22.8 Binary string retrieval for external data. 204
22.9 Tables used with SQL/MED. 205

23 Foreign-data wrapper interface routines. 216

23.1 <foreign-data wrapper interface routine>. 216
23.2 <foreign-data wrapper interface routine> invocation. 221
23.3 AdvanceInitRequest(). 223
23.4 AllocQueryContext(). 224
23.5 AllocWrapperEnv(). 225
23.6 Close(). 227
23.7 ConnectServer(). 228
23.8 FreeExecutionHandle(). 230
23.9 FreeFSConnection(). 232
23.10 FreeQueryContext(). 233
23.11 FreeReplyHandle(). 234
23.12 FreeWrapperEnv(). 235
23.13 GetNextReply(). 236
23.14 GetNumReplyBoolVE(). 237
23.15 GetNumReplyOrderBy(). 238
23.16 GetNumReplySelectElems(). 239
23.17 GetNumReplyTableRefs(). 240
23.18 GetOpts(). 241
23.19 GetReplyBoolVE(). 243
23.20 GetReplyCardinality(). 244
23.21 GetReplyDistinct(). 245
23.22 GetReplyExecCost(). 246
23.23 GetReplyFirstCost(). 247
23.24 GetReplyOrderElem(). 248
23.25 GetReplyReExecCost)(. 249
23.26 GetReplySelectElem(). 250
23.27 GetReplyTableRef(). 251
23.28 GetSPDHandle(). 252
23.29 GetSRDHandle(). 253
23.30 GetStatistics(). 254
23.31 GetWPDHandle(). 256
23.32 GetWRDHandle(). 257
23.33 InitRequest(). 258
23.34 Iterate(). 262
23.35 Open(). 264
23.36 ReOpen(). 265
23.37 Execution of Open(). 266
23.38 TransmitRequest(). 270
23.39 AllocDescriptor(). 273
23.40 FreeDescriptor(). 274

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

23.41 GetAuthorizationId() . . . 275
23.42 GetBoolVE() . . . 276
23.43 GetDescriptor() . . . 277
23.44 GetDistinct() . . . 279
23.45 GetNumBoolVE() . . . 280
23.46 GetNumChildren() . . . 281
23.47 GetNumOrderByElems() . . . 282
23.48 GetNumRoutMapOpts() . . . 283
23.49 GetNumSelectElems() . . . 284
23.50 GetNumServerOpts() . . . 285
23.51 GetNumTableColOpts() . . . 286
23.52 GetNumTableOpts() . . . 288
23.53 GetNumTableRefElems() . . . 289
23.54 GetNumUserOpts() . . . 290
23.55 GetNumWrapperOpts() . . . 291
23.56 GetOrderByElem() . . . 292
23.57 GetRoutMapOpt() . . . 293
23.58 GetRoutMapOptName() . . . 295
23.59 GetRoutineMapping() . . . 297
23.60 GetSelectElem() . . . 298
23.61 GetSelectElemType() . . . 299
23.62 GetServerName() . . . 300
23.63 GetServerOpt() . . . 301
23.64 GetServerOptByName() . . . 303
23.65 GetServerType() . . . 305
23.66 GetServerVersion() . . . 306
23.67 GetSQLString() . . . 307
23.68 GetTableColOpt() . . . 308
23.69 GetTableColOptByName() . . . 310
23.70 GetTableOpt() . . . 312
23.71 GetTableOptByName() . . . 313
23.72 GetTableRefElem() . . . 315
23.73 GetTableRefElemType() . . . 316
23.74 GetTableRefTableName() . . . 317
23.75 GetTableServerName() . . . 318
23.76 GetTRDHandle() . . . 319
23.77 GetUserOpt() . . . 320
23.78 GetUserOptByName() . . . 322
23.79 GetValExprColName() . . . 324
23.80 GetValueExpDesc() . . . 325
23.81 GetValueExpKind() . . . 326
23.82 GetValueExpName() . . . 327
23.83 GetValueExpTable() . . . 328
23.84 GetVEChild() . . . 329
23.85 GetWrapperLibraryName() . . . 330
23.86 GetWrapperName() . . . 331
23.87 GetWrapperOpt() . . . 332

© ISO/IEC 2023 – All rights reserved
Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

23.88 GetWrapperOptByName() . . . 334
23.89 SetDescriptor() . . . 336
23.90 GetDiagnostics() . . . 341

24 Information Schema . . . 345
24.1 Information Schema digital artifact . . . 345
24.2 ATTRIBUTES view . . . 345
24.3 COLUMN_OPTIONS view . . . 346
24.4 COLUMNS view . . . 347
24.5 FOREIGN_DATA_WRAPPER_OPTIONS view . . . 348
24.6 FOREIGN_DATA_WRAPPERS view . . . 349
24.7 FOREIGN_SERVER_OPTIONS view . . . 350
24.8 FOREIGN_SERVERS view . . . 351
24.9 FOREIGN_TABLE_OPTIONS view . . . 352
24.10 FOREIGN_TABLES view . . . 353
24.11 ROUTINE_MAPPING_OPTIONS view . . . 354
24.12 ROUTINE_MAPPINGS view . . . 355
24.13 USER_MAPPING_OPTIONS view . . . 356
24.14 USER_MAPPINGS view . . . 357
24.15 Short name views . . . 358

25 Definition Schema . . . 362
25.1 Definition Schema digital artifact . . . 362
25.2 COLUMN_OPTIONS base table . . . 362
25.3 DATA_TYPE_DESCRIPTOR base table . . . 364
25.4 FOREIGN_DATA_WRAPPER_OPTIONS base table . . . 371
25.5 FOREIGN_DATA_WRAPPERS base table . . . 372
25.6 FOREIGN_SERVER_OPTIONS base table . . . 373
25.7 FOREIGN_SERVERS base table . . . 374
25.8 FOREIGN_TABLE_OPTIONS base table . . . 375
25.9 FOREIGN_TABLES base table . . . 376
25.10 ROUTINE_MAPPING_OPTIONS base table . . . 377
25.11 ROUTINE_MAPPINGS base table . . . 378
25.12 SQL_SIZING base table . . . 379
25.13 TABLES base table . . . 380
25.14 USAGE_PRIVILEGES base table . . . 381
25.15 USER_MAPPING_OPTIONS base table . . . 382
25.16 USER_MAPPINGS base table . . . 383

26 Status codes . . . 384
26.1 SQLSTATE . . . 384

27 Conformance . . . 387
27.1 Claims of conformance to SQL/MED . . . 387
27.2 Additional conformance requirements for SQL/MED . . . 387

Annex A (informative) **SQL conformance summary . . . 390**

Annex B (informative) **Implementation-defined elements . . . 408**

Annex C (informative) **Implementation-dependent elements . . . 422**

Annex D (informative) **SQL optional feature taxonomy . . . 427**

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO IEC 09075-9-2023

Annex E (informative) **Deprecated features** 429
Annex F (informative) **Incompatibilities with ISO/IEC 9075:2016** 430
Annex G (informative) **Defect Reports not addressed in this edition of this document** 431
Annex H (informative) **Example header files** 432
H.1 C header file sqlcli.h 432
H.2 COBOL library item SQLCLI 433
Annex I (informative) **SQL/MED model** 435
Bibliography 439
Index 440

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Tables

Table

Page

1 Valid datalink file control options. . . . 16
2 Sequence of actions during the execution of foreign server requests. . . . 26
3 Fields used in foreign-data wrapper diagnostics areas. . . . 39
4 Fields in foreign-data wrapper descriptor areas. . . . 41
5 Data type correspondences for Ada. . . . 115
6 Data type correspondences for C. . . . 115
7 Data type correspondences for COBOL. . . . 116
8 Data type correspondences for Fortran. . . . 116
9 Data type correspondences for M. . . . 116
10 Data type correspondences for Pascal. . . . 116
11 Data type correspondences for PL/I. . . . 117
12 Codes used for SQL data types in Dynamic SQL. . . . 125
13 SQL-statement codes. . . . 153
14 Abbreviated SQL/CLI generic names. . . . 155
15 Codes used for implementation data types in SQL/CLI. . . . 157
16 Codes used for application data types in SQL/CLI. . . . 157
17 Codes used to identify SQL/CLI routines. . . . 158
18 Codes and data types for implementation information. . . . 158
19 Codes used for datalink attributes. . . . 158
20 Data types of attributes. . . . 158
21 SQL/CLI data type correspondences for Ada. . . . 160
22 SQL/CLI data type correspondences for C. . . . 160
23 SQL/CLI data type correspondences for COBOL. . . . 160
24 SQL/CLI data type correspondences for Fortran. . . . 161
25 SQL/CLI data type correspondences for M. . . . 161
26 SQL/CLI data type correspondences for Pascal. . . . 161
27 SQL/CLI data type correspondences for PL/I. . . . 161
28 Codes used for <table reference> types. . . . 205
29 Codes used for <value expression> kinds. . . . 205
30 Codes used for foreign-data wrapper diagnostic fields. . . . 205
31 Codes used for foreign-data wrapper descriptor fields. . . . 206
32 Codes used for foreign-data wrapper handle types. . . . 208
33 Ability to retrieve foreign-data wrapper descriptor fields. . . . 208
34 Ability to set foreign-data wrapper descriptor fields. . . . 210
35 Foreign-data wrapper descriptor field default values. . . . 213
36 Codes used for the format of the character string transmitted by GetSQLString(). . . . 215
37 SQLSTATE class and subclass codes. . . . 384
38 Implied feature relationships of SQL/MED. . . . 389
D.1 Feature taxonomy for optional features. . . . 427
I.1 Legend for SQL/MED interfaces. . . . 436
I.2 Legend for SQL/MED information flow. . . . 437

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub

Full document details: ISO IEC 09075-9-2023

Figures

Figure

Page

1 SQL/MED interfaces. 435
2 SQL/MED information flow. 437

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

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 fifth edition cancels and replaces the fourth edition (ISO/IEC 9075-9:2016), which has been technically revised. It also incorporates the Technical Corrigenda ISO/IEC 9075-9:2016/Cor.1:2019 and ISO/IEC 9075-9:2016/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 fifth edition of ISO/IEC 9075-9 is designed to be used in conjunction with the following editions of other parts of the ISO/IEC 9075 series, all published in 2023:

— ISO/IEC 9075-1, sixth edition;

— ISO/IEC 9075-2, sixth edition;

Documents
Document Sharing Hub

© ISO/IEC 2023 – All rights reserved

Full document details: ISO IEC 09075-9-2023

— ISO/IEC 9075-3, sixth edition;

— ISO/IEC 9075-4, seventh 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.

Documents
Document Sharing Hub

© ISO/IEC 2023 – All rights reserved

Full document details: ISO IEC 09075-9-2023

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 related to this document.

5) Clause 5, "Lexical elements", defines the lexical elements of the language specified in this document.

6) Clause 6, "Scalar expressions", defines the elements of the language that produce scalar values.

7) Clause 7, "Query expressions", defines the elements of the language that produce rows and tables of data.

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 foreign tables, foreign servers, and foreign-data wrappers.

10) Clause 10, "Schema definition and manipulation", defines facilities related to foreign tables and datalink type support for creating and managing a schema.

11) Clause 11, "Access control", defines facilities for controlling access to SQL-data.

12) Clause 12, "SQL-client modules", defines SQL-client modules and externally-invoked procedures.

13) Clause 13, "Additional data manipulation rules", defines additional rules for data manipulation.

14) Clause 14, "Session management", defines the SQL-session management statements.

15) Clause 15, "Dynamic SQL", defines the dynamic SQL statements.

16) Clause 16, "Embedded SQL", defines the embedded SQL statements.

17) Clause 17, "Diagnostics management", defines the diagnostics management facilities.

18) Clause 18, "Call-Level Interface specifications", defines facilities for using SQL through a Call-Level Interface.

19) Clause 19, "SQL/CLI routines", defines each of the routines that comprise the Call-Level Interface.

20) Clause 20, "URLs", specifies the format of URLs used in this document.

21) Clause 21, "Catalog manipulation", defines facilities for creating, altering, and dropping foreign servers and foreign-data wrappers.

22) Clause 22, "SQL/MED common specifications", specifies common facilities used by SQL/MED.

23) Clause 23, "Foreign-data wrapper interface routines", specifies the interaction between an SQL-server and a foreign-data wrapper.

24) Clause 24, "Information Schema", defines viewed tables that contain schema information.

25) Clause 25, "Definition Schema", defines base tables on which the viewed tables containing schema information depend.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

26) Clause 26, "Status codes", defines values that identify the status of the execution of SQL-statements and the mechanisms by which those values are returned.

27) Clause 27, "Conformance", specifies the way in which conformance to this document may be claimed.

28) Annex A, "SQL conformance summary", is an informative Annex. It summarizes the conformance requirements of the SQL language.

29) Annex B, "Implementation-defined elements", is an informative Annex. It lists those features for which the body of 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.

30) 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.

31) 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.

32) 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.

33) Annex F, "Incompatibilities with ISO/IEC 9075:2016", is an informative Annex. It lists incompatibilities with the previous edition of this document.

34) 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 this document. No new problems have been created in the drafting of this edition of this document.

35) Annex H, "Example header files", is an informative Annex. It provides examples of typical definition files for application programs using the SQL Call-Level Interface.

36) Annex I, "SQL/MED model", is an informative Annex. It uses annotated diagrams to illustrate the more important concepts of the model of SQL/MED, including the relationships between the SQL-server, foreign-data wrappers, and foreign servers.

In the text of this document, Clauses and Annexes begin new odd-numbered pages, and in Clause 5, "Lexical elements", through Clause 27, "Conformance", Subclauses begin new pages. Any resulting blank space is not significant.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Information technology — Database language SQL —

Part 9:
Management of External Data (SQL/MED)

1 Scope

This document defines extensions to Database Language SQL to support management of external data through the use of foreign-data wrappers and datalink types.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

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

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

W3C XML 1.0 *Extensible Markup Language (XML) Version 1.0, W3C Recommendation*. Edited by: Bray, Tim et al. 26 November 2008, revised 7 February 2013
Available at: https://www.w3.org/TR/xml

W3C XML 1.1 *Extensible Markup Language (XML) Version 1.1, W3C Recommendation*. Edited by: Bray, Tim et al. 16 August 2006, revised 29 September 2006
Available at: https://www.w3.org/TR/xml11

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

3 Terms and definitions

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

3.1
Valid XML document
Valid XML 1.0 document (3.2) or *Valid XML 1.1 document* (3.3)

3.2
Valid XML 1.0 document
well-formed XML document that meets certain further constraints, as defined in XML 1.0

3.3
Valid XML 1.1 document
well-formed XML document that meets certain further constraints, as defined in XML 1.1

3.4
XML document
XML 1.0 document (3.5) or *XML 1.1 document* (3.6)

3.5
XML 1.0 document
well-formed XML document, as defined in XML 1.0

3.6
XML 1.1 document
well-formed XML document, as defined in XML 1.1

3.7
DTD
XML 1.0 document type declaration (3.8) or *XML 1.1 document type declaration* (3.9)

3.8
XML 1.0 document type declaration
document type declaration, as defined in XML 1.0

3.9
XML 1.1 document type declaration
document type declaration, as defined in XML 1.1

3.10
access token
encrypted value returned under certain conditions by an SQL-server in combination with the File Reference of a datalink value

Note 1 to entry: An access token is either a *read token* or a *write token*.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

3.11
datalink
value, of data type DATALINK, referencing some file that is not part of the SQL-environment

Note 1 to entry: The file is assumed to be managed by some external file manager.

3.12
datalinker
implementation-dependent component for enabling integrity control, recovery, and access control for external files

3.13
external data
data that is not managed by an SQL-server involved in an SQL-session, but that is nevertheless accessible to that SQL-session

3.14
foreign-data wrapper
named collection of routines, invocable by the SQL-server, supporting the programming interface specified for such routines in this document

3.15
foreign server
named server, external to the SQL-environment, but known to the SQL-server, that manages external data

3.16
foreign server request
statement that an SQL-server submits to a foreign-data wrapper

3.17
foreign table
named table whose rows are supplied when needed by some foreign server

Note 1 to entry: The mechanism by which these rows are supplied is provided by a foreign-data wrapper. The data constituting a foreign table is not part of the SQL-environment.

3.18
integrity control option
link control option specifying the level of integrity of the link between a datalink and the file that it references

3.19
link control
property of a column of data type DATALINK, specifying the extent to which the links between datalinks in that column and the files they reference are to be monitored (in various specific manners)

3.20
read permission option
link control option specifying how permission to read external files referenced by certain datalinks is determined

3.21
recovery option
link control option specifying whether or not point in time recovery is required for the files referenced by certain datalinks

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

3.22
routine mapping
implementation-defined mapping of an SQL-invoked routine to an equivalent concept maintained by a foreign server

3.23
unlink option
link control option specifying the action to be taken when certain sites occupied by datalinks are updated or deleted

3.24
user mapping
implementation-defined mapping of an authorization identifier to an equivalent concept maintained by a foreign server

3.25
write permission option
link control option specifying how permission to write files referenced by certain datalinks is determined

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

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/-9/ed-5/en/` to download digital artifacts for this document. To download the syntax defined in a plain-text format, select the file named `ISO_IEC_9075-9(E)_MED.bnf.txt`. To download the syntax defined in an XML format, select the file named `ISO_IEC_9075-9(E)_MED.bnf.xml`.

4.2 Data types

This Subclause modifies Subclause 4.2, "Data types", in ISO/IEC 9075-2.

4.2.1 Naming of predefined types

This Subclause modifies Subclause 4.2.2, "Naming of predefined types", in ISO/IEC 9075-2.

Insert after the 1st paragraph: SQL defines a predefined data type named by the following <key word>: DATALINK.

Insert after the 3rd paragraph: For reference purposes, the data type DATALINK is referred to as a (or the) *datalink type*.

4.2.2 Data type terminology

This Subclause modifies Subclause 4.2.4, "Data type terminology", in ISO/IEC 9075-2.

Insert into the 11th paragraph, after the last list item:

— A type *T* is *DATALINK-ordered* if *T* is *S*-ordered, where *S* is the set of datalink types.

4.3 Columns, fields, and attributes

This Subclause modifies Subclause 4.15, "Columns, fields, and attributes", in ISO/IEC 9075-2.

Insert after the 8th paragraph: The term *constituent* is defined for values such that a value *V2* either is or is not a constituent of a value *V1*.

NOTE 1 — For example, the integer 2 and the character string 'one' are both constituents of the row value denoted by `ROW(2, 'one')`. By contrast, the integer 3 is not a constituent of that row value.

Insert after the 8th paragraph: *V2* is an *immediate constituent* of *V1* if any of the following are true.

— *V1* is a value of some predefined data type or of some distinct type whose source type is some predefined data type and *V2* is identical to *V1*.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— *V1* is a value of some structured type *ST* and, for some attribute *A* of *ST*, *V2* is identical to *V1.A*().

— *V1* is a value of some row type *RT* and, for some field *F* of *RT*, *V2* is identical to *V1.F*.

— *V1* is a value of some collection type *CT* and *V2* is an element of *V1*.

Insert after the 8th paragraph: *V2* is a *constituent* of *V1* if *V2* is an immediate constituent of *V1* or there is some value *V3* such that *V3* is an immediate constituent of *V1* and *V2* is a constituent of *V3*.

4.4 Tables

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

4.4.1 Introduction to tables

This Subclause modifies Subclause 4.17.1, "Introduction to tables", in ISO/IEC 9075-2.

Augment the 3rd paragraph by adding "foreign table" to the list of table types.

4.4.2 Base tables

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

4.4.2.1 Foreign tables

The data constituting a foreign table is not part of the SQL-environment. Instead, its rows are supplied when needed by some foreign server, known as the source of the foreign table. The mechanism by which these rows are supplied is provided by a foreign-data wrapper (see Subclause 4.13, "Foreign-data wrappers").

4.4.3 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 a foreign table uniquely identifies a multiset of rows.

4.4.4 Table descriptors

This Subclause modifies Subclause 4.17.7, "Table descriptors", in ISO/IEC 9075-2.

Augment the 1st paragraph by adding "foreign table descriptor" to the list of possible table descriptors..

Insert after the 6th paragraph: A foreign table descriptor describes a foreign table. In addition to the components of every table descriptor, a foreign table descriptor includes the following components.

— The name of the foreign table.

— A foreign server name, identifying the descriptor of the foreign server that is the source of the foreign table.

— A generic options descriptor.

— An indication of whether the foreign table is updatable or not.

NOTE 2 — This document currently restricts foreign tables such that they are neither insertable-into nor updatable. It is possible that this restriction will not appear in future editions.

4.4.5 Syntactic analysis of derived tables and cursors

This Subclause modifies Subclause 4.17.8, "Syntactic analysis of derived tables and cursors", in ISO/IEC 9075-2.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Insert into the 7th paragraph, after the 1st list item:

— If *TORQN* identifies a foreign table, then *TORQN* has no generally underlying table specifications.

4.5 Functional dependencies

This Subclause modifies Subclause 4.26, "Functional dependencies", in ISO/IEC 9075-2.

4.5.1 Known functional dependencies in a foreign table

There are no rules in this document to determine known functional dependencies in a foreign table. However, implementation-defined rules may determine known functional dependencies, if any, in a foreign table.

4.6 SQL-schemas

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

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

— A foreign table descriptor.

Augment the 5th paragraph by adding "foreign tables" to the list of objects that are identified by <table name>s.

4.7 SQL-statements

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

4.7.1 SQL-statements classified by function

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

4.7.1.1 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:

— <import foreign schema statement>

— <foreign table definition>

— <alter foreign table statement>

— <drop foreign table statement>

— <foreign server definition>

— <alter foreign server statement>

— <drop foreign server statement>

— <foreign-data wrapper definition>

— <alter foreign-data wrapper statement>

— <drop foreign-data wrapper statement>

— <user mapping definition>

— <alter user mapping statement>

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— <drop user mapping statement>

— <routine mapping definition>

— <alter routine mapping statement>

— <drop routine mapping statement>

4.7.1.2 SQL-session statements

This Subclause modifies Subclause 4.41.2.7, "SQL-session statements", in ISO/IEC 9075-2.

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

— <set passthrough statement>

4.8 Basic security model

This Subclause modifies Subclause 4.42, "Basic security model", in ISO/IEC 9075-2.

4.8.1 Privileges

This Subclause modifies Subclause 4.42.2, "Privileges", in ISO/IEC 9075-2.

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

— foreign table

— foreign-data wrapper

— foreign server

NOTE 3 — Privileges granted on foreign tables are not privileges to use the data constituting foreign tables, but privileges to use the definitions of the foreign tables. The privileges to access the data constituting the foreign tables are enforced by the foreign server, based on the user mapping. Consequently, a request by an SQL-client to access external data can raise exceptions.

Insert into the 11th paragraph, after the last list item:

— foreign-data wrapper

— foreign server

4.9 SQL-transactions

This Subclause modifies Subclause 4.43, "SQL-transactions", in ISO/IEC 9075-2.

4.9.1 Properties of SQL-transactions

This Subclause modifies Subclause 4.43.3, "Properties of SQL-transactions", in ISO/IEC 9075-2.

Augment the 3rd paragraph by adding "foreign tables" to the list of tables.

4.10 SQL-sessions

This Subclause modifies Subclause 4.45, "SQL-sessions", in ISO/IEC 9075-2.

4.10.1 SQL-session properties

This Subclause modifies Subclause 4.45.3, "SQL-session properties", in ISO/IEC 9075-2.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Insert before the 13th paragraph: At any time during an SQL-session, the SQL-server may obtain a WrapperEnvHandle for a foreign-data wrapper and an FSConnectionHandle for a foreign server.

Insert into the 13th paragraph, after the last list item:

— Zero or more {foreign-data wrapper name : WrapperEnvHandle} pairs.

— Zero or more {foreign server name : FSConnectionHandle} pairs.

— A pass-through flag.

— A pass-through foreign server name, if any.

— Zero or more {<statement name> : ExecutionHandle} pairs.

Insert before the 13th paragraph: At the end of every SQL-session, every FSConnection handle that is contained in the SQL-session context is freed.

Insert before the 13th paragraph: At the end of every SQL-session, every WrapperEnv handle that is contained in the SQL-session context is freed.

Insert before the 13th paragraph: An SQL-session has a pass-through flag that is initially set to *False* when the SQL-session is started. The successful execution of a <set passthrough statement> that contains a <foreign server name> changes the pass-through flag to *True*. An SQL-session whose pass-through flag is *True* additionally has a pass-through foreign server name. Every time a <set passthrough statement> is executed, all {<SQL statement name> : ExecutionHandle} pairs are removed from the current SQL-session context. Every time a <set passthrough statement> that contains a <foreign server name> *FSN* is successfully executed, the pass-through foreign server name included the current SQL-session context is set to *FSN*. Every time a <prepare statement> is executed after a <set passthrough statement> that contains a <foreign server name> has been executed successfully, an {<SQL statement name> : Execution-Handle} pair is made part of the current SQL-session context. Every time a <deallocate prepared statement> is successfully executed after a <set passthrough statement> that identifies a <foreign server name> has been executed, the corresponding {<SQL statement name> : ExecutionHandle} pair is removed from the current SQL-session context. Every time a <set passthrough statement> that specifies OFF is executed, the pass-through flag is set to *False* and the pass-through foreign server name is deleted from the current SQL-session context.

4.11 Introduction to SQL/CLI

This Subclause modifies Subclause 4.2, "Introduction to SQL/CLI", in ISO/IEC 9075-3.

Insert after the 22nd paragraph: The BuildDataLink routine can be used to build a datalink value. The GetDataLinkAttr routine can be used to extract the attributes of a datalink value.

4.12 Foreign servers

A *foreign server* is a named server, external to the SQL-environment but known to the SQL-server, that manages external data. Such external data is manifested as SQL-data by use of a mechanism called a *foreign-data wrapper* (see Subclause 4.13, "Foreign-data wrappers").

NOTE 4 — The abbreviation "FDW" is sometimes used to evoke the notion of a "foreign-data wrapper".

A *foreign server descriptor* is a catalog element, identified by a *foreign server name* and created by invoking a <foreign server definition>. A foreign server descriptor consists of the following.

— A foreign server name, identifying the foreign server locally to the SQL-server.

— The authorization identifier of the owner of the foreign server descriptor.

— The name of the foreign-data wrapper.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— A generic options descriptor.

— Optionally, the foreign server type.

— Optionally, the foreign server version.

The possible values of server type and server version, and their meanings, are implementation-defined (IV126).

A foreign server descriptor is said to be *owned by* or to have been *created by* the current authorization identifier for the SQL-session when the <foreign server definition> was invoked.

A foreign server descriptor can be modified by an <alter foreign server statement> and destroyed by a <drop foreign server statement>.

A foreign server can be an *SQL-aware foreign server* or a *non-SQL-aware foreign server*. An SQL-aware foreign server is a foreign server that has the ability to process a subset of statements conforming to ISO/IEC 9075, particularly a subset of the <query specification> statement, in a standard-conforming manner. A non-SQL-aware foreign server is a foreign server that has no ability to process SQL language. If the foreign-data wrapper associated with a non-SQL-aware foreign server provides some (limited or conforming) ability to process SQL language, then the effect is that the foreign server can be treated as though it is an SQL-aware foreign server.

NOTE 5 — Some SQL-aware foreign servers can be, in fact, SQL-servers. However, because they are not in the same SQL-environment as the SQL-server responding to an SQL-client, they are managed only through foreign-data wrappers and are treated as foreign servers. Such foreign servers can concurrently respond to SQL-clients of their own; this does not change the relationships specified in this document.

Some foreign servers, especially SQL-aware foreign servers, admit the concept of a schema and the concept of a table that are similar to SQL-schemas and to base tables, respectively. Such servers may (and SQL-aware foreign servers do) maintain schema information about those entities, such as the Information Schema and Definition Schema specified in ISO/IEC 9075-11.

If a foreign server maintains schema information about entities analogous to SQL-schemas and base tables, then execution of an <import foreign schema statement> retrieves information about the tables (either all or only some, as specified in the <import foreign schema statement>) associated with the named SQL-schema analog and effectively performs one or more <foreign table definition> statement executions.

If a foreign server does not maintain such information or does not admit the concept of a schema, then foreign tables managed by that server shall be specified by means of explicit <foreign table definition>s.

ISO/IEC 9075-2 does not specify the manner in which the SQL-server and the foreign-data wrapper interact to cause information about foreign tables to be retrieved by execution of an <import foreign schema statement>. In particular, no foreign-data wrapper interface routines are specified to support such interaction. Such interaction is implementation-dependent (UW006).

4.13 Foreign-data wrappers

A foreign-data wrapper is the mechanism by which the SQL-server accesses external data managed by foreign servers. Every foreign server is accessed through exactly one foreign-data wrapper, but one foreign-data wrapper can be used to access several different foreign servers. A foreign-data wrapper is made up of foreign-data wrapper interface routines and a set of routines written in a programming language. Foreign-data wrapper interface routines are used to access every foreign server whose descriptor includes the name of that foreign-data wrapper. It is possible for a foreign-data wrapper to exist that is not used to access any foreign server.

A *foreign-data wrapper descriptor* is a catalog element, identified by a *foreign-data wrapper name* and created by invoking a <foreign-data wrapper definition>. A <foreign-data wrapper definition> specifies the foreign-data wrapper name, a library name that identifies a library containing the foreign-data

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

wrapper interface routines, and the name of the language in which the foreign-data wrapper interface routines are written.

A foreign-data wrapper descriptor consists of the following.

— A foreign-data wrapper name.

— The authorization identifier of the owner of the foreign-data wrapper descriptor.

— The name of the language in which the foreign-data wrapper interface routines are written.

— A generic options descriptor.

— A library name.

A foreign-data wrapper descriptor can be modified by an <alter foreign-data wrapper statement> and destroyed by a <drop foreign-data wrapper statement>.

4.14 User mappings

A user mapping is an SQL-environment element, pairing an authorization identifier *U* or the special identifier PUBLIC, denoting all <authorization identifier>s in the SQL-environment, with a foreign server *FS*. It defines how to map *U* to an equivalent concept known to *FS* when a foreign table whose source is *FS* is to be accessed during an SQL-session when the current authorization identifier is *U*. The mapping is specified by generic options defined by the foreign-data wrapper.

A user mapping is defined by invoking a <user mapping definition>. Invocation of a <user mapping definition> results in the creation of a user mapping descriptor in the SQL-environment. A user mapping descriptor consists of the following.

— An authorization identifier.

— A foreign server name, identifying a foreign server descriptor.

— A generic options descriptor.

A user mapping descriptor can be modified by an <alter user mapping statement> and destroyed by a <drop user mapping statement>.

4.15 Routine mappings

A routine mapping is an SQL-environment element, pairing an SQL-invoked routine *SIR* with a foreign server *FS*. It defines how to map *SIR* to an equivalent concept known to *FS* when a foreign table *FT* whose source is *FS* is to be accessed and the foreign server request that includes *FT* also includes a reference to *SIR*. The mapping is specified by generic options defined by the foreign-data wrapper.

A routine mapping is defined by invoking a <routine mapping definition>. Invocation of a <routine mapping definition> results in the creation of a routine mapping descriptor in the SQL-environment. A routine mapping descriptor consists of the following

— The name of the routine mapping.

— The specific routine name of the SQL-invoked routine.

— A foreign server name, identifying a foreign server descriptor.

— A generic options descriptor.

A routine mapping descriptor can be modified by an <alter routine mapping statement> and destroyed by a <drop routine mapping statement>.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

4.16 Generic options

Several of the objects used in connection with external data support the specification of *generic options*. These objects are foreign-data wrappers, foreign servers, foreign tables, columns of foreign tables, user mappings, and routine mappings. A generic option is an option name paired with an optional option value. Both the option name and the permissible ranges of option values of a generic option are defined by the foreign-data wrappers. A set of generic options is described by a generic options descriptor. A generic options descriptor is included in the descriptor of the object to which it pertains. The generic options are stored in the SQL-server for the foreign-data wrapper to retrieve when the foreign-data wrapper needs this information.

Generic options may be specified in either <foreign-data wrapper definition>, <foreign server definition>, <foreign table definition>, <user mapping definition>, <routine mapping definition>, <alter foreign-data wrapper statement>, <alter foreign server statement>, <alter foreign table statement>, <alter user mapping statement>, or <alter routine mapping statement>.

Generic options are specific to the object for which they are defined. For example, the generic options for a foreign table are most likely different from the generic options for a foreign server, in both option names and option values. Furthermore, generic options are highly dependent on the foreign-data wrapper that is used to access the external data. For example, the generic options for a foreign server that uses a foreign-data wrapper *A* might be totally different from the generic options specified for another foreign server that uses a foreign-data wrapper *B*. Even the fact that the option names of two generic options for two different foreign-data wrappers might be the same does not necessarily mean that the semantics and therefore the permissible ranges of option values are the same.

Since an SQL-server cannot anticipate the different kinds of foreign-data wrappers with which it is likely to deal, no generic option can ever be determined by the SQL-server or by this document. Only a foreign-data wrapper can specify generic options for that foreign-data wrapper, or for a foreign server, a foreign table, a column of a foreign table, a user mapping, or a routine mapping for which it is used.

A *generic options descriptor* is either an empty list or a list consisting of one or more option names, each option name being paired with at most one option value.

4.17 Capabilities and options information

The SQL-server needs information from the foreign-data wrapper about the capabilities of the foreign-data wrapper itself, about the foreign server accessed through the foreign-data wrapper, and about certain schema elements (foreign tables and their columns, user mappings) managed by the foreign server. The SQL-server also needs information about options supported by the foreign-data wrapper, the foreign server, and certain schema elements. The SQL-server invokes the `GetOpts()` routine to request the capabilities and other information from a foreign-data wrapper.

The specific capabilities and other information of a foreign-data wrapper, a foreign server, or any schema element managed by a foreign server that are reported to the SQL-server in response to an invocation of `GetOpts()` are partly specified in this document and partly implementation-defined. In general, each capability or other piece of information that is reported corresponds to a generic option associated with the object being queried by the invocation.

The capabilities and other information is returned in a buffer whose contents may comprise an XML document or that may be returned in a format defined by the foreign-data wrapper. If the contents comprise an XML document, then it shall be a valid XML document, the format of which is specified by a DTD (XML Document Type Declaration) that is either internal to the XML document or external (requiring that it be available to the SQL-server in an implementation-defined (IW014) manner).

NOTE 6 — This edition of this document specifies the use of a DTD. It is possible that future editions will specify the use of an XML Schema, either as an alternative to a DTD or instead of a DTD.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

4.18 Datalinks

4.18.1 Introduction to datalinks

A datalink is a value of the DATALINK data type. A datalink references some file that is not part of the SQL-environment. The file is assumed to be managed by some external file manager. A datalink is conceptually represented by the following.

— File Reference: A character string forming a reference to an external file.

— SQL-Mediated Read Access Indication: A Boolean value, where *True*, in datalink *DL* indicates that the referenced file, being linked to the SQL-environment, is accessible to be read only by use of the specially provided operations (see below) on *DL*.

— SQL-Mediated Write Access Indication: A Boolean value, where *True*, in datalink *DL* indicates that the referenced file, being linked to the SQL-environment, is accessible to be modified only by use of the specially provided operations (see below) on *DL*.

— Write Token: An implementation-dependent (UV032) value that represents an access token that is used to read or modify the File Reference. This value can be the null value.

— Construction Indication: A character string indicating how the datalink was constructed. Possible values are: NEWCOPY, PREVIOUSCOPY, and the null value.

The File Reference of a datalink is accessible by invoking operators defined in this document. The character set of the File Reference, referred to as the *datalink character set* is implementation-defined (IV127).

The purpose of datalinks is to provide a mechanism to synchronize the integrity control, recovery, and access control of the files and the SQL-data associated with them. This document standardizes the way that an SQL-server is made aware of datalink values and how applications retrieve information about the files identified by datalink values. The mechanisms that enable integrity control, recovery, and access control for the files represented by the datalink values are implementation-dependent (UW007). These mechanisms are collectively called the *datalinker*.

A file is *linked* to the SQL-environment whenever execution of an SQL-data change statement causes a value *DL1* that references that file to appear in some datalink column whose descriptor includes the link control FILE LINK CONTROL. If the read permission option included in the column descriptor is DB, then access to the referenced file is said to be *SQL-mediated*. This is indicated by setting the SQL-Mediated Read Access Indication of *DL1* to *True*, and *DL1* is said to be an *SQL-mediated datalink*. If the read permission option included in the column descriptor is not DB, then the SQL-Mediated Read Access Indication of *DL1* is set to *False*. If the write permission option included in the column descriptor is ADMIN, then this is indicated by setting the SQL-Mediated Write Access Indication of *DL1* to *True*. If the write permission option included in the column descriptor is not ADMIN, then the SQL-Mediated Write Access Indication of *DL1* is set to *False*.

Execution of an SQL-data change statement that causes a value *DL2* to appear in a datalink column defined with the link control NO LINK CONTROL does not cause any file to be linked to the SQL-environment.

A linked file cannot be renamed or deleted by any agency outside of the SQL-environment. A datalink value always references just one file. A file is *unlinked* from the SQL-environment whenever execution of an SQL-data change statement causes a datalink that references that file to be removed from some datalink column whose descriptor includes the link control FILE LINK CONTROL. The actions that occur when a datalink is removed from a column depend on the link control options that are specified in the column descriptor of that column. The file might be deleted, or the datalinker might return control of the file to the external data manager.

With the function provided by datalinks and the datalinker, it is possible to specify that access to the files should be mediated by the SQL-server rather than by the external data manager. When access to the files is mediated by an SQL-server, any request to access a file shall operate on an SQL-mediated datalink to

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

obtain a character string with which to reference the file, using one of the operators provided for that purpose. This character string is constructed by combining the File Reference of a datalink value with an encrypted value called an *access token*. An access token is either a *read token* or a *write token*, depending on the function that is used to construct the character string. The generation of the access token and the method of combining it with the File Reference is implementation-dependent (UA072). When the application uses the returned character string value to access a file, the datalinker checks to see if the access token is *valid*. If it is valid, then the application is allowed to access the file pointed to by the File Reference. Every attempt by an application to access, without a valid access token, a file referenced by an SQL-mediated datalink is unsuccessful. The time at which a valid access token ceases to be valid is implementation-defined (IV013).

The content of an SQL-mediated file cannot be modified, unless the SQL-Mediated Write Access Indication of the datalink value *DL* referencing this file is *True*. After an application has modified the file, it uses a <datalink value constructor> that specifies either DLNEWCOPY or DLPREVIOUSCOPY to construct a new datalink value. This new datalink value is then used to update the site that contains *DL*.

NOTE 7 — Updating the site that contains a datalink in the manner described here is called "update-in-place".

Datalinks are not comparable. A datalink is assignable only to sites of type DATALINK.

A datalink data type is described by a *datalink data type descriptor*. A datalink data type descriptor consists of the name DATALINK and the set of *link control options*.

— The link control (NO LINK CONTROL or FILE LINK CONTROL).

— The integrity control option (ALL, SELECTIVE, or NONE).

— The read permission option (FS or DB).

— The write permission option (FS, ADMIN, or BLOCKED). If the write permission option is ADMIN, then additionally the access token indication (either NOT REQUIRING TOKEN FOR UPDATE or REQUIRING TOKEN FOR UPDATE).

— The recovery option (NO or YES).

— The unlink option (RESTORE, DELETE, or NONE).

The meanings of the various link control options are given here.

— NO LINK CONTROL: Although every File Reference shall conform to the Format and Syntax Rules of Subclause 20.1, "URL format", it is permitted for there to be no file referenced by that File Reference. This option implies that the integrity control option is NONE, the read permission option is FS, the write permission option is FS, the recovery option is NO and the unlink option is NONE, and no explicit syntax to specify these options is permitted.

— FILE LINK CONTROL: Every File Reference shall reference an existing file. Further file control depends on the link control options.

— INTEGRITY ALL: Files referenced by File References cannot be deleted or renamed, except possibly through the use of operations on the column in question, invoked as part of some SQL-session.

— INTEGRITY SELECTIVE: Files referenced by File References can be deleted or renamed using operators provided by the file manager, unless a datalinker is installed in connection with the file manager.

— INTEGRITY NONE: Files referenced by File References can only be deleted or renamed using operators provided by the file manager. This option is not available if FILE LINK CONTROL is specified.

— READ PERMISSION FS: Permission to read files referenced by datalinks is determined by the file manager.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— READ PERMISSION DB: Datalinks of this type are SQL-Mediated. That is to say, permission to read files referenced by such datalinks is determined by the SQL-implementation.

— WRITE PERMISSION FS: Permission to write files referenced by datalinks is determined by the file manager.

— WRITE PERMISSION ADMIN REQUIRING TOKEN FOR UPDATE: Permission to write files referenced by datalinks is determined by the SQL-implementation and the datalinker. This option is only available if READ PERMISSION DB is also specified. If a site that was declared with this write permission is updated, then the access token used to open and modify the file is required to be contained in the file reference specified in the invocation of the functions DLNEWCOPY or DLPREVIOUSCOPY that yield the value with which the site is updated.

— WRITE PERMISSION ADMIN NOT REQUIRING TOKEN FOR UPDATE: Permission to write files referenced by datalinks is determined by the SQL-implementation and the datalinker. This option is only available if READ PERMISSION DB is also specified. If a site that was declared with this write permission is updated, then an access token is not required to be contained in the file reference specified in the invocation of the functions DLNEWCOPY or DLPREVIOUSCOPY that yield the value with which the site is updated.

— WRITE PERMISSION BLOCKED: Write access to files referenced by datalinks is not available. Updates can, however, arise indirectly through the use of some implementation-defined mechanism.

— RECOVERY YES: Enables *point in time recovery* of files referenced by datalinks.

NOTE 8 — "point in time recovery" is an implementation-defined mechanism that provides for recovery that is coordinated between the SQL-server and the files of external file manager referenced by datalinks.

— RECOVERY NO: Point in time recovery of files referenced by datalinks is disabled.

— ON UNLINK RESTORE: When a file referenced by a datalink is unlinked, the external file manager attempts to reinstate the ownership and permissions that existed when that file was linked.

— ON UNLINK DELETE: A file referenced by a datalink is deleted when it is unlinked.

— ON UNLINK NONE: When a file referenced by a datalink is unlinked, there is no change in the ownership and permissions occasioned by that unlinking.

Table 1, "Valid datalink file control options", specifies what combinations of datalink file control options are allowed.

Table 1 — Valid datalink file control options

Integrity	Read permission	Write permission	Recovery	Unlink
ALL	FS	FS	NO	NONE
ALL	FS	BLOCKED	NO	RESTORE
ALL	FS	BLOCKED	YES	RESTORE
ALL	DB	BLOCKED	NO	RESTORE
ALL	DB	BLOCKED	NO	DELETE
ALL	DB	BLOCKED	YES	RESTORE
ALL	DB	BLOCKED	YES	DELETE

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Integrity	Read permission	Write permission	Recovery	Unlink
ALL	DB	ADMIN	NO	RESTORE
ALL	DB	ADMIN	NO	DELETE
ALL	DB	ADMIN	YES	RESTORE
ALL	DB	ADMIN	YES	DELETE
SELECTIVE	FS	FS	NO	NONE

NOTE 9 — In Table 1, "Valid datalink file control options", the write permission option ADMIN is an abbreviation for both ADMIN REQUIRING TOKEN FOR UPDATE and ADMIN NOT REQUIRING TOKEN FOR UPDATE.

The default value of a site whose declared type is DATALINK is the null value. Datalinks are subject to certain restrictions. As a consequence of these restrictions, neither datalinks nor expressions whose declared type is DATALINK-ordered can appear in the following (among other places).

— <comparison predicate>.

— <general set function>.

— <group by clause>.

— <order by clause>.

— <unique constraint definition>.

— <referential constraint definition>.

— <select list> of a <query specification> that has a <set quantifier> of DISTINCT.

— <select list> of an operand of UNION, INTERSECT, and EXCEPT.

— Columns used for matching when forming a <joined table>.

The implementation-defined (IL018) *maximum datalink length* determines the amount of space, in octets, that is allocated for the following objects.

— A host variable of data type DATALINK.

— An argument of declared type DATALINK to an invocation of an external routine.

— The value returned by an invocation of an external function whose result type is DATALINK.

The maximum datalink length constrains the values of expressions whose declared type is DATALINK such that every such value can be assigned to a host variable, substituted for a parameter to an external routine, or returned by an invocation of an external function.

4.18.2 Operations involving datalinks

4.18.2.1 Operators that operate on datalinks

<url complete expression> returns the File Reference of a given datalink, possibly combined with a read token.

<url complete for write expression> returns the File Reference of a given datalink, possibly combined with a write token.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

<url complete only expression> returns the File Reference, excluding any access token, of a given datalink.

<url path expression> returns the path, including any read token, of the File Reference of a given datalink.

<url path for write expression> returns the path, including any write token, of the File Reference of a given datalink.

<url path only expression> returns the path, excluding any access token, of the File Reference of a given datalink.

<url scheme expression> returns the scheme of the File Reference of a given datalink.

<url server expression> returns the host of the File Reference of a given datalink.

NOTE 10 — "host", "scheme", and "path" are defined in Subclause 6.6, "<datalink value function>".

4.18.2.2 Other operators involving datalinks

A <datalink value constructor> specifies either DLVALUE, DLNEWCOPY, or DLPREVIOUSCOPY. DLVALUE returns a datalink value, given only a File Reference, returns the corresponding datalink. DLNEWCOPY and DLPREVIOUSCOPY return a datalink value, given a File Reference and an indication of whether the File Reference includes a write token.

The datalink value returned by DLNEWCOPY indicates to the SQL-server that the content of the file, referenced by that datalink, is different (i.e., the content has changed, but not the URL) from what was previously referenced by the datalink.

The datalink value returned by DLPREVIOUSCOPY indicates to the SQL-server that the content of the file might have changed, but the application is not interested in maintaining the changed file. The original file is restored in an implementation-dependent (UW008) fashion.

4.19 Foreign-data wrapper interface

4.19.1 Introduction to foreign-data wrapper interface

A foreign-data wrapper interface consists of the signatures of the routines that make up a given foreign-data wrapper. These routines serve the following purposes.

— Allocate and deallocate resources.

— Control connections to foreign servers.

— Receive data from the SQL-server about the foreign server request to be executed at the foreign server.

— Send data from the foreign server to the SQL-server about the foreign server request that the foreign server is willing to execute.

— Initiate and terminate the execution of foreign server requests by the foreign server.

4.19.2 Handles

A *handle* is a value of INTEGER data type that identifies an allocated resource that provides session state information about a foreign server, a foreign-data wrapper, or a foreign server session of interest to connected components of that session. Handles presented as arguments to invocations of foreign-data wrapper interface routines enable the invoker to give or obtain the information they reference. The handle of a particular resource is allocated by the keeper of the state information — either the foreign-data wrapper or the SQL-server — to enable the SQL-server or the foreign-data wrapper, respectively, to access that state information. Although the declared type for a handle is INTEGER, its value has no meaning in any other context and should not be used as a numeric operand or modified in any way.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

The following are the handles specified in the foreign-data wrapper interface, presented in approximately the order in which they are materialized in Table 2, "Sequence of actions during the execution of foreign server requests". The operations that cause their creation and destruction are given in Table 2, "Sequence of actions during the execution of foreign server requests".

— **WrapperEnv handle**: This handle is allocated by a foreign-data wrapper to reference information during the interaction with the SQL-server. It identifies an *allocated FDW-environment* and is allocated via a call from the SQL-server to the `AllocWrapperEnv()` routine. This handle shall be allocated before any foreign server requests are made to a foreign-data wrapper. It remains valid until the SQL-server invokes `FreeWrapperEnv(WH)`, where *WH* is the handle in question.

— **Server handle**: This handle is allocated by the SQL-server to reference a foreign server. A foreign-data wrapper uses this handle to obtain information about a foreign server to which it needs to connect. Routines associated with a server handle allow information to be obtained about such things as the server name, server type, server version, etc. This handle is allocated implicitly and presented by the SQL-server to a foreign-data wrapper by invoking `ConnectServer()`.

— **FSConnection handle**: This handle is allocated by a foreign-data wrapper to reference information about a foreign server session. It is allocated via a call to the `ConnectServer()` routine. This handle shall be allocated before any foreign server requests to be executed during that foreign server session are presented to a foreign-data wrapper.

— **Query Context handle**: This handle is allocated by the foreign-data wrapper to reference information that spans multiple foreign server requests. The SQL-server uses it to indicate to the foreign-data wrapper that identical value expression handles in the same query context (denoted by the same Query Context handle) but in different foreign server requests (represented by different Request handles) represent identical value expressions. It is foreign-data wrapper implementation-dependent (UA012) whether the foreign-data wrapper uses this information to re-use the previously evaluated value expression or whether the foreign-data wrapper re-evaluates the value expression. The handle remains valid until the SQL-server invokes `FreeQueryContext()`.

— **Request handle**: This handle is allocated by the SQL-server to reference a foreign server request that is to be executed by a foreign server. A request handle may reference a simple foreign server request, such as `SELECT * FROM T`, or it may reference a complex foreign server request that includes predicates, joins, ordering, etc. A request handle is used by the foreign-data wrapper to retrieve (for example) the names of foreign tables referenced in the from clause, the names of column references in the select list, etc., using foreign-data wrapper interface SQL-server routines. This handle is allocated implicitly.

— **Table Reference handle**: This handle is allocated by the SQL-server to reference a <table reference> contained in the <from clause> of a <query specification>. This handle is allocated implicitly.

— **Value Expression handle**: This handle is allocated by the SQL-server to reference a <value expression> contained in a foreign server request. This handle is allocated implicitly.

— **Reply handle**: This handle is allocated by a foreign-data wrapper to reference the subset of foreign server requests it is capable of executing. This handle is allocated via a call during a foreign server session to the `InitRequest()` routine and remains valid in that foreign server session until it is the argument to an invocation of `FreeReplyHandle()`.

— **Execution handle**: This handle is allocated by a foreign-data wrapper. In decomposition mode, it is used to reference the information the foreign-data wrapper needs to process the foreign server request referenced by the corresponding reply handle and the information associated with the data resulting from the processing of the foreign server request. This handle is allocated via a call to the `InitRequest()` routine, which sets the associated PASSTHROUGH flag to *False*. In pass-through mode, this handle is used to reference the information that the foreign-data wrapper needs to process the foreign server request that is sent to the foreign server. This handle is allocated via a call to the `TransmitRequest()` routine, which sets the associated PASSTHROUGH flag to *True*.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

NOTE 11 — "decomposition mode" and "pass-through mode" are defined in Subclause 4.19.4.6, "Decomposition and pass-through modes".

— **Wrapper handle**: This handle is allocated implicitly by the SQL-server to reference the information about a foreign-data wrapper.

— **User handle**: This handle is allocated implicitly by the SQL-server to reference information about the user on whose behalf a connection to a foreign server is being made.

— **Descriptor handle**: This handle is allocated by either the SQL-server or a foreign-data wrapper to reference a foreign-data wrapper descriptor area.

— **Routine Mapping handle**: This handle is implicitly allocated by the SQL-server to reference an allocated routine mapping description.

4.19.3 Foreign server sessions

A *foreign server session* is the sequence of operations performed by a foreign-data wrapper on a particular FSConnection handle during the existence of that handle.

A foreign server session on FSConnection handle *FSCH* begins with the invocation of `ConnectServer()` that brings *FSCH* into existence and ends with the invocation of `FreeFSConnection(`*`FSCH`*`)`.

4.19.4 Foreign-data wrapper interface routines

4.19.4.1 Introduction to foreign-data wrapper interface routines

The terms *foreign-data wrapper interface SQL-server routine* and *foreign-data wrapper interface wrapper routine* are used to distinguish routines provided by the SQL-server from routines provided by a foreign-data wrapper, respectively. A foreign-data wrapper interface routine that is both an SQL-server routine and a wrapper routine is referred to as a *foreign-data wrapper interface general routine*.

The foreign-data wrapper interface routines of a given SQL-implementation are either all functions or all procedures, the choice being implementation-defined (IA099). They are functions if their return codes are values returned by their invocations and they are procedures if their return codes are instead assigned to an output parameter named ReturnCode. The specific terms *foreign-data wrapper interface function* and *foreign-data wrapper interface procedure* are used when it is necessary to distinguish between the two kinds.

4.19.4.2 Handle routines

— **GetServerName**: This routine returns the name of a foreign server given a server handle.

— **GetServerType**: This routine returns the type of a foreign server given a server handle.

— **GetServerVersion**: This routine returns the version of a foreign server given a server handle.

— **GetNumServerOpts**: This routine returns the number of generic options associated with a foreign server given a server handle.

— **GetServerOpt**: This routine returns the generic option name and its value given a server handle and the position of the option in the options list.

— **GetServerOptByName**: This routine returns the generic option value given a server handle and the name of the option.

— **GetNumTableRefElems**: This routine returns the number of <table reference>s in the <from clause> of a query given a request handle.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— **GetTableRefElem**: This routine returns the table reference handle of a <table reference> in the <from clause> of a query given a request handle and the position of the <table reference> in the <from clause>.

— **GetTableRefElemType**: This routine returns the "type" of a <table reference> given the table reference handle. The only possible return value is TABLE_NAME.

— **GetTableRefTableName**: This routine returns the table name given a table reference handle.

— **GetNumSelectElems**: This routine returns the number of <value expression>s in the <select list> of a <query specification> given a request handle.

— **GetSelectElem**: This routine returns the value expression handle of a <value expression> in the <select list> of a <query specification> given a request handle and the position of the <value expression> in the <select list>.

— **GetSelectElemType**: This routine returns the kind of a <value expression> given the value expression handle. Possible return values are COLUMN_NAME, OPERATOR, PARAMETER, and CONSTANT.

— **GetValExprColName**: This routine returns the name of the column, given a value expression handle.

— **GetNumReplyTableRefs**: This routine returns the number of table references from the original foreign server request that the foreign-data wrapper is capable of accessing, given a reply handle.

— **GetReplyTableRef**: This routine returns the number of the table reference in the original request that the foreign-data wrapper is capable of accessing, given a reply handle and a number that ranges from 1 (one) to the value returned by the `GetNumReplyTableRefs()` routine.

— **GetNumReplySelectElems**: This routine returns the number of select list elements from the original foreign server request that the foreign-data wrapper is capable of accessing, given a reply handle.

— **GetReplySelectElem**: This routine returns the number of the select list element in the original foreign server request that the foreign-data wrapper is capable of accessing, given a reply handle and a number that ranges from 1 (one) to the value returned by the `GetNumReplySelectElems()` routine.

— **GetNumReplyBoolVE**: This routine returns the number of <boolean value expression>s simply contained in the <where clause> of the original foreign server request that the foreign-data wrapper is capable of handling, given a reply handle.

— **GetReplyBoolVE**: This routine returns the number of a <boolean value expression> element from the <where clause> in the original foreign server request that the foreign-data wrapper is capable of handling, given a reply handle and a number that ranges from 1 (one) to the value returned by the `GetNumReplyBoolVE()` routine.

— **GetReplyDistinct**: This routine returns information identifying whether the foreign-data wrapper is capable of providing distinct rows in the result set, given a reply handle.

— **GetReplyCardinality**: This routine returns an estimate of the cardinality of the result set associated with the reply, given a reply handle.

— **GetReplyFirstCost**: This routine returns a value that represents the estimated cost to retrieve the first row of the result set associated with the reply, given a reply handle. Larger values represent greater costs.

— **GetReplyExecCost**: This routine returns a value that represents the estimated cost to retrieve the result set associated with the reply, given a reply handle. Larger values represent greater costs.

— **GetReplyReExecCost**: This routine returns a value that represents the estimated cost to re-execute the reply, given a reply handle. Larger values represent greater costs.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— **GetNumReplyOrderBy**: This routine returns the number of columns that are used to order the result that the foreign-data wrapper is capable of handling, given a reply handle.

— **GetReplyOrderElem**: This routine returns the number of a <value expression> from the <select list> used to order the result in the original foreign server request that the foreign-data wrapper is capable of handling, given a reply handle and a number that ranges from 1 (one) to the value returned by the `GetNumReplyOrderBy()` routine.

— **GetNextReply**: This routine returns a new reply handle and execution handle for the original foreign server request, given a reply handle.

— **GetNumBoolVE**: This routine returns the number of <boolean value expression>s simply contained in the <where clause> of a <query specification>, given a request handle.

— **GetBoolVE**: This routine returns a handle for a <boolean value expression> from the <where clause> of a <query specification>, given a request handle and a number that ranges from 1 (one) to the value returned by the `GetNumBoolVE()` routine.

— **GetDistinct**: This routine returns information whether the query specifies DISTINCT or ALL, given a request handle.

— **GetNumOrderByElems**: This routine returns the number of columns that are used to order the result, given a request handle.

— **GetOrderByElem**: This routine returns a handle for a <value expression> used to order the result of a query, given a request handle and a number that ranges from 1 (one) to the value returned by the `GetNumOrderByElems()` routine.

— **GetValueExpKind**: This routine returns the kind of a <value expression>, given a value expression handle. Possible return values are COLUMN_NAME, OPERATOR, PARAMETER, CONSTANT.

— **GetNumChildren**: This routine returns the number of <value expression>s immediately contained in the containing <value expression>, given a value expression handle.

— **GetVEChild**: This routine returns a handle for a <value expression> immediately contained in the containing <value expression>, given a value expression handle and a number that ranges from 1 (one) to the value returned by the `GetNumChildren()` routine.

— **GetValueExpName**: This routine returns the name associated with a <value expression>, given a value expression handle.

— **GetValueExpTable**: This routine returns a table reference handle with which the table associated with the <value expression> is associated.

— **GetValueExpDesc**: This routine returns a handle for a value expression descriptor describing a <value expression>.

— **GetAuthorizationId**: This routine returns the authorization identifier associated with a user mapping, given a user handle.

— **GetTableColOpt**: This routine returns the generic option name and its value, given a table reference handle, column name and the position of the option in the options list.

— **GetTableColOptByName**: This routine returns the generic option value, given a table reference handle, a column name and the name of the option.

— **GetTableOpt**: This routine returns the generic option name and its value, given a table reference handle and the position of the option in the options list.

— **GetTableOptByName**: This routine returns the generic option value, given a table reference handle and the name of the option.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— **GetTableServerName**: This routine returns the name of the foreign server associated with a foreign table, given a table reference handle.

— **GetNumTableColOpts**: This routine returns the number of generic options associated with a column of a foreign table, given a table reference handle and a column name.

— **GetNumTableOpts**: This routine returns the number of generic options associated with a foreign table, given a table reference handle.

— **GetNumUserOpts**: This routine returns the number of generic options associated with a user mapping, given a user handle.

— **GetNumWrapperOpts**: This routine returns the number of generic options associated with a foreign-data wrapper, given a wrapper handle.

— **GetUserOpt**: This routine returns the generic option name and its value, given a user handle and the position of the option in the options list.

— **GetUserOptByName**: This routine returns the generic option value, given a user handle and the name of the option.

— **GetWrapperLibraryName**: This routine returns the name of the library associated with a foreign-data wrapper, given a wrapper handle.

— **GetWrapperName**: This routine returns the name of a foreign-data wrapper, given a wrapper handle.

— **GetWrapperOpt**: This routine returns the generic option name and its value, given a wrapper handle and the position of the option in the options list.

— **GetWrapperOptByName**: This routine returns the generic option value, given a wrapper handle and the name of the option.

— **GetDescriptor**: This routine, given a descriptor handle and the identification of a descriptor area field, retrieves the value of the specified field from a descriptor area.

— **SetDescriptor**: This routine, given a descriptor handle, the identification of a descriptor area field, and a new value to be assigned to that field, sets the value of the specified field of a descriptor area.

— **GetSPDHandle**: This routine returns the SPDHandle given an ExecutionHandle.

— **GetSRDHandle**: This routine returns the SRDHandle given an ExecutionHandle.

— **GetTRDHandle**: This routine returns the TRDHandle given a TableReferenceHandle.

— **GetWPDHandle**: This routine returns the WPDHandle given an ExecutionHandle.

— **GetWRDHandle**: This routine returns the WRDHandle given an ExecutionHandle.

— **GetSQLString**: This routine returns a character string representation of the query that is associated with the request handle.

— **GetRoutineMapping**: This routine returns the routine mapping handle for an allocated routine mapping description, given a value expression handle.

— **GetRoutMapOptName**: This routine returns the generic option value, given the routine mapping handle and the name of the option.

— **GetRoutMapOpt**: This routine returns the generic option name and its value, given a routine mapping handle and the position of the option in the option list.

— **GetNumRoutMapOpts**: This routine returns the number of generic options associated with a routine mapping, given a routine mapping handle.

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

4.19.4.3 Initialization routines

— **AdvanceInitRequest**: This routine is used by the SQL-server to cause a foreign server request to be prepared. The routine has three input parameters: a previously allocated FSConnection handle, a previously allocated QueryContext handle, and a request handle that describes the foreign server request. The routine has two output parameters: a reply handle that describes how much of the foreign server request the foreign-data wrapper is willing to handle, and an execution handle to the state information the foreign-data wrapper needs to process the foreign server request and the data rows that will be returned. This routine uses the `InitRequest()` routine multiple times to generate multiple reply handle/execution handle pairs. Additional reply handle/execution handle pairs can be retrieved by the SQL-server using the `GetNextReply()` routine.

— **AllocDescriptor**: This routine is used by the foreign-data wrapper to request that the SQL-server allocate a foreign-data wrapper descriptor area for use in exchanging information about values required to execute a foreign server request or values expected to be returned from an execution of a foreign server request.

— **AllocQueryContext**: This routine is used by the SQL-server to retrieve a Query Context handle that the SQL-server will use to indicate to the foreign-data wrapper that identical Value Expression handles in different foreign server requests refer to identical value expressions.

— **AllocWrapperEnv**: This routine is used by the SQL-server to allow the foreign-data wrapper to perform any initialization steps and allocate and initialize any necessary global data structures. It has a single input parameter, a WrapperHandle, that describes the information about the foreign-data wrapper maintained by the SQL-server, and a single output parameter, a WrapperEnv handle, which is a handle to the foreign-data wrapper's newly initialized global data structures.

— **ConnectServer**: This routine is used by the SQL-server to request access to a foreign server, and allows the foreign-data wrapper associated with that foreign server to establish a connection (if necessary) and set up any required state information. ConnectServer has three input parameters: a previously allocated WrapperEnv handle, a server handle that describes the foreign server for which the SQL-server is requesting a connection, and a UserHandle that describes the user mapping maintained by the SQL-server. The routine has one output parameter, the newly allocated FSConnection handle for the foreign server.

— **GetOpts**: This routine is used by the SQL-server to request that the foreign-data wrapper return information about the capabilities and other aspects of the foreign-data wrapper, the foreign server, some foreign table at the foreign server, or some foreign column of some foreign table at the foreign server. This routine is invoked by the SQL-server, and executed by the foreign-data wrapper, whenever the SQL-server requires information about options supported by the foreign-data wrapper and the foreign server. It is thus invoked under implementation-dependent (UA033) circumstances.

— **InitRequest**: This routine is used to generate a reply handle and an execution handle for a given foreign server request. The routine has two input parameters: a previously allocated FSConnection handle, and a request handle that describes the foreign server request. The routine has two output parameters: a reply handle that describes how much of the foreign server request the foreign-data wrapper is willing to handle, and an execution handle to the state information the foreign-data wrapper needs to process the foreign server request and the data rows that will be returned.

4.19.4.4 Access routines

— **Open**: This routine is used by the SQL-server to allow the foreign-data wrapper to allocate any resources necessary to perform the operations represented by the ExecutionHandle (and described by a ReplyHandle previously returned by an invocation of either `AdvanceInitRequest()` or `GetNextReply()`). The routine has one input parameter: a previously allocated ExecutionHandle.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

— **Iterate**: This routine is used by the SQL-server to iteratively retrieve data from a foreign-data wrapper. The routine has one input parameter, a previously allocated ExecutionHandle. As a result of this call, the foreign-data wrapper will associate the row with the ExecutionHandle. The SQL-server may invoke this routine until all data is returned.

— **ReOpen**: This routine may be used by the SQL-server to allow a foreign-data wrapper to re-initialize any resources necessary to re-execute the operations represented by the ExecutionHandle (and described by a ReplyHandle previously returned by either the `AdvanceInitRequest()` routine or the `GetNextReply()` routine). This routine allows an SQL-server to re-execute the operations associated with an ExecutionHandle multiple times, for example, if the work to be done by the foreign-data wrapper represents the inner node of a join being processed by the SQL-server. The routine has one input parameter: a previously allocated ExecutionHandle.

— **Close**: This routine is used by the SQL-server to allow a foreign-data wrapper to free any resources that had been allocated to perform the operations represented by the ExecutionHandle. The SQL-server invokes this routine after it is done processing a foreign server request that initiated the communication with the foreign-data wrapper. The routine has one input parameter: a previously allocated ExecutionHandle.

— **GetStatistics**: This routine is used by the SQL-server to request statistics, if any, related to the foreign server request previously sent to the foreign-data wrapper. Such statistics are entirely implementation-defined in nature. This routine is invoked by the SQL-server, and executed by the foreign-data wrapper, whenever the SQL-server requires statistics that may be provided by the foreign-data wrapper and the foreign server. It is thus invoked under implementation-dependent (UA034) circumstances.

— **TransmitRequest**: This routine is used by the SQL-server to transmit a foreign server request in the native language of the foreign server to the foreign server in pass-through mode. The foreign server analyzes the transmitted foreign server request and returns information about that foreign server request to the SQL-server. This information includes: Whether the foreign server request requires one or more input values in order to be executed; and whether the foreign server request returns one or more result values upon execution. This information is associated with the descriptors attached to the execution handle that is the output parameter of this routine. This routine has three input parameters: a previously allocated FSConnection handle, a string containing the foreign server request, and an integer indicating the string length.

4.19.4.5 Termination routines

— **FreeDescriptor**: This routine is used by the foreign-data wrapper to request that the SQL-server deallocate a foreign-data wrapper descriptor area and to free the memory and other resources used by that descriptor.

— **FreeExecutionHandle**: This routine is used by the SQL-server to allow the foreign-data wrapper to free the resources associated with an ExecutionHandle after the SQL-server has determined that it no longer needs the information encapsulated by the ExecutionHandle. This routine has one input parameter, a previously allocated ExecutionHandle.

— **FreeFSConnection**: This routine is used by the SQL-server to terminate a connection to a foreign server. It allows the foreign-data wrapper to disconnect from the foreign server and to free any resources associated with the connection, such as the FSConnection handle. It has one input parameter: a previously allocated FSConnection handle.

— **FreeQueryContext**: This routine is used by the SQL-server to allow the foreign-data wrapper to free all resources it may have associated with a Query Context handle. This routine has one input parameter, a previously allocated Query Context handle.

— **FreeReplyHandle**: This routine is used by the SQL-server to allow the foreign-data wrapper to free the resources associated with a ReplyHandle after the SQL-server has determined that it no

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

longer needs the information encapsulated by the ReplyHandle. This routine has one input parameter, a previously allocated ReplyHandle.

— **FreeWrapperEnv**: This routine is used by the SQL-server to terminate communication with a foreign-data wrapper. It allows the foreign-data wrapper to free any global resources it had allocated, such as the WrapperEnv handle. The routine has one input parameter, a previously allocated WrapperEnv handle.

4.19.4.6 Decomposition and pass-through modes

Depending on whether the pass-through flag in the current SQL-session context is set to *True* or *False*, the SQL-server is said to be either in *pass-through mode* or *decomposition mode*. When the SQL-server is in decomposition mode, the SQL-server analyzes the SQL-client request and invokes the `AdvanceInitRequest()` routine to communicate foreign server request to the foreign-data wrapper. When the SQL-server is in pass-through mode, the SQL-server does not analyze the SQL-client request and invokes the `TransmitRequest()` routine to communicate foreign server request to the foreign-data wrapper.

4.19.4.7 Sequence of actions during the execution of foreign server requests

For decomposition mode, Table 2, "Sequence of actions during the execution of foreign server requests", shows the sequence of actions as described by the General Rules of Subclause 7.1, "<table reference>", when a foreign table is identified by the <table name> simply contained in the <table reference>.

For pass-through mode, Table 2, "Sequence of actions during the execution of foreign server requests", shows the sequence of actions that is likely to occur when an SQL-client requests the preparation and execution of statements using dynamic SQL.

Table 2 — Sequence of actions during the execution of foreign server requests

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
1	Receives a query from SQL-client that involves data from a foreign table in a <table reference>. Determines relationship of foreign table → foreign server → foreign-data wrapper.	Receives from the SQL-client a statement to be executed in pass-through mode that involves one foreign server. Determines relationship of the foreign server to the foreign-data wrapper.			
2	Creates a **WrapperHandle** and associates information about the foreign-data wrapper with that handle.				
3	Invokes the **AllocWrapperEnv (WrapperHandle, WrapperEnvHandle)** routine to initialize the foreign data wrapper.		⇒		

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
4			⇐	Invokes the **Get... (WrapperHandle, ...)** routines in the SQL-server to retrieve information about the foreign-data wrapper that the SQL-server has stored in its Information Schema.	
NOTE 12 — This information is provided in the <foreign-data wrapper definition>.					
5	Executes the **Get... (WrapperHandle, ...)** routines as requested from the foreign-data wrapper.		⇒		
6				Allocates global data structures associated with the wrapper and associates them with the **WrapperEnvHandle** and performs any initialization required.	
7	Frees the **WrapperHandle**.				
NOTE 13 — If the current SQL-session context already includes a {foreign-data wrapper name : WrapperEnvHandle} pair that could be used, then Step 2 through Step 7 are optional.					
8	Creates a **ServerHandle** and associates information about the foreign server with that handle.				
9	Creates a **UserHandle** and associates information about the current user with that handle.				
10	Invokes the **ConnectServer (WrapperEnvHandle, ServerHandle, UserHandle, FSConnectionHandle)** routine in the foreign-data wrapper to establish a connection to the foreign server that contains some of the data required to process the SQL-server client's query.		⇒		
11			⇐	Invokes the **Get... (ServerHandle,...)** routines in the SQL-Server to retrieve all information about the foreign server that the SQL-server has stored in its Information Schema.	
NOTE 14 — This information is provided in the <foreign server definition>.					
12	Executes the **Get... (ServerHandle, ...)** routines as requested from the foreign data wrapper.		⇒		

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
13			⇐	Invokes the **Get... (UserHandle, ...)** routines in the SQL-Server to retrieve all information about the user that the SQL-server has stored in its Information Schema.	
NOTE 15 — This information is provided in the <user mapping definition>.					
14	Executes the **Get... (UserHandle, ...)** routines as requested from the foreign data wrapper.		⇒		
15				Allocates global data structures associated with the foreign server and associates them with a **FSConnectionHandle**, establishes a connection to the foreign server, and performs any initialization required.	
16	Frees the **ServerHandle**.				
17	Frees the **UserHandle**.				
NOTE 16 — If the current SQL-session context already includes a {foreign server name : FSConnectionHandle} pair that could be used, then Step 8 through Step 17 are optional.					
18	Invokes the **AllocQueryContext()** routine in the foreign-data wrapper to obtain a **QueryContextHandle**.		⇒		
19				Allocates data structures to utilize the information about the query context and associates them with the **QueryContextHandle**.	
20	Creates a **RequestHandle** and associates with it the information about the part of query that could be handled by the foreign-data wrapper.				

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
21	Creates as many **TableReference-Handle**s as are needed and associates with each of them the information about a particular <table reference>.				
22	Creates a **TableReference-Descriptor (TRD)** for the result of the <query specification> described by the **RequestHandle**, and sets all the fields with details about each of the columns.				
23	Creates a **Wrapper-Parameter-Descriptor (WPD)** to describe the <dynamic parameter specification>s in the <query specification> described by the **RequestHandle**, and sets all the fields with details about each of the <dynamic parameter specification>s.				

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
24	Creates as many **ValueExpressionHandle**s as are needed and associates with each of them the information about a particular <value expression> in the <select list> and <where clause>, respectively.				
25	Invokes the **AdvanceInitRequest (FSConnectionHandle, RequestHandle, ReplyHandle, ExecutionHandle, QueryContextHandle)** routine in the foreign-data wrapper to find out how much of the request the foreign server can actually process.	Invokes the **TransmitRequest (FSConnectionHandle, RequestString, StringLength, ExecutionHandle)** routine to allow the foreign-data wrapper and the foreign server to analyze the foreign server request.	⇒		
26			⇐	Invokes the **Get... (RequestHandle, ...)**, **GetTRDHandle (TableReferenceHandle)**, **GetDescriptor (TRD)**, **Get... (TableReferenceHandle, ...)**, and **Get... (ValueExpressionHandle, ...)** routines in the SQL-server to examine the SQL-server's request and to determine how much of the request the foreign-data wrapper can handle.	Executes the **TransmitRequest ()** routine.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
NOTE 17 — The foreign-data wrapper could invoke the **GetSQLString (RequestHandle, StringFormat, SQLString, BufferLength, StringLength)** routine in the SQL-server to examine the foreign server request and to determine how much of the foreign server request the foreign-data wrapper can handle, instead of the **Get... (RequestHandle, ...)**, **Get... (TableReferenceHandle, ...)**, **Get... (ValueExpressionHandle, ...)**, **GetTRDHandle (TableReferenceHandle)**, and **GetDescriptor (TRD)** routines.					
27	Executes the **Get... (RequestHandle, ...)**, **GetTRDHandle (TableReferenceHandle)**, **GetDescriptor (TRD)**, **Get... (TableReferenceHandle, ...)**, and **Get... (ValueExpressionHandle, ...)** routines as requested by the foreign-data wrapper.		⇒		
NOTE 18 — If the foreign-data wrapper invoked the **GetSQLString (RequestHandle, StringFormat, SQLString, BufferLength, StringLength)** routine, then this routine is executed instead of the **Get... (RequestHandle, ...)**, **Get... (TableReferenceHandle, ...)**, **Get...(ValueExpressionHandle, ...)**, **GetTRDHandle(TableReferenceHandle)**, and **GetDescriptor (TRD)** routines.					
28				Creates a **ReplyHandle** and associates with it the information about the part of query that could actually be handled by the foreign-data wrapper.	
29			⇐	Creates an **ExecutionHandle** and associates with it the information about the actual execution plan.	

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
30			⇐	Invokes the **AllocDescriptor ()** routine to create two descriptors (SRD and SPD) and associates both of them with the **ExecutionHandle**. Initializes the descriptors with default values wherever applicable.	Invokes the **AllocDescriptor ()** routine to create two descriptors (WRD and SRD) to describe the columns of the result table and associates both of them with the **ExecutionHandle**. Initializes both the descriptors with default values wherever applicable. Sets the fields in the WRD to correspond to the result columns associated with the **ExecutionHandle**.
31	Executes the **AllocDescriptor ()** routine as requested by the foreign-data wrapper.		⇒		
32			⇐		Invokes the **AllocDescriptor ()** routine to create two descriptors, WPD and SPD, to describe <dynamic parameter specification>s. Associates both of them with the **ExecutionHandle**. Initializes both the descriptors with default values wherever applicable. Sets the fields in the WPD to correspond to the dynamic parameters associated with the **ExecutionHandle**.

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
33		Executes the **AllocDescriptor ()** routine as requested by the foreign-data wrapper.	⇒		
34				Repeats steps Step 25 through Step 33 to create multiple **ReplyHandle / ExecutionHandle** pairs for the same foreign server request.	
NOTE 19 — Step 34 is optional.					
35	Invokes the **Get...(ReplyHandle, ...)** routines in the foreign-data wrapper to incorporate the work that a wrapper can do into the execution plan for the SQL-server client's query.		⇒		
36			⇐	Executes the **Get...(ReplyHandle, ...)** routines as requested by the SQL-server.	
37	Invokes the **GetNextReply( )** routine to retrieve another **ReplyHandle** for the same foreign server request		⇒		
38			⇐	Executes the **GetNextReply()** routine and returns a **ReplyHandle** and **ExecutionHandle**.	
39	Repeats Step 35 through Step 38.				

© ISO/IEC 2023 – All rights reserved

Documents
Document Sharing Hub
Full document details: ISO IEC 09075-9-2023

Step	SQL-server		Flow	Foreign-data wrapper	
	Decomposition	**Pass-through**		**Decomposition**	**Pass-through**
NOTE 20 — Step 37 through Step 38 are optional.					
40	Invokes the **FreeReplyHandle (ReplyHandle)** routine in the foreign-data wrapper to indicate that the **ReplyHandle** is no longer required.		⇒		
41				Frees resources associated with **ReplyHandle**.	
42	Repeats steps Step 20 through Step 41.				
NOTE 21 — Step 42 is optional.					
43	Invokes the **GetSRDHandle (ExecutionHandle)** routine to get the SRD.	Invokes the **GetWRDHandle (ExecutionHandle)** and **GetSRDHandle (ExecutionHandle)** routines to get the WRD and the SRD, respectively.	⇒		
44			⇐	Executes the **GetSRDHandle (ExecutionHandle)** routine as requested by the SQL-server.	Executes the **GetWRDHandle (ExecutionHandle)** and **GetSRDHandle (ExecutionHandle)** routines as requested by the SQL-server.
45		Invokes the **GetDescriptor (WRD)** routine multiple times to get all the information associated with WRD.			
46	Invokes the **SetDescriptor (SRD)** routine multiple times to populate appropriate fields in SRD.				

© ISO/IEC 2023 – All rights reserved
Documents Document Sharing Hub
Full document details: ISO IEC 09075-9-2023