ISO/IEC JTC 1/SC 32 N 0369

Date: 1999-10-12

REPLACES: N0339

ISO/IEC JTC 1/SC 32

Data Management and Interchange

Secretariat: United States of America (ANSI)
Administered by Pacific Northwest National Laboratory on behalf of ANSI

DOCUMENT TYPE	Text for CD Ballot
TITLE	CD ISO/IEC 9579/Am2:200y (E) Information technology — Distribution Schema for Remote Database Access
SOURCE	John Hadjioannou - Editor
PROJECT NUMBER	31.04.02.00
STATUS	Please return all ballots to the SC 32 Secretariat no later than 12 January 2000
REFERENCES	
ACTION ID.	LB
REQUESTED ACTION	Please return all ballots to the SC 32 Secretariat no later than 12 January 2000
DUE DATE	2000-01-12
Number of Pages	51
LANGUAGE USED	English
DISTRIBUTION	P & L Members SC Chair WG Conveners and Secretaries

Douglas Mann, Secretariat, ISO/IEC JTC 1/SC 32

Pacific Northwest National Laboratory *, 901 D Street, SW., Suite 900, Washington, DC, 20024-2115, United States of America

Telephone: +1 703 575 2114; Facsimile; +1 703 681 9180; E-mail: MannD@battelle.org

*Pacific Northwest National Laboratory (PNL) administers the ISO/IEC JTC 1/SC 32 Secretariat on behalf of ANSI

ISO/IEC JTC1 SC32 N _____

SC32 WG5 N 0028r1

Title:	CD ISO/IEC 9579/Am2:200y (E) Information technology — Distribution Schema for Remote Database Access
Source:	SC32 WG5 meeting, Matsue, Japan, May 1999
Date:	8 October 1999
Requested Action:	Issue for CD Ballot

Editor's Preface:

This text was initially prepared under the BSI/DTI Consultancy Drafting Scheme project SP6/3.1(355), Distribution Schema for RDA in January 1999, it has subsequently been revised as a result of the meeting of SC32 in Matsue, Japan, May 1999.

The background for this work is in:

RDA-KAN-23 (WG3 N2012) Remote Database Access – Strategic Directions

Document Conventions

Informative Paragraphs

This document contains informative paragraphs that serve to clarify aspects of this International Standard, or which provide information about flexibility open to implementers of the protocol defined by this International Standard. Such paragraphs are introduced by the word "NOTE" followed by a number identifying the paragraph and are set out as in the following example:

NOTE 1 – This is an example of an informative note.

Discussion Paragraphs

This document contains informative paragraphs that serve to explain why certain design decisions have been taken to facilitate review by experts involved in the development of this International Standard. Such paragraphs are introduced by the word "DISCUSSION" followed by a number identifying the paragraph and are set out as in the following example:

DISCUSSION 1 – This is an example of a discussion note.

Discussion paragraphs will be deleted from this document prior to publication as an International Standard.

Incomplete Marker

This document contains markers that indicate that additional text is required, or existing text may need to be modified, prior to completion of this document.

Such markers appear with a surrounding box and a grey background as in the example below:

Example of Incomplete Marker

Incomplete markers will be satisfied and removed from this document prior to publication.

Editor:

John Hadjioannou
Blue Star Information Systems Ltd
197 Grasmere Way
Linslade
Leighton Buzzard
LU7 7QB

Tel: *+44 1525 374667*
Fax: *+44 1525 850427*

Email: *john@minster.co.uk*

NOTE TO REVIEWERS

Both the SC32 project relating to this work calls for the production of an Amendment.

It is not yet known whether this work will ultimately be progressed as (a) an amendment to 9579, (b) a new edition of 9579, or (c) a new independent standard.

This text has been prepared as a stand-alone document to ease review and progression.

Following discussion between SC32 WG5 and SC32 WG3 in Matsue, Japan, in May 1999 this document has been substantially revised in the following ways:

1. The mapping of remote data is now done at the Catalog level rather than the Table level. Remote SQL-catalogs are mapped (aliased) to SQL-catalogs at the distribution controller. This has resulted in significant simplification and permitted the elimination of the DDA-information Schema, many of the tables in the DDA-distribution Schema and the need for four-part names.
2. The purpose of the language constructs to manipulate the DDA-distribution Schema has been clarified.

The authentication and security related parts of this document will be aligned with the work on Extended Security as that project matures.

COMMITTEE DRAFT
INTERNATIONAL
STANDARD

ISO/IEC 9579/Am2

First Edition
200y-xx-xx

Information technology — Distribution Schema for Remote Database Access

Version: CD
Standard: ISO/IEC 9579/Am2:200y (E)
Date: 8 October 1999
Copyright:

Reference number
ISO/IEC 9579/Am2:200y (E)

Contents

Contents ii

Tables v

Figures vi

Foreword vii

Introduction viii

1 Scope 1

2 Normative References 2

3 Definitions, Conventions and Notations 3

3.1 Definitions 3

3.2 Conventions 4
3.2.1 Convention for Figures 4
3.2.2 Naming of Concepts 4
3.2.3 Specification of DDA elements 4
3.2.4 Evaluation of Rules 4

3.3 Notations 5

4 Model, Concepts and Facilities 6

4.1 Model 6
4.1.1 Processor Model 6
4.1.2 Privilege Model 7
4.1.3 Transaction Model 8
4.1.4 Security Model 8

4.2 DDA concepts 9
4.2.1 DDA-catalog 9
4.2.2 DDA-distribution Schema 9

4.2.3 DDA-distribution Statements 9
4.2.4 DDA-controller 9
4.2.5 DDA-mapped Tables 10
4.2.6 Limitations and Requirements 10
4.3 The mapping of SQL concepts 12

5 DDA-distribution Schema 13
5.1 Server Location Table 14
5.2 Catalog Location Table 15
5.3 Fragment Table 16
5.4 Replicate Table 18
5.5 ROLE_AUTHORIZATION_DESCRIPTORS base table 19
5.6 ROLES base table 20
5.7 SERVER AUTHENTICATION base table 21
5.8 USERS base table 22

6 DDA-distribution Statements 23
6.1 Distribution Schema Statements 23
6.2 DDA-distribution Schema Definition 24
6.3 DDA-distribution Schema Elements 26
6.3.1 Server Location Definition 26
6.3.2 Server Authentication 26
6.3.3 Mapped Table Definition 27

7 Exceptions 29
7.1 Exception codes for DDA-specific Conditions 29

8 Conformance 30
8.1 DDA Conformance 30
8.2 Claims of Conformance 30

Annex A Conformance Proforma 33
A.1. Identification 33
A.2. Supplier Details 33
A.3. Implementation Details 34
A.4. DDA Support 34

Annex B Extensions to RDA.....35

Annex C Examples of DDA architecture 37

C.1. Local SQL-servers 37

C.2. Remote SQL-servers.....37

C.3. Remote DDA-server 38

Tables

Table 1–SQLSTATE class and subclass values for DDA-specific conditions..... 29

Figures

Figure 1–Model of SQL-environment 6
Figure 2–Model of DDA-environment 6
Figure 3–Model of a DDA-server 7
Figure 4–Local SQL-servers 37
Figure 5–Remote SQL-servers 37
Figure 6–Remote DDA-server 38

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 mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 9579/Am2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 32, *Data management and interchange.*

This is the first edition of an International Standard addressing the Management and Operation of facilities to provide access to SQL-data, independent of its actual location. It addresses the requirements of geographically distributed enterprises that need to both maintain a level of local autonomy and provide a simple interface to multiple SQL-implementations managing SQL-data.

Annexes A to B form an integral part of this International Standard. Annex C is for information only.

Introduction

The Distributed Database Access for SQL International Standard is complementary to the standards for Database Language SQL and Remote Database Access for SQL. It addresses the need for access to a geographically dispersed set of SQL-implementations which may be under independent administration.

Distributed Database Access for SQL defines how multiple SQL-implementations, each providing selected SQL-data can be regarded as a composite SQL-implementation. The component SQL-data may be:

- at different sites,
- on different hardware systems,
- managed by different Database Management Systems,
- managed by their own Database Administrators.

Distributed Database Access for SQL supports the Database Language SQL and uses the Remote Database Access for SQL protocol for system interworking. Extensions to these that are required to support Distributed Database Access are defined in this International Standard.

Information technology — Distribution Schema for Remote Database Access

1 Scope

This International Standard, Distribution Schema for Remote Database Access (DDA), defines a model for the architecture and operation of a geographically distributed data management facility where each site contains SQL-implementations supporting SQL-catalogs, SQL-schemas and SQL-data. The sites are connected by communications facilities that may support the Remote Database Access protocol (RDA/SQL) protocol carried over a Transport Layer.

The architecture identifies a functional Distribution Controller, which provides data location independence and interface to SQL-data such that execution of a single SQL-statement by the user may involve manipulation of data from more than one SQL-implementation. The operation of the Distribution Controller is determined by a Distribution Schema, the DDA-distribution Schema, which identifies the SQL-data available from each participating SQL-implementation and specifies the mapping of this data to SQL-catalogs available at the Distribution Controller.

This International Standard relies upon the facilities provided by all parts of ISO/IEC 9075 (SQL) and ISO/IEC 9579 (RDA).

This International Standard defines:

- a model for distributed database access,
- the DDA-distribution Schema,
- DDA-distribution Statements, specified in the style of SQL, for defining and maintaining the DDA-distribution Schema

Normative annexes provide:

- a Conformance Proforma,
- amendments enhancing ISO/IEC 9579 (RDA)

DISCUSSION 1 – These amendments may be processed separately and removed from this document.

Informative annexes provide:

- a collection of examples of architectures that conform to this International Standard

This International Standard does not constrain:

- conforming implementations to use any particular processor decomposition,

DISCUSSION 2 – Specifically, implementers need not actually materialise a discrete DDA-controller.

This International Standard does not define:

- algorithms for query decomposition or for the combining of partial results,
- recovery mechanisms in the event that transaction co-ordination fails,
- mechanisms for the asynchronous replication of data

2 Normative References

The following International Standards contain provisions, which through reference in this text constitute provisions of this International Standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 9075:199x *Information technology – Database Languages SQL*

DISCUSSION 3 – It will be necessary to identify the specific SQL documents here.

ISO/IEC 9579:2000 *Information technology – Remote Database Access for SQL*

ISO/IEC 10032:1995 *Information technology – Reference Model of Data Management*

3 Definitions, Conventions and Notations

3.1 Definitions

For the purposes of this International Standard, the definitions given in the following International Standards apply:

– ISO/IEC 9075 (SQL),

DISCUSSION 4 – It will be necessary to identify the specific SQL documents here.

– ISO/IEC 9075-3 (SQL/CLI),

– ISO/IEC 9579 (RDA).

The following terms are defined in ISO/IEC 10032:

– Distribution Controller,

– Distribution Schema.

In addition, the following definitions apply:

Distributed Database: an SQL-implementation that presents the appearance of a single SQL-implementation whilst deriving its SQL-data from other SQL-implementations according to a Distribution Schema.

NOTE 1 – The term “Database” without qualification is not used in this International Standard since it is so commonplace as to be ambiguous.

3.2 Conventions

3.2.1 Convention for Figures

The convention used for figures is that defined in the Reference Model of Data Management, ISO/IEC 10032.

3.2.2 Naming of Concepts

Where a concept has been defined in ISO/IEC 9075 (SQL) and used in this International Standard, the name used in ISO/IEC 9075 (SQL) is used for that concept.

Where a concept has been defined in ISO/IEC 9579 (RDA) and used in this International Standard, the name used in ISO/IEC 9579 (RDA) is used for that concept.

Concepts whose name begins with 'DDA-' are defined in this International Standard. Concepts whose name begins with 'SQL-' are defined in ISO/IEC 9075 (SQL). Concepts whose name begins with 'RDA-' are defined in ISO/IEC 9579 (RDA).

3.2.3 Specification of DDA elements

The definition of each DDA element, depending on the nature of the element, may have the following parts:

- *function*, a short statement of the purpose of the element,
- *definition*, the definition of the element in SQL,
- *syntax*, the syntax of the element in BNF,
- *description*, a sequence of Rules describing the element,
- *syntax rules*, a sequence of Rules as defined in ISO/IEC 9075 (SQL) for Syntax Rules,
- *general rules*, a sequence of Rules as defined in ISO/IEC 9075 (SQL) for General Rules.

3.2.4 Evaluation of Rules

The evaluation of Descriptions, Syntax Rules and General Rules is as defined in ISO/IEC 9075 (SQL)

3.3 Notations

DISCUSSION 5 – There are no notations at present. This subclause is retained as a placemarker.

4 Model, Concepts and Facilities

4.1 Model

4.1.1 Processor Model

ISO/IEC 9075 (SQL) defines an SQL-environment, SQL-client, SQL-server and SQL-catalog. ISO/IEC 9579 (RDA) defines a Service User, RDA-client, RDA-server and Transport Provider. Figure 1 shows how these interrelate.

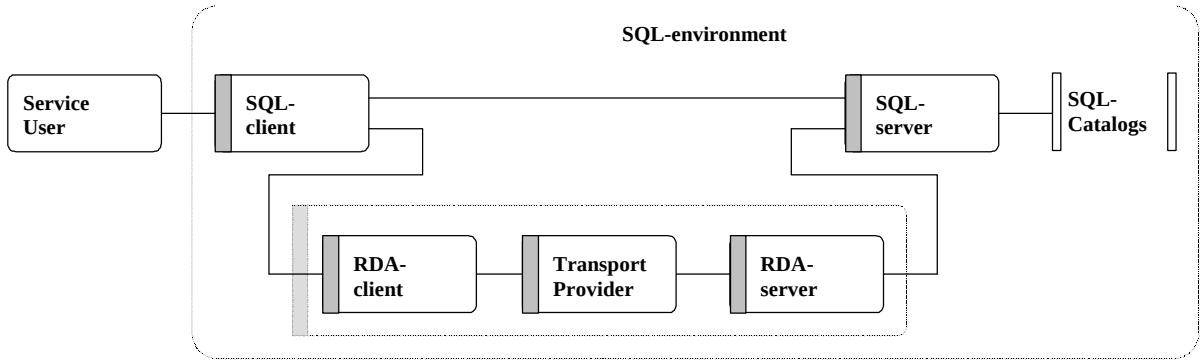

Figure 1 – Model of SQL-environment

This International Standard defines facilities for presenting multiple SQL-servers to a Service Provider in such a way that they appear as a single (composite) SQL-server with (composite) SQL-catalogs in a way described by a Distribution Schema. These are termed DDA-server and DDA-catalogs respectively, as shown in Figure 2.

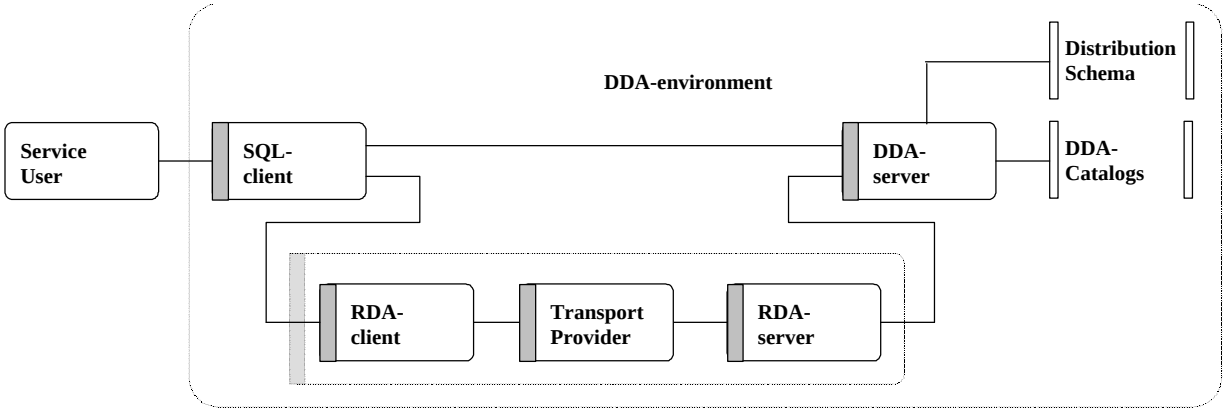

Figure 2 – Model of DDA-environment

A DDA-server is an SQL-server with an internal composition defined in this International Standard and modelled in Figure 3. Within a DDA-server, a DDA-controller acts as an SQL-client in respect to one or more component SQL-servers.

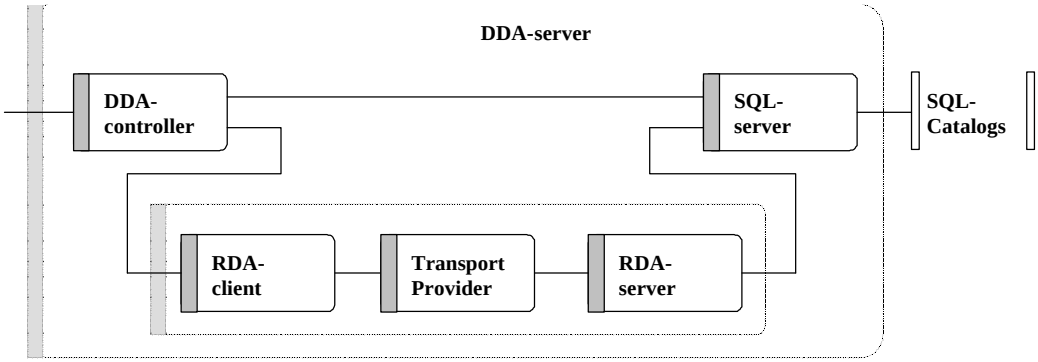

Figure 3 – Model of a DDA-server

An SQL-server included directly in a DDA-server is termed a *component* SQL-server where to refer to it as an SQL-server would be ambiguous.

The SQL-server components in Figure 3 may themselves be DDA-servers.

NOTE 2 – Thus, the definition of a DDA-server is recursive. A Distributed Database may therefore be composed of SQL-servers that are themselves Distributed Databases.

4.1.2 Privilege Model

DISCUSSION 6 – A detailed definition of how SQL-authorisation and RDA-authentication are used in a distributed environment is being developed. This subclause may be substantially revised as a result.

A DDA-server is an SQL-server with the privilege model and mechanisms defined in ISO/IEC 9075 (SQL).

DISCUSSION 7 – Reviewers to note that 9075 requires the creator of a schema be its owner and that only the owner can create new schema objects.

The DDA Privilege Model assumes a database administrator which is an instance of an RMDM Distribution Definer.

The DDA Privilege Model relates schema definition and ownership in the Distributed Database environment to the privilege models for the SQL Server on each site.

The Distributed Database database administrator is the owner of the DDA-information Schema and the DDA-distribution Schema. Users of a DDA-server are identified by their auth_ids. The database administrator may define roles and users may be privileged to play these roles.

Each component SQL-server has its own database administrator and its own privilege control. When an SQL-session is established with a component SQL-server, authorization information provided by the DDA-server is used. This gives access to public objects and to objects accessible for that authorization.

DISCUSSION 8 – Reviewers to note that 9075 does not support the notion of an SQL-server having a database administrator, although a Distributed Database may have one.

4.1.3 Transaction Model

DISCUSSION 9 – Nothing beyond SQL is needed at present. This subclause is retained as a placemarker.

NOTE 3 – Because of the semantics of SQL-transactions, all component SQL-implementations that participate in the evaluation of an SQL-statement by a DDA-server must be available for the evaluation to complete successfully.

4.1.4 Security Model

There is assumed to be total mutual trust either between the service user and DDA-services, or between DDA-services and the component SQL-servers.

DISCUSSION 10 – A detailed definition of how SQL-authorisation and RDA-authentication information is used across multiple connections defining how the authentication and access control mechanisms can take into account both the Service User and the DDA-services is being developed. Use of these enhanced facilities would enable the trust requirement to be relaxed.

DISCUSSION 11 – It may be necessary to say different things depending upon whether RDA is involved or not.

4.2 DDA concepts

4.2.1 DDA-catalog

A DDA-catalog is an SQL-catalog available through a DDA-server. The DDA-catalog is derived from the SQL-catalogs of the component SQL-servers.

4.2.2 DDA-distribution Schema

The DDA-catalogs of a DDA-server are derived from the SQL-catalogs of the component SQL-servers according to a mapping defined by data defined by an SQL-schema available to DDA-services termed the DDA-distribution Schema.

The DDA-distribution Schema describes data which:

- Identifies the location of component SQL-servers by reference to Transport Mapping and Transport Address,
- Identifies characteristics of component SQL-servers,
- Identifies the SQL-server that manages each element of SQL-data defined within the DDA-catalogs.

NOTE 4 – The DDA-distribution Schema may be itself distributed.

The DDA-distribution Schema is defined in clause 5

4.2.3 DDA-distribution Statements

An implementation may wish to inhibit a Service User from updating the DDA-distribution Schema directly. Statements for defining and maintaining the DDA-distribution Schema are defined in clause 6. These are in the style of ISO/IEC 9075 (SQL).

4.2.4 DDA-controller

A DDA-controller maps an SQL-session established between an SQL-client and a DDA-server to SQL-sessions between DDA-services and component SQL-servers.

A DDA- controller:

- Processes the operations that maintain the DDA-distribution Schema,
- Establishes SQL-sessions with component SQL-servers,
- Decomposes SQL-statements that address the component SQL-catalogs into SQL-statements and send these to the relevant component SQL-servers,
- Combines the results of component SQL-statements

NOTE 5 – The Decomposition of SQL-statements and the combination of associated results are not further defined by this International Standard.

An implementation of a DDA-controller may restrict the repertoire of SQL-statements that it may process.

DISCUSSION 12 – It may be appropriate to provide a "profiling" of these for conformance purposes – for example, permitting only <select statement>s may be a commercially useful and viable subset.

A DDA-controller is an instance of a Distribution Controller as defined in ISO/IEC 10032 (RMDM).

4.2.5 DDA-mapped Tables

DISCUSSION 13 – Is this definition required?

DDA-mapped tables are tables that are constructed from tables on more than one SQL-implementation. The SQL-implementations that contain parts of a mapped table are termed contributing SQL-implementations.

A mapped table may be partitioned into a set of disjoint fragments, such that any given row of the table is in one and only one fragment.

Fragments (or the whole table if the table is not fragmented) may be stored on more than one SQL-implementation, in which case each replicate shall have equivalent rows.

New rows are allocated to fragments according to a sequence of <insert condition>s defined for the fragments.

NOTE 6 – For database integrity there should be check conditions in the contributing SQL-implementation schemas to ensure that rows cannot be inserted locally into the wrong contributing SQL-implementation.

The wording of the above note needs to be reviewed, and perhaps promoted from a note to a requirement.

4.2.6 Limitations and Requirements

DISCUSSION 14 – These need to be kept under review. The language of this clause should be revised to align with the rest of this document and with SQL.

In any DDA-server there is at most one active SQL-connection between a DDA-controller any one SQL-server.

NOTE 7 – Providing a mapping from an arbitrary schema to a set of underlying dispersed schemas is inherently complex and the most general solution is probably too complex to be economically implemented. This International Standard aims to limit the complexity by specifying constraints that can reasonably be met when the underlying databases have the type of common features that would apply if they belong to a single Enterprise (or association) with data definition standards.

The integrity of the underlying component SQL-servers is managed by their own administrators, i.e. by the owners of the Schemas for the Databases extant on each SQL-implementation.

The local schema owner has full control over access to data and the subset of the data that is available to users of a distributed database.

Data in the distributed database may be derived from data in component databases according to derivation rules.

If Domain Names or UDTs are to be used for the DDB data then the definitions of these must be identical on all SQL-implementations. The DDB Administrator is responsible for ensuring that functions invoked exist and are correct on all the sites where they may be executed.

DISCUSSION 15 – More general "common universe of discourse" text may be better

DISCUSSION 16 – There is a requirement for database statistics but the method for gathering stats is both server and implementation dependent and hence may be different on each site, do we need to include some common mandatory elements or functions?

4.3 The mapping of SQL concepts

DISCUSSION 17 – There are no SQL concepts mapped at present. Reviewers are asked to comment as to whether concepts such as SQL-catalog, SQL-user name, SQL-role need to appear here.

DISCUSSION 18 – Is it appropriate to describe how INSERT, SELECT, UPDATE, DELETE work on mapped tables? Reviewers are asked to comment.

5 DDA-distribution Schema

The DDA-distribution Schema identifies the SQL-servers that participate in the Distributed Database and describes how the DDA-catalog is derived from the component SQL-catalogs.

```
CREATE SCHEMA DISTRIBUTION_SCHEMA
      AUTHORIZATION DISTRIBUTION_SCHEMA
```

DISCUSSION 19 – Is there a relationship between the Authorization DISTRIBUTION_SCHEMA and the Authorization RDA_SUPERVISOR of the base standard?

The DDA-distribution Schema contains:

- The mapping of DDA-catalogs to SQL-catalogs
- The description of how mapped tables are constructed from component table fragments
- Access Control information related to users of the Distributed Database.

The base tables are all defined in a <schema definition> for the schema named DISTRIBUTION_SCHEMA. The table definitions are as complete as the definitional power of SQL allows. Table definitions are supplemented with assertions where appropriate.

For stylistic reasons, the following text needs to be made consistent with the text in the base document

Each definition consists of three parts:

- The function of the definition is stated.
- The SQL definition of the object is presented.
- An explanation of the object.

The specification provides only a model of the base tables that are required, and does not imply that an implementation shall provide the functionality in the manner described in this clause.

DISCUSSION 20 – It may be appropriate to forbid self-reference and other circularities.

5.1 Server Location Table

Function

Identify SQL Servers known to the Distribution Controller and define access to them.

DISCUSSION 21 – This is an amended version of the SERVER_LOCATION table in the base standard. Should this be in the RDA Amendments annex?

Definition

```
CREATE TABLE SERVER_LOCATION
   (
   SERVER_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   TRANSPORT_MAPPING SMALLINT NOT NULL,
   TRANSPORT_ADDRESS INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   RDA_DESTINATION_SERVER_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   INFORMATION_SCHEMA_CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER NOT
NULL,
   CONSTRAINT SERVER_LOCATION_PRIMARY_KEY PRIMARY KEY (SERVER_NAME)
   )
```

Description

For each SQL-server *S* recorded in SERVER_LOCATION:

SERVER_NAME: the SQL-server name of *S* as known to the Service User.

TRANSPORT_MAPPING: the code identifying the Transport Mapping through which a Transport Connection to *S* can be established as defined in Table 17 of ISO/IEC 9579 (RDA).

TRANSPORT_ADDRESS: the Transport Address of *S* within the Transport Provider identified by TRANSPORT_MAPPING.

DISCUSSION 22 – Should TRANSPORT_ADDRESS in the base document be extended to URLs?

RDA_DESTINATION_SERVER_NAME: the Destination SQL-server name of *S* within the RDA-server environment identified by the values of TRANSPORT_MAPPING and TRANSPORT_ADDRESS taken together.

5.2 Catalog Location Table

Function

Maps SQL-catalogs within the DDA-server to SQL-catalogs of component SQL-servers.

Definition

```
CREATE TABLE CATALOG_LOCATION
   (
   CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   DDA_COMPONENT_SERVER_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   DDA_COMPONENT_CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   CONSTRAINT CATALOG_LOCATION_PRIMARY_KEY PRIMARY KEY (CATALOG_NAME)
   )
```

Description

For each SQL-catalog *C* recorded in CATALOG_LOCATION:

CATALOG_NAME: the SQL-catalog name of *C* as known to the Service User.

DDA_COMPONENT_SERVER_NAME: the SQL-server name the SQL-server *S* at which *C* resides.

DDA_COMPONENT_CATALOG_NAME: the SQL-catalog name of *C* within *S*.

5.3 Fragment Table

Function

Defines how mapped tables that are fragmented are constructed from component table fragments.

Definition

```
CREATE TABLE FRAGMENT
   (
   CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER,
   SCHEMA_NAME INFORMATION_SCHEMA.IDENTIFIER,
   TABLE_NAME INFORMATION_SCHEMA.IDENTIFIER,
   FRAGMENT_ID INTEGER,
   FRAGMENT_CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER,
   FRAGMENT_SCHEMA_NAME INFORMATION_SCHEMA.IDENTIFIER,
   FRAGMENT_TABLE_NAME INFORMATION_SCHEMA.IDENTIFIER,
   IN_CONDITION CHARACTER VARYING (2000);
   CONSTRAINT FRAGMENT_PRIMARY_KEY PRIMARY KEY (CATALOG_NAME,
SCHEMA_NAME, TABLE_NAME, FRAGMENT_ID),
   CONSTRAINT FRAGMENT_MAPPED_TABLE_CHECK CHECK ((CATALOG_NAME,
SCHEMA_NAME, TABLE_NAME) IN (SELECT CATALOG_NAME, SCHEMA_NAME,
TABLE_NAME FROM TABLES WHERE TABLE_TYPE = 'FRAGMENTED'))
   )
```

DISCUSSION 23 – Would it be better for IN_CONDITION to be a reference to a routine? Reviewers are asked to comment.

DISCUSSION 24 – It is implied that TABLE_TYPE in the TABLES base table is extended to include the type 'FRAGMENTED'. How is this to be done? It is suggested that whatever is done here is aligned with the approach taken by DBL for FOREIGN tables in SQL/MED.

Description

Fore each mapped table *T* there is an entry *F* recorded in FRAGMENTS for each fragment *F* of *M*:

CATALOG_NAME: the SQL-catalog name of *M* as known to the Service User.

SCHEMA_NAME: the SQL-schema name of *M* as known to the Service User.

TABLE_NAME: the SQL-table name of *M* as known to the Service User.

FRAGMENT_ID: a code identifying *F* among the fragments of *M*. The FRAGMENT_IDs of *M* form a dense set starting from 1.

DISCUSSION 25 – Is "dense set" sufficiently explicit? Is it appropriate since changes might be required as the set changes over time.

FRAGMENT_CATALOG_NAME: the SQL-catalog name of *F* as known to the Service User.

FRAGMENT_SCHEMA_NAME: the SQL-schema name of *F* as known to the Service User.

FRAGMENT_TABLE_NAME: the SQL-table name of *F* as known to the Service User.

If there are more than one fragment for a mapped table the fragment for storing a row is determined by evaluation of the IN_CONDITIONs in turn starting with fragment 1. The row is stored in the first fragment for which the condition evaluates TRUE. An exception is raised if no fragment is selected.

DISCUSSION 26 – Alternatively, we could require that at least one condition evaluates to TRUE, or we could specify that a particular fragment (first or last) is assumed true.

5.4 Replicate Table

Function

Defines how mapped tables that are replicated are constructed from component table fragments.

Definition

```
CREATE TABLE REPLICATES
   (
   CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER,
   SCHEMA_NAME INFORMATION_SCHEMA.IDENTIFIER,
   TABLE_NAME INFORMATION_SCHEMA.IDENTIFIER,
   REPLICATE_ID INTEGER,
   REPLICATE_CATALOG_NAME INFORMATION_SCHEMA.IDENTIFIER,
   REPLICATE_SCHEMA_NAME INFORMATION_SCHEMA.IDENTIFIER,
   REPLICATE_TABLE_NAME INFORMATION_SCHEMA.IDENTIFIER,
   CONSTRAINT FRAGMENT_PRIMARY_KEY PRIMARY KEY (CATALOG_NAME,
SCHEMA_NAME, TABLE_NAME, REPLICATE_ID),
   CONSTRAINT FRAGMENT_MAPPED_TABLE_CHECK CHECK ((CATALOG_NAME,
SCHEMA_NAME, TABLE_NAME) IN (SELECT CATALOG_NAME, SCHEMA_NAME,
TABLE_NAME FROM TABLES WHERE TABLE_TYPE = 'REPLICATED'))
   )
```

DISCUSSION 27 – It is implied that TABLE_TYPE in the TABLES base table is extended to include the type 'REPLICATED'. How is this to be done? It is suggested that whatever is done here is aligned with the approach taken by DBL for FOREIGN tables in SQL/MED.

Description

Fore each mapped table *T* there is an entry *R* recorded in REPLICATES for each replicate *R* of *M*:

CATALOG_NAME: the SQL-catalog name of *M* as known to the Service User.

SCHEMA_NAME: the SQL-schema name of *M* as known to the Service User.

TABLE_NAME: the SQL-table name of *M* as known to the Service User.

REPLICATE_ID: a code identifying *R* among the replicates of *M*. The REPLICATE_IDs of *M* form a dense set starting from 1.

DISCUSSION 28 – Is "dense set" sufficiently explicit? Is it appropriate since changes might be required as the set changes over time.

REPLICATE_CATALOG_NAME: the SQL-catalog name of *R* as known to the Service User.

REPLICATE_SCHEMA_NAME: the SQL-schema name of *R* as known to the Service User.

REPLICATE_TABLE_NAME: the SQL-table name of *R* as known to the Service User.

5.5 ROLE_AUTHORIZATION_DESCRIPTORS base table

DISCUSSION 29 – This subclause will be revised or removed to align with Extended Security

Function

Contains a representation of the role authorization descriptors.

Definition

```
CREATE TABLE ROLE_AUTHORIZATION_DESCRIPTORS
   (
   ROLE_NAME INFORMATION_SCHEMA.SQL_IDENTIFIER,
   GRANTEE INFORMATION_SCHEMA.SQL_IDENTIFIER
   CHECK ( GRANTEE IN ( SELECT ROLE_NAME FROM ROLES )
OR GRANTEE IN ( SELECT USER_NAME FROM USERS ) ),
   GRANTOR INFORMATION_SCHEMA.SQL_IDENTIFIER
   CHECK ( GRANTOR IN ( SELECT ROLE_NAME FROM ROLES )
OR GRANTOR IN ( SELECT USER_NAME FROM USERS ) ),
   IS_GRANTABLE INFORMATION_SCHEMA.CHARACTER_DATA
   CHECK ( IS_GRANTABLE IN ( 'YES', 'NO' ) ),
   CONSTRAINT ROLE_AUTHORIZATION_DESCRIPTORS_PRIMARY_KEY
   PRIMARY KEY ( ROLE_NAME, GRANTEE ),
   CONSTRAINT ROLE_AUTHORIZATION_DESCRIPTORS_FOREIGN_KEY_ROLES
   FOREIGN KEY ( ROLE_NAME )
   REFERENCES ROLES,
   )
```

Description

1) A row is (or rows are) inserted into this table whenever a <grant role statement> or <role definition> is executed unless the necessary row already exists, in which case the existing row may be modified to change the IS_GRANTABLE column. A row is (or rows are) deleted from this table whenever a <revoke role statement> or <drop role> is executed.

2) The value of ROLE_NAME is the <role name> of some <role granted> by the <grant role statement> or the <role name> of a <role definition>.

3) The value of GRANTEE is an <authorization identifier>, possibly PUBLIC, or <role name> specified as a <grantee> contained in a <grant role statement>, or the <authorization identifier> of the current SQL-session when the <role definition> is executed.

4) The value of GRANTOR is the <authorization identifier> of the user or role who granted the role identified by ROLE_NAME to the user or role identified by the value of GRANTEE.

5) The values of IS_GRANTABLE have the following meanings:

   YES The described role is grantable.
   NO The described role is not grantable.

6) A role is grantable if the WITH ADMIN OPTION is specified in the <grant role statement> or a <role definition> is executed.

5.6 ROLES base table

DISCUSSION 30 – This subclause will be revised or removed to align with Extended Security

Function

The ROLES table has one row for each <role name> for each role known to the database management system.

DISCUSSION 31 – This subclause will be reviewed to align it with Extended Security

Definition

```
CREATE TABLE ROLES
   (
   ROLE_NAME INFORMATION_SCHEMA.SQL_IDENTIFIER PRIMARY KEY,
   CONSTRAINT ROLES_CHECK CHECK ( ROLE_NAME NOT IN ( SELECT USER_NAME
FROM USERS ) )
   )
```

Description

1) A row is inserted into this table each time a <role definition> is executed. A row is deleted from this table each time the <drop role statement> is executed.

2) The value of ROLE_NAME is the <role name> defined by <role definition>.

5.7 SERVER AUTHENTICATION base table

DISCUSSION 32 – This subclause will be revised or removed to align with Extended Security

Function

Define the authorisation identifiers to be used for given users on other servers.

DISCUSSION 33 – This subclause will be reviewed to align it with Extended Security

Definition

```
CREATE TABLE SERVER_AUTHENTICATION
   (
   SERVER_NAME INFORMATION_SCHEMA.IDENTIFIER NOT NULL,
   USER_NAME INFORMATION_SCHEMA.SQL_IDENTIFIER,
   CHECK (USER_NAME IN ( SELECT ROLE_NAME FROM ROLES ) OR USER_NAME IN (
SELECT USER_NAME FROM USERS ) ),
   AUTH_ID INFORMATION_SCHEMA.SQL_IDENTIFIER,
   CONSTRAINT PRIMARY_KEY (SERVER_NAME, USER_NAME),
   CONSTRAINT FOREIGN_KEY_SERVER_NAME FOREIGN KEY (SERVER_NAME)
REFERENCES SERVER_LOCATION,
   CONSTRAINT FOREIGN_KEY_USER_NAME_CHECK
   )
```

Description

1) SERVER_NAME identifies an SQL_Server *S*.
2) USER_NAME identifies either a User or a Role declared for this schema.
3) AUTH_ID is the auth_id that this user is entitled to use for the server *S*.

NOTE 8 – The USER_NAME is a foreign key to either ROLES table or USERS table but not both.

5.8 USERS base table

DISCUSSION 34 – This subclause will be revised or removed to align with Extended Security

Function

The USERS table has one row for each <authorization identifier> referenced in the DDA-information Schema. These are all those <authorization identifier>s that may grant privileges as well as those that may create a schema, or currently own a schema created through a <schema definition>.

Definition

```
CREATE TABLE USERS
   (
   USER_NAME INFORMATION_SCHEMA.SQL_IDENTIFIER
   CONSTRAINT USERS_PRIMARY_KEY PRIMARY KEY
   CONSTRAINT USERS_CHECK CHECK ( USER_NAME NOT IN ( SELECT ROLE_NAME
FROM ROLES ) )
   )
```

Description

1) The values of USER_NAME are <authorization identifier>s that are known as users within the DDA-distribution Schema.

6 DDA-distribution Statements

6.1 Distribution Schema Statements

To ensure consistency, language constructs are provided which may be used by the Service User to maintain the DDA-distribution schema

DISCUSSION 35 – This is consistent with the approach taken by SQL for maintaining the Information Schema

NOTE 9 – The syntax of the DDA-distribution statements is intended to be consistent with the style and format of SQL.

Statements are provided for:

- Creating a Distribution Schema
- Creating, Dropping and Altering Servers and Catalogs
- Creating, Dropping and Altering mapped tables
- Creating, Dropping and Altering users

DISCUSSION 36 – This facilities for users and their authorisation will be revised to align with Extended Security

The following subclauses of this clause have not been revised. Once the Distribution Schema definition is agreed, ballot comments will be submitted which will replace these syntax definitions with syntax consistent with both the Distribution Schema and subclause **Error! Reference source not found.**.

6.2 DDA-distribution Schema Definition

Function

This clause defines the structure of a DDA-distribution Schema definition.

Format

```
<distributed data definition> ::= CREATE DISTRIBUTION SCHEMA
   <schema name clause>
   [ <schema character set or path> ]     -- not core
   [ <schema element>... ]

<schema character set or path> ::= <schema character set specification>
   | <schema path specification>
   | <schema character set specification> <schema path specification>
   | <schema path specification> <schema character set specification>

<schema name clause> ::= <schema name>
   | AUTHORIZATION <schema authorization identifier>
   | <schema name> AUTHORIZATION <schema authorization identifier>

<schema authorization identifier> ::= <authorization identifier>

<schema character set specification> ::= DEFAULT CHARACTER SET
<character set specification>

<schema path specification> ::= <path specification>

<schema element> ::= <distributed schema element>
   | <view definition>
   | <character set definition>  -- not core
   | <collation definition>  - not core
   | <translation definition> - not core
   | <user-defined type definition> -- constrained
   | <schema routine>
   | <grant statement>
   | <role definition>
   | <grant role statement>

<distributed schema element> ::= <server location definition>
   | <server authentication>
   | <mapped table definition>
   | <partitioned table definition>
```

Syntax Rules

1) There must be at least one <server location definition>.

General Rules

2) A row is inserted into the Distributed Schema Schemata table with the <schema name> defined in the schema name clause.

3) <distributed schema element> components are defined in following clauses. Other elements are defined in IS 9075.

6.3 DDA-distribution Schema Elements

This subclause defines the constructs in the DDA-distribution Statements which are specific to the DDA-distribution Schema, or which differ from the equivalent statements in IS 9075.

6.3.1 Server Location Definition

Function

Name the servers contributing to the DDB and provide connection information

Syntax

```
<server location definition> ::=
   CREATE SERVER LOCATION <server name> TRANSPORT MAPPING <transport
name> ADDRESS <address>
   RDA_DESTINATION <rda server name>
   INFORMATION SCHEMA <catalog name>
```

Syntax Rules

1) < server name> is unique amongst all the Server Location Definitions in the DDA-distribution Schema.

General Rules

1) A row is inserted into the Server Location table of the DDA-distribution Schema with values ....

6.3.2 Server Authentication

Function

Identify the authentication identifier to be used for a given user for access to data at a server.

Syntax

```
<server authentication> ::= LOGIN [ FOR < authorisation identifier> ]
AT <server name> IS <authorisation identifier>
```

Syntax Rules

1) Let *A* be the < authorisation identifier> in the FOR subclause. If there is no FOR subclause then *A* is the user AUTHORISATION_ID for this session.
2) Let *AA* be the <authorisation identifier> in the IS subclause.
3) Let *S* be the SQL-implementation identified by < server name>.
4) There is at most one <server authentication> for any *A* and *S*.

General Rules

1) A row is inserted into the User Authorisation Table specifying that the authorisation identifier to be used when a connection is made to server *S* for user *A* is *AA*.

2) Tables available to user *AA* in *S* are available to user *A* of this schema with the same priviliges.

6.3.3 Mapped Table Definition

Function

Creates a description of a mapped table in the DDA-distribution Schema. A mapped table is mapped to tables in the contributing SQL-implementations using partitioning, replication, or both.

Syntax

```
< mapped table definition > ::= CREATE MAPPED TABLE <table name >
<table element list> )
<component list>
<component list> ::= < component>
| <conditional component> {, < conditional component> }...
<conditional component>  ::=   IF < insert condition> <component>
<component> ::= AT <server table list>
<server table ref list> ::= <server table ref> [ {, <server table ref>
} ... ]
<server table ref> ::=  <server name> [ ALIAS <table name> ]
NOTE: there are many ways of defining allocation schemes and one could
make some sort of case for most of them! This one, and the CASE one are
the most obvious.
<table element list> ::=
<left paren> <table element> [ { <comma> <table element> }... ] <right
paren>
<table element> ::=
<column definition>
| <table constraint definition>
```

Rules

1) Let *T* be the value of <table name> in < Mapped Table Assertion>.

2) *T* is unique among the Views and Mapped tables in this Schema.

3) Let S be the value of < server name> in a < server table ref> and ST be,

   a) if there is no ALIAS subclause the value T,

   b) else the value of <table name> in the ALIAS subclause.

4) <conditional expression> is a boolean expression referencing only columns in T and constants.

5) For each server S in the <server table list> there is a table with name ST. Further, for each server S the name ST is unique amongst the set of names ST in the component list.

The tables ST in all the SQL-implementations have identical structure. That is, they have the same column names in the same order and with the same data types.

Table T is defined to have the same structure as each of the tables ST.

Better - define the columns here - then they could be a subset of those in the servers, and in a different order.

6) The Tables ST in the SQL-implementations may have columns in a different order and may contain additional columns with NULL values allowed.

7) <insert condition> is a boolean expression ranging over column values in the table T.

NOTE: This may involve a function defined in the Distributed Schema and returning fragment number, e.g. (f(row) = 1) if we allow functions - routines - to be defined in the DDA-distribution Schema.

General Rules

1) Each <conditional component> in the <component list> defines a fragment of the table T, such that the fragments are disjoint. That is, the fragments have no rows in common.

2) If <server table ref list> contains more than one <server table ref> then the tables in the list are replicates, each containing the same set of rows.

3) When a new row is inserted into the table the <condition>'s are evaluated in the order of the <component list> until one of them evaluates to TRUE. If there is no <condition> the effect is as if a condition was always TRUE. The row is then inserted into each of the tables in the corresponding <server table list>. If no <condition> evaluates to TRUE then an exception is raised.

Alternative - the last <condition> must be unconditional TRUE

4) An exception is raised if an attempt to insert a row into a component table fails.

5) A row defining table T is inserted into the DDA-distribution Schema Tables table, and

   a) A row is inserted into the Fragment Table for each <component>

   b) A row is inserted into the Replicated_Fragment table for each <server table ref>.

   c) A row is inserted into the Columns table for each column defined in the <table element list>

   d) We could have a table to link columns that feature in the <insert condition> to the fragments - is it required?

7 Exceptions

The DDA rules include rules that when evaluated may result in the raising of an exception.

DISCUSSION 37 – There are currently no DDA-specific conditions. This clause is included as a placemarker.

7.1 Exception codes for DDA-specific Conditions

The SQL conditions that may be raised as a result of the invocation of a DDA-distribution Schema Statement are listed in Table 1 together with the corresponding status codes.

Table 1 – SQLSTATE class and subclass values for DDA-specific conditions

Condition	Class	Subcondition	Subclass
RDA-specific condition	HZ		10001
			10002
			10003
			10004
			10005
			10006
			10007
			10008
			10009
			10010
			10011
			10012
			10013
			10014
			10015
			10016
			10017
			10018
			10019
			10020
			10021
			10022

8 Conformance

An implementation of DDA-services shall claim conformance as defined in this clause.

8.1 DDA Conformance

An implementation that claims conformance to this International Standard shall provide a, DDA-distribution Schema and DDA-distribution Statements as defined by this International Standard.

An implementation that claims conformance to this International Standard must also conform to ISO/IEC 9075 (SQL).

> *DISCUSSION 38 – It is expected that international review may wish to adjust this conformance statement. An alternative would be to permit separate conformance to each of DDA-distribution Schema and DDA-distribution Statements to be made. In this case, the Conformance proforma should have a question for each of these.*

8.2 Claims of Conformance

Claims of conformance to this International Standard shall be accompanied by a completed proforma for the information listed in Annex A.

Annex A Conformance Proforma

(normative)

A copy of the proforma in this annex is to be completed for each DDA implementation that claims to conform to this International Standard.

All items in 8.2 to 0 shall be completed and either the items in A.2 or **Error! Reference source not found.** according to the entry in item 0.

A.1. Identification

A.1.1 Conformance statement for: ISO/IEC 9579/Am2:200y (E), Information Technology — Distributed Database Access for SQL.
A.1.2 Date of Statement:

A.2. Supplier Details

A.2.1 Organisation:
A.2.2 Contact Name(s):
A.2.3 Address:
A.2.4 Telephone:
A.2.5 Fax:
A.2.6 Email:
A.2.7 Other contact information:

A.3. Implementation Details

A.3.1	Implementation Name:
A.3.2	Version:
A.3.3	Hardware name:
A.3.4	Hardware version:
A.3.5	Operating system name:
A.3.6	Operating system version:
A.3.7	Special configuration requirements:
A.3.8	Other implementation information:

A.4. DDA Support

A.4.1	Specify a list of amendments implemented (or none):
A.4.2	Specify a list of technical corrigenda implemented (or none):

Annex B Extensions to RDA

(normative)

DISCUSSION 39 – There are no extensions to RDA at present

Annex C Examples of DDA architecture

(informative)

This Annex presents a number of system configurations that conform to this International Standard.

For the purposes of this Annex, the term “remote” is used informally to describe components that are connected by a communications medium, and the term “local” is used informally to describe components that are not remote. For the purposes of this Annex, RMDM diagram conventions are varied to explicitly show multiple instances of a processor for purposes of clarification.

C.1. Local SQL-servers

A DDA-controller may be used to combine local SQL-environments and present them as one SQL-environment to a Service User as modelled in Figure 4.

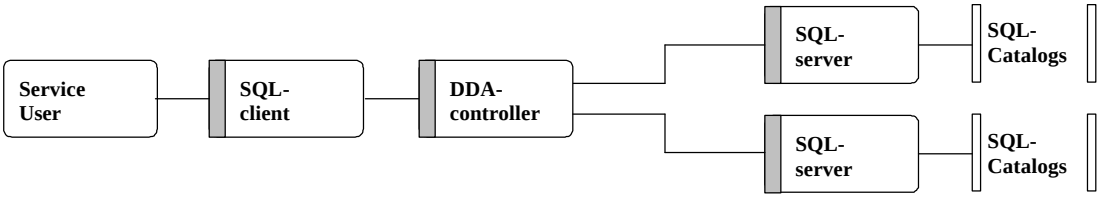

Figure 4 – Local SQL-servers

C.2. Remote SQL-servers

DDA-controllers may be used to combine SQL-environments which are remote from the Service User, so as to present them as one SQL-environment to a Service User as modelled in Figure 5 which shows two clients both connecting to two remote servers.

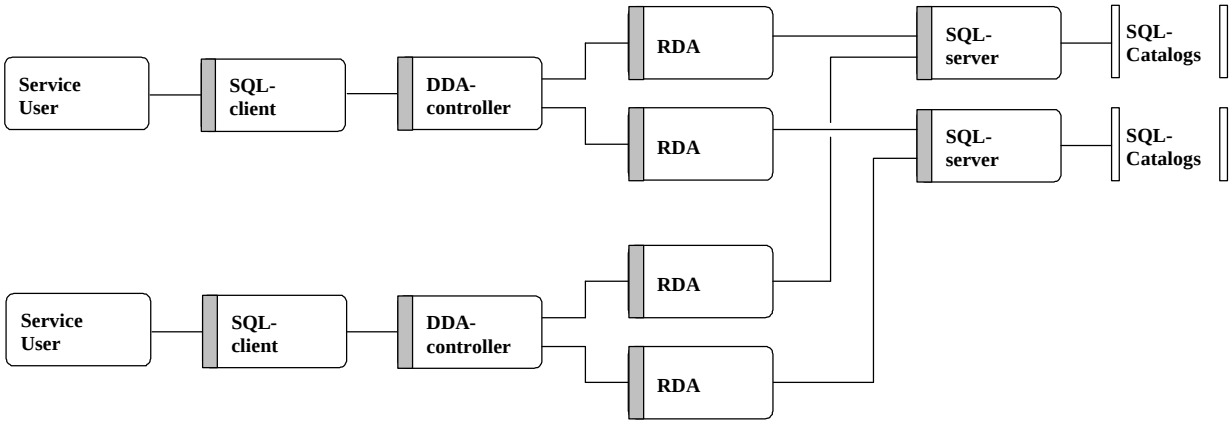

Figure 5 – Remote SQL-servers

C.3. Remote DDA-server

An SQL-client may connect to a remote SQL-server that is itself a Distributed Database, using a DDA-controller to combine SQL-environments, which may themselves be remote from the DDA-services so as to present them as one SQL-server to the SQL-client as modelled in Figure 6.

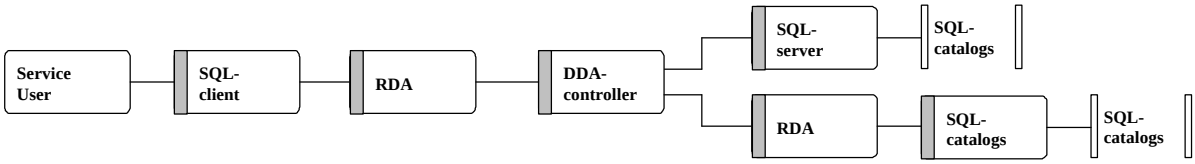

Figure 6 – Remote DDA-server

NOTE 10 – This International Standard does not require that a remote SQL-server that is distributed to contain conforming DDA-services or to use RDA to communicate with its component SQL-servers.