

GWD-R
GGF DAIS Working Group

Editors
Mario Antonioletti, University of Edinburgh
Brian Collins, IBM
Amy Krause, University of Edinburgh
Simon Laws, IBM
James Magowan, IBM
Susan Malaika, IBM
Norman W Paton, University of Manchester

Category: INFORMATIONAL

May 21, 2004

Web Services Data Access and Integration - The Relational Realisation (WS-DAIR)

Status of This Memo

This memo provides information regarding the specification of service-based interfaces to data resources. The specification is presently a draft for discussion. It does not define any standards or technical recommendations. Distribution is unlimited.

Copyright Notice

Copyright © Global Grid Forum (2004). All Rights Reserved.

Abstract

Data resources play a significant role in many applications across multiple domains. Web services provide implementation neutral facilities for describing, invoking and orchestrating collections of networked resources. The GGF (Global Grid Forum) Open Grid Services Architecture (OGSA), and its associated specifications, defines consistent interfaces through web services between components of the grid infrastructure. Both the web and grid communities would benefit from the provision of consistent and agreed web service interfaces for data resources and the systems that manage them.

This document, *Web Services Data Access and Integration: The Relational Realisation (WS-DAIR)*, presents a specification for a collection of data access interfaces for relational data resources, which extends interfaces defined in the *Web Services Data Access and Integration* document [WS-DAI], which in turn is based on the *OGSA Data Services* document [Data Services].

This document is presented for discussion within the GGF Database Access and Integration Services (DAIS) Working Group, with a view to the document evolving to become a proposed recommendation. There are several respects in which the current proposal is incomplete, but it is hoped that the material included is sufficient to allow an informed discussion to take place concerning both its form and substance.

Related DAIS specifications define how other data resources and systems can be described and manipulated through web services. The DAIS specifications form part of a broader activity within the GGF to develop OGSA. The DAIS specifications can be applied in regular web services environments or as part of a grid fabric.

Contents

Abstract.....	1
1. Introduction.....	3
1.1 Specification Scope.....	3
1.2 Specification Organisation.....	3
1.3 Interface Composition.....	4
2. Notational Conventions.....	4
3. Terminology and Concepts.....	5
3.1 Terminology.....	5
3.2 Concepts.....	5
3.3 Relationships with other specifications.....	6
4. DataDescription.....	6
4.1 RelationalDescription.....	6
4.2 SQLResponseDescription.....	8
4.3 RowSetDescription.....	8
4.4 SQLFactoryDescription.....	9
5. DataAccess.....	10
5.1 SQLAccess.....	10
5.2 SQLResponseAccess.....	11
5.3 RowSetAccess.....	13
6. Derived Data Access.....	14
6.1 SQLFactory.....	14
7. DataManagement.....	16
8. Mapping to WSDL.....	16
9. Security Considerations.....	17
10. Conclusion.....	17
Editor Information.....	17
Contributor Information.....	18
Acknowledgements.....	18
Intellectual Property Statement.....	19
Full Copyright Notice.....	19
References.....	20
Appendix A.1 – SQLAccess WSDL Port Types.....	21
Appendix A.2 – SQLAccess XML Schema.....	21
Appendix A.3 – SQLAccess WSDL.....	24
Appendix B.1 – SQLResponseAccess WSDL Port Types.....	29
Appendix B.2 – SQLResponseAccess XML Schema.....	30
Appendix B.3 – SQLResponseAccess WSDL.....	32
Appendix C.1 – RowSetAccess WSDL Port Types.....	38
Appendix C.2 – RowSetAccess XML Schema.....	39
Appendix C.3 – RowSetAccess WSDL.....	41

1. Introduction

Data access plays a central role for many types of Grid applications. Data access generally involves the retrieval, manipulation and insertion of data, which may be stored using a range of different formats and infrastructures. For service-based architectures this requires establishing a flexible framework for dealing with data requests and integrating or exposing existing functionality for managing data and moving data to be retrieved from, or inserted into, a relational Data Resource.

This document presents a specification for a collection of data access interfaces for relational Data Resources. A relational Data Resource is a data source/sink, together with any associated management infrastructure, that are characteristic of relational database systems, e.g., can be queried or updated using SQL or any other suitable relational query/update language. The interfaces are thus categorized according to the support they provide for:

- Data Description
- Data Access
- Data Factories
- Data Management

As such, this document should be read in conjunction with both the *OGSA Data Services* document [Data Services] and the generic *Web Services Data Access and Integration* document [WS-DAI], which define the base interfaces that are extended in this document. These specifications are being developed for representing data resources as web services, and form part of a broader activity within the Global Grid Forum to develop the Open Grid Services Architecture (OGSA). This document does not mandate how the interfaces are composed into services. The proposed interfaces may be used in isolation or in conjunction with others.

1.1 Specification Scope

The *OGSA Data Services* document [Data Services] introduces *DataAccess*, *DataFactory* and *DataManagement* interfaces, and discusses the role of *Data Description* in the provision of service-based interfaces to data resources. These are extended in the *Web Services Data Access and Integration* document [WS-DAI].

This specification extends the interfaces presented in the *Web Services Data Access and Integration* document [WS-DAI] to allow access to and description of relational Data Resources. The interfaces presented here are aligned with the base types provided in the *OGSA Data Services* document [Data Services]. The relational Data Resources are assumed to be composed of databases and tables, which are accessible using SQL. In addition, common relational resources such as stored procedures are also described.

The *DataManagement* interface has not been considered fully, but information on this topic is retained in Section 7.

1.2 Specification Organisation

This specification separates the function of a Data Service from its operational representation as expressed by WSDL. To this end, this document discusses interfaces generally and the Mapping represents these interfaces using concrete WSDL, as described in Section 8. This approach allows functions to be mapped to WSDL in many different ways. However when the wider standards space has stabilized, a specific mapping will be advocated.

1.3 Interface Composition

This specification does not mandate how interfaces are composed into services; the proposed interfaces may be used in isolation or in conjunction with others. Viable compositions of interfaces will, initially, follow established patterns for data access.

Here a Data Service provides `SQLAccess` and `RowSetAccess` interfaces for a relational Data Service that is associated with a relational database.

2. Notational Conventions

The key words “MUST,” “MUST NOT,” “REQUIRED,” “SHALL,” “SHALL NOT,” “SHOULD,” “SHOULD NOT,” “RECOMMENDED,” “MAY,” and “OPTIONAL” are to be interpreted as described in RFC-2119 [RFC2199]

When describing concrete XML schemas, this specification uses the notational convention of [WS-Security]. Specifically, each member of an element’s children or attributes property is described using an XPath-like notation (e.g., `/x:MyHeader/x:SomeProperty/@value1` indicates that namespace *x* is being used, the root element *MyHeader* and a child element *SomProperty* with an attribute *value1*). The use of `{any}` indicates the presence of an element wildcard (`<xsd:any/>`). The use of `@{any}` indicates the presence of an attribute wildcard (`<xsd:anyAttribute/>`).

Italicised element names are used when the element is intended to be specified by subsequent DAIS specifications.

This specification generally adopts the terminology defined in the *OGSA Data Services* document [Data Services]. In particular the terms Data Service, Data Resource and Data Set are used. The *OGSA Data Services* document is still evolving and this terminology may change in future versions of the DAIS Working Group specifications.

This specification uses namespace prefixes throughout; these are listed in the table below. Note that the choice of any namespace prefix is arbitrary and is not semantically significant.

Prefix	Namespace
http	http://www.w3.org/2002/06/wSDL/http
wSDL	http://schemas.xmlsoap.org/wSDL/
xsd	http://www.w3.org/2001/XMLSchema
xsi	http://www.w3.org/2001/XMLSchema-instance
wsdai	http://www.ggf.org/namespaces/2004/05/WS-DAI
wsdair	http://www.ggf.org/namespaces/2004/05/WS-DAIR
wsdairs	http://www.ggf.org/namespaces/2004/05/WS-DAIRS
wrs	http://java.sun.com/xml/ns/jdbc/webrowset.xsd
wsa	http://schemas.xmlsoap.org/ws/2004/03/addressing
wsbf	http://www.ibm.com/xmlns/stdwip/web-services/WS-BaseFaults
wsrl	http://www.ibm.com/xmlns/stdwip/web-services/WS-ResourceLifetime
wsrp	http://www.ibm.com/xmlns/stdwip/web-services/WS-ResourceProperties
wsx	http://schemas.xmlsoap.org/ws/2004/03/mex
wsp	http://schemas.xmlsoap.org/ws/2002/12/policy

3. Terminology and Concepts

3.1 Terminology

The model independent terminology, e.g., Data Resource, is given in the *Web Services Data Access and Integration* document [WS-DAI].

3.2 Concepts

3.2.1 DataDescription

The *DataDescription* interfaces allow a description of data represented by Data Services to be provided. No operations are defined within these interfaces. The model independent specification for these is given in the *Web Services Data Access and Integration* document [WS-DAI]. Here they are extended to provide a description of relational Data Resources. These are the main points of extension for relational Data Resources:

- *RelationalDescription*: provides information about relational logical schemas that describe Databases, Domains, Tables, Constraints, Columns, ColumnTypes, Keys and Views that a Data Service may represent. It also includes StoredProcedures, UserDefinedTypes, UserDefinedFunctions and Triggers. It also describes physical schemas that describe indexes, sizes of tables and statistics on column values.
- *RowSetDescription*: provides information about a particular instance of a query result that a Data Service may represent. This interface will make available information about the schema for representing the query result and the number of rows within the RowSet.

These capabilities are described in Section 5.

3.2.2 DataAccess

DataAccess operations allow relational Data Resources to be modified through insertion or updates, or queried through an appropriate language. The following Data Access interfaces are defined:

- *SQLAccess* – provides access to a relational Data Resource.
- *SQLResponseAccess* – provides access to each type of Response that can result from the execution of a SQLExpression.
- *RowSetAccess* – provides access to a set of rows which are usually the result of a SQLExpression containing a SELECT statement.

These are covered in more detail in Section 4.

3.2.3 DataFactory

The *DataFactory* interfaces allow data represented in relational Data Resources, usually as the result of a query or update, to be instantiated as Data Services. The specializations in this instance thus deal with the type of SQLExpression that can be passed to a *DataFactory* to expose the results in a meaningful fashion. The properties and interfaces that will be supported by these Data Services are specified in the schema for the creation parameters. *DataFactory* specializations are:

- *SQLExecuteFactory* – provides access to a relational Data Resource

These are covered in more detail in Section 6.

3.3 Relationships with other specifications

WS-DAIR does not provide its own query/update languages for relational Data Resources. Instead, it acts as a conduit for existing relational query and update languages to be conveyed to the appropriate relational Data Resources, in this instance relational Data Resources or a Data Resource that supports relational type queries. As such WS-DAIR relies on existing relational query and update languages. In this document, interface support is provided for languages based on the following standards:

- **SQL:** an ISO standard defining a language for querying and updating relational Data Resources [SQL2003].
- **WebRowSet:** a Java Community Process standard for relational results is one of the valid *ResponseFormats* for responses from SQLAccess operations [JSR114].
- **CIM:** is the Common Information Model, a DMTF standard, to which the DAIS-WG and the CGS-WG plan to submit a proposal for extension to include relational database properties and data management operations [CIM].

4. DataDescription

The DataDescription interfaces allow metadata (terms) to be made available. DataDescription interfaces are provided for use with relational Data Resources and for RowSets.

4.1 RelationalDescription

4.1.1 Behavioral Properties

The metadata described in this section are associated with a relational Data Resource.

4.1.1.1 LanguageCapabilities

```
<xsd:complexType name="LanguageCapabilitiesType">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="LanguageCapabilities"
  type="wsdair:LanguageCapabilitiesType" />
```

/wsdair:LanguageCapabilities

Describes the dialect of the SQL language that the underlying relational database management system should support. Possible values could be *SQL92Expression*, *SQL99Expression*, *SQL03Expression*, *SQL07Expression*.

The above property will be defined in a proposal arising from a joint activity of the DAIS-WG and the CGS-WG to extend the DMTF Common Information Model (CIM).

4.1.2 Informational Properties

DataDescription provides information about a single relational Data Resource represented by a Data Service.

4.1.2.1 RelationalSchema

```
<xsd:complexType name="RelationalSchemaType">
  <xsd:sequence>
    <xsd:element name="Content?" type="xsd:string"/>
  </xsd:sequence>
```

```

</xsd:complexType>

<xsd:element name="RelationalSchema" type="wsdair:RelationalSchemaType"
/>

```

/wsdair:RelationalSchema

Describes the schema of the relational data, for example Databases, Domains, Tables, Constraints, Columns, ColumnTypes, Keys, Views and Indexes.

4.1.2.2 StoredProcedures

```

<xsd:complexType name="StoredProcedureListType">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="StoredProcedures"
  type="wsdair:StoredProcedureListType" />

```

/wsdair:StoredProcedures

Describes the names, input and output types of the stored procedures available.

4.1.2.3 UserDefinedTypes

```

<xsd:complexType name="UserDefinedTypesListType">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="UserDefinedTypes"
  type="wsdair:UserDefinedTypesListType" />

```

/wsdair:UserDefinedTypes

Describes the names and definitions of the user defined types available.

4.1.2.4 UserDefinedFunctions

```

<xsd:complexType name="UserDefinedFunctionListType">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="UserDefinedFunctions"
  type="wsdair:UserDefinedFunctionListType" />

```

/wsdair:UserDefinedFunctions

Describes the names and definitions of the user defined functions available.

4.1.2.5 Triggers

```

<xsd:complexType name="Triggers">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

```

```
<xsd:element name="Triggers" type="wsdair:TriggersType" />
```

/wsdair:Triggers

Describes the names and definitions of the triggers available.

The above properties will be defined in a proposal arising from a joint activity of the DAIS-WG and the CGS-WG to extend the DMTF Common Information Model (CIM).

4.2 SQLResponseDescription

4.2.1 Behavioral Properties

No metadata is currently defined.

4.2.2 Informational Properties

4.2.2.1 NumberOfRowSets

```
<xsd:element name="NumberOfRowSets" type="xsd:int" />
```

/wsdair:NumberOfRowSets

The total number of *RowSets* in the *SQLResponse*.

4.2.2.2 NumberOfUpdateCounts

```
<xsd:element name="NumberOfUpdateCounts" type="xsd:int" />
```

/wsdair:NumberOfUpdateCounts

The total number of *UpdateCounts* in the *SQLResponse*.

4.2.2.3 NumberOfReturnValues

```
<xsd:element name="NumberOfReturnValues" type="xsd:int" />
```

/wsdair:NumberOfReturnValues

The total number of *ReturnValues* in the *SQLResponse*.

4.2.2.4 NumberOfOutputParameters

```
<xsd:element name="NumberOfOutputParameters" type="xsd:int" />
```

/wsdair:NumberOfOutputParameters

The total number of *OutputParameters* in the *SQLResponse*.

4.2.2.5 NumberOfSQLCommunicationsAreas

```
<xsd:element name="NumberOfSQLCommunicationsAreas" type="xsd:int" />
```

/wsdair:NumberOfSQLCommunicationsAreas

The total number of *SQLCommunicationsAreas* in the *SQLResponse*.

4.3 RowSetDescription

4.3.1 Behavioral Properties

The metadata described in this section are associated with a *RowSet*.

4.3.1.1 CursorDirection

```
<xsd:element name="CursorDirection">
  <xsd:simpleType>
```



```

    <xsd:restriction base="xsd:token">
      <xsd:enumeration value="ForwardOnly"/>
      <xsd:enumeration value="ForwardAndReverse"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

```

/wsdairs:CursorDirection

Describes whether the RowSet can be navigated in a forward only or a forward and reverse direction.

4.3.1.2 CursorHeldOverTxnBoundary

```

<xsd:element name="CursorHeldOverTxnBoundary" type="xsd:boolean" />

```

/wsdairs:CursorHeldOverTxnBoundary

Describes whether the RowSet can still be navigated after a transaction has been committed.

4.3.2 Informational Properties

4.3.2.1 RowSetSchema

```

<xsd:complexType name="RowSetSchemaType">
  <xsd:sequence>
    <xsd:element ref="wrs:metadata"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:element name="RowSetSchema" type="wsdairs:RowSetSchemaType" />

```

/wsdairs:RowSetSchema

For example, WebRowSet, see [JSR114].

4.3.2.2 NumberOfRows

```

<xsd:element name="NumberOfRows" type="xsd:int" />

```

/wsdairs:NumberOfRows

The total number of rows in the result set.

4.4 SQLFactoryDescription

4.4.1 Behavioral Properties

4.4.1.1 LanguageCapabilities

```

<xsd:complexType name="LanguageCapabilitiesType">
  <xsd:sequence>
    <xsd:element name="Content" type="xsd:string"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:element name="LanguageCapabilities"
  type="wsdair:LanguageCapabilitiesType" />

```

/wsdair:LanguageCapabilities

Describes the dialect of the SQL language that the underlying relational database management system should support. Possible values could be *SQL92Expression*, *SQL99Expression*, *SQL03Expression*, *SQL07Expression*.

The above property will be defined in a proposal arising from a joint activity of the DAIS-WG and the CGS-WG to extend the DMTF Common Information Model (CIM).

5. DataAccess

5.1 SQLAccess

This *SQLAccess* interface provides access to the underlying relational Data Resource by means of SQL statements.

5.1.1 Overview - SQLAccess

Data Access collects together messages that directly access and modify the data represented by a Data Service along with the behavioral properties which describe the behavior of these access messages, as, for example, in Figure 1:

A relational Data Service implements the *SQLAccess* operations and exposes the *RelationalDescription* informational properties. In this example a consumer uses the *SQLExecute* message to submit a *SQLExpression*. The associated *SQLResponse* message will contain some combination of *SQLExecuteResponseTypes*. The actual combination will depend upon the actual *SQLExpression*, for example:

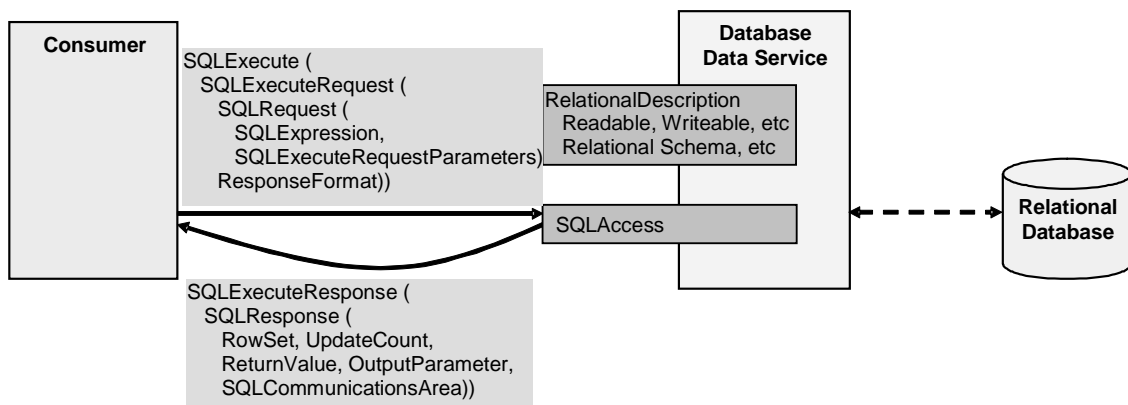


Figure 1 – Overview – SQLAccess::SQLExecute

- SELECT
 - *RowSet* - (1)
 - *SQLCommunicationsArea* (0 to n)
- INSERT, UPDATE, DELETE
 - *UpdateCount* - (1)
 - *SQLCommunicationsArea* – (0 to n)
- StoredProcedure
 - *RowSet* - (0 to n)
 - *ReturnValue* – (0 to n)
 - *OutputParameters* - (0 to n)
 - *SQLCommunicationsArea* – (0 to n)
- UserDefinedFunction
 - *ReturnValue* – (1 to n)
 - *SQLCommunicationsArea* – (0 to n)

The Consumer will need to process the *SQLResponse* appropriately.

5.1.2 Operations

5.1.2.1 SQLAccess::SQLExecute

Direct an *SQLExpression* and optional *SQLExecuteRequestParameters* to the relational Data Resource.

The *SQLExecuteRequestParameters* are primarily for use with Stored Procedures and User Defined Functions but it is also intended that an *InputParameter* be used to provide a reference to a dataset as an *InputParameter* to a *SQLExpression* containing a bulk load or similar update statement.

Input

- *SQLExecuteRequest* - the *SQLExecute* operation that is to be run on the relational Data Resource.
 - *SQLRequest*
 - *SQLExpression* - (1) - any SQL statement
 - *SQLExecuteRequestParameters* - (0 to 1)
 - *InputParameters* - (0 to n)
 - *OutputParameters* - (0 to n)
 - *InOutParameters* - (0 to n)
 - *ResponseFormat* - the format(s), selected from the *SQLExecuteResponseTypes* property, which the *SQLResponse* will conform to.

Output

- *SQLExecuteResponse* – the *SQLResponse* returned in the *ResponseFormats* from the *SQLExecute* operation.
 - *SQLResponse*
 - *RowSet* - (0 to n) - e.g. *WebRowSet* see [JSR114]
 - *UpdateCount* - (0 to n)
 - *ReturnValue* - (0 to n)
 - *OutputParameter* - (0 to n)
 - *SQLCommunicationsArea* - (0 to n)
 - *SQLState* - (1 to n) – an XOPEN or SQL99 code identifying the Exception, Warning or Message.
 - *VendorCode* - (1 to n) – a database vendor-specific code for the Exception, Warning or Message.
 - *MessageText* - (1 to n) – a text description of the Exception, Warning or Message.

Faults

- *InvalidSQLExecuteRequest* - XML syntax error or XML schema non-compliance.
- *InvalidSQLExecuteRequestParameters* - Parameters do not match *SQLExpression*.
- *InvalidResponseFormat* - *ResponseFormat* not valid.
- *OtherFault* – any other fault.

5.2 SQLResponseAccess

This allows access to each *SQLExecuteResponseType* in the *SQLResponse* data by executing the appropriate *SQLResponseAccess* operation.

5.2.1 Operations

5.2.1.1 SQLResponseAccess::GetRowSet

Get a *RowSet* from the *GetRowSetResponse*.

Input

- *GetRowSetRequest*
 - *RowSetNumber* - (1) – the number of the required *RowSet*.

Output

- *GetRowSetResponse*
 - *RowSet* - (1) – the requested *RowSet* e.g. *WebRowSet* see [JSR114].

Faults

- *InvalidGetRowSetRequest* - XML syntax error or XML schema non-compliance.
- *InvalidRowSetNumber* - not a valid *RowSetNumber*.
- *OtherFault* – any other fault.

5.2.1.2 *SQLResponseAccess::GetUpdateCount*

Get an *UpdateCount* from the *GetUpdateCountResponse*.

Input

- *GetUpdateCountRequest*
 - *UpdateCountNumber* - (1) – the number of the required *UpdateCount*.

Output

- *GetUpdateCountResponse*
 - *UpdateCount* - (1) – the requested *UpdateCount*.

Faults

- *InvalidUpdateCountRequest* - XML syntax error or XML schema non-compliance.
- *InvalidUpdateCountNumber* - not a valid *UpdateCountNumber*.
- *OtherFault* – any other fault.

5.2.1.3 *SQLResponseAccess::GetReturnValue*

Get a *ReturnValue* from the *GetReturnValueResponse*.

Input

- *GetReturnValueRequest*
 - *ReturnValueNumber* - (1) – the number of the required *ReturnValue*.

Output

- *GetReturnValueResponse*
 - *ReturnValue* - (1) – the requested *ReturnValue*.

Faults

- *InvalidGetReturnValueRequest* - XML syntax error or XML schema non-compliance.
- *InvalidReturnValueNumber* - not a valid *ReturnValueNumber*.
- *OtherFault* – any other fault.

5.2.1.4 *SQLResponseAccess::GetOutputParameter*

Get an *OutputParameter* from the *GetOutputParameterResponse*.

Input

- *GetOutputParameterRequest*
 - *OutputParameterNumber* - (1) – the number of the required *OutputParameter*.

Output

- *GetOutputParameterResponse*
 - *OutputParameter* - (1) – the requested *OutputParameter*.

Faults

- *InvalidOutputParameterRequest* - XML syntax error or XML schema non-compliance.
- *InvalidOutputParameterNumber* - not a valid *OutputParameterNumber*.
- *OtherFault* – any other fault.

5.2.1.5 *SQLResponseAccess::GetSQLCommunicationsArea*

Get a *SQLCommunicationsArea* from the *GetSQLCommunicationsAreaResponse*.

Input

- *GetSQLCommunicationsAreaRequest*
 - *SQLCommunicationsAreaNumber* - (1) – the number of the required *SQLCommunicationsArea*.

Output

- *GetSQLCommunicationsAreaResponse*
 - *SQLCommunicationsArea* - (1) – the requested *SQLCommunicationsArea*.
 - *SQLState* - (1 to n) – an XOPEN or SQL99 code identifying the Exception, Warning or Message.
 - *VendorCode* - (1 to n) – a database vendor-specific code for the Exception, Warning or Message.
 - *MessageText* - (1 to n) – a text description of the Exception, Warning or Message.

Faults

- *InvalidSQLCommunicationsAreaRequest* - XML syntax error or XML schema non-compliance.
- *InvalidSQLCommunicationsAreaNumber* - not a valid *SQLCommunicationsAreaNumber*.
- *OtherFault* – any other fault.

5.3 RowSetAccess

This allows access to the underlying data by means of rows.

5.3.1 Operations**5.3.1.1 RowSetAccess::GetTuples**

Return a specified number of tuples from a service that represents a result set.

Input

- *GetTuplesRequest*
 - *SQLRequest*
 - *StartPosition* - the position of the first tuple to be returned (First tuple is position 1).
 - *Count* - the number of tuples.

Output

- *GetTuplesResponse*
 - *SQLResponse*
 - *RowSet* - (0 or 1) - e.g. *WebRowSet* see [JSR114].
 - *SQLCommunicationsArea* - (0 to n)
 - *SQLState* - (1 to n) – an XOPEN or SQL99 code identifying the Exception, Warning or Message.
 - *VendorCode* - (1 to n) – a database vendor-specific code for the Exception, Warning or Message.
 - *MessageText* - (1 to n) – a text description of the Exception, Warning or Message.

Faults

- *InvalidGetTuplesRequest* - XML syntax error or XML schema non-compliance.
- *InvalidStartPosition* - not a valid *StartPosition*; cannot start with tuple specified (out of bounds value).
- *InvalidCount* - not a valid *Count*; cannot return that number of tuples.
- *OtherFault* – any other fault.

6. Derived Data Access

6.1 SQLFactory

The SQLExecuteFactory operation is used to create a service representing a relational Data Resource which fulfills the desired behavior, exposes the desired interfaces and represents the results of the SQL Query.

6.1.1 Overview – SQLFactory::SQLExecuteFactory with SQLResponseAccess::GetRowSet

This factory pattern allows a new Data Service relational Data Resource relationship to result from messages on another Data Service. This ability to derive one Data Service from another to provide different views of the same relational Data Resources leads to a collection of notionally related Data Service instances, for example, see Figure 2.

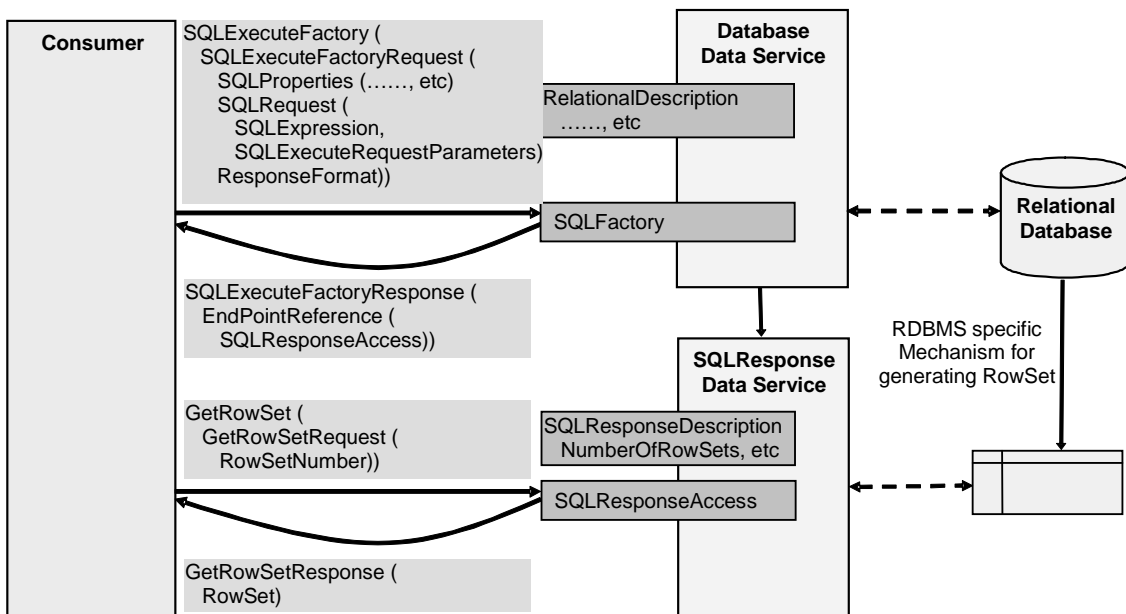


Figure 2 - Overview - SQLFactory::SQLExecuteFactory with SQLResponseAccess::GetRowSet

This example presents a SQLAccess interface. The SQLExecuteFactory operation is used to construct the derived SQLResponse Data Service. This service provides access to the *RowSet* resulting from a *SQLExpression* against the Relational Database, assuming that the expression contains a SELECT statement. The *RowSet* is a subset or restriction of the data in the database and is presented in tabular form. The *RowSet* could be stored as a table in a relational database or decoupled from the database, but the important distinction here is that the data is represented as a collection of rows that does not implement the SQLAccess portType. Instead, the SQLResponse Data Service presents the *SQLResponseAccess* collection of operations that allows the *RowSet* to be retrieved but does not provide facilities for submitting SQL expressions.

6.1.2 Overview – SQLFactory::SQLExecuteFactory with SQLResponseAccess::RowSetSelectionFactory with RowSetAccess::GetTuples

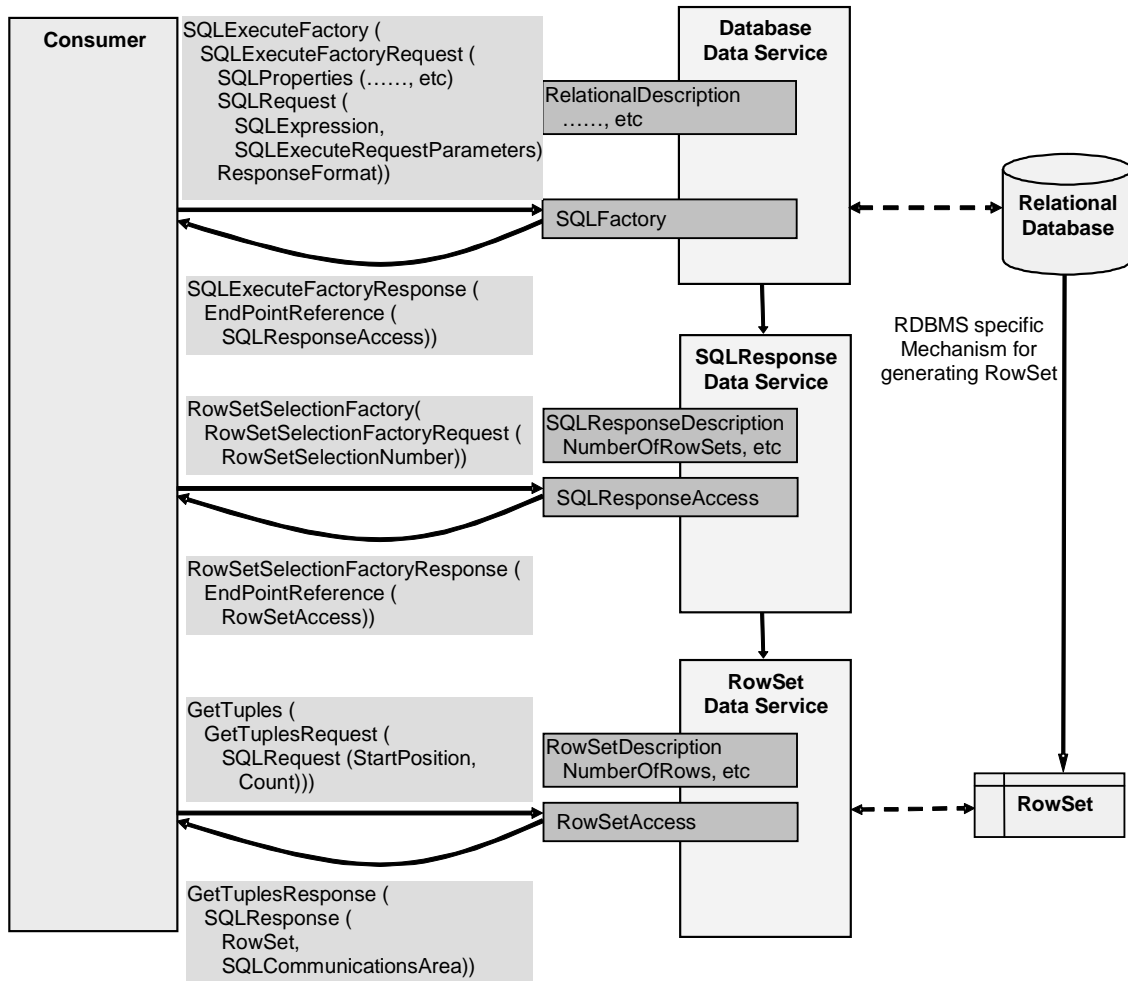


Figure 3 - Overview - SQLFactory::SQLExecuteFactory with SQLResponseAccess::RowSetSelectionFactory with RowSetAccess::GetTuples

This example presents a SQLAccess interface. The SQLExecuteFactory operation is used to construct the derived SQLResponse Data Service which in turn is used to construct the derived RowSet Data Service. This service provides access to tuples in the RowSet resulting from a SQLExpression against the Relational Database.

6.1.3 Operations

6.1.3.1 SQLFactory::SQLExecuteFactory

Create a new Data Service that corresponds to the results of an SQL Query.

Input

- *SQLExecuteFactoryRequest*
 - *SQLProperties* – Behavioral Properties
 - *SQLRequest*
 - *SQLExpression* - (1) – any SQL statement.
 - *SQLExecuteRequestParameters* – (0 to n)
 - *InputParameters* - (0 to n)
 - *OutputParameters* - (0 to n)

- *InOutParameters* - (0 to n)
 - *ResponseFormat* - the format(s), selected from the *SQLExecuteResponseTypes* property which the *EndPointReference* will refer to.

Output

- *SQLExecuteFactoryResponse*
 - *EndPointReference* - (1) – to *SQLResponseAccess* operation.

Faults

- *InvalidSQLExecuteFactoryRequest* - XML syntax error or XML schema non-compliance.
- *InvalidSQLExecuteRequestParameters* - *SQLExecuteRequestParameters* do not match *SQLExpression*.
- *InvalidDataServiceParameters* - Parameters not valid.
- *InvalidResponseFormat* - *ResponseFormat* not valid.
- *OtherFault* – any other fault.

6.1.3.2 *SQLResponseAccess::RowSetSelectionFactory*

Get an *EndPointReference* to a *RowSetAccess* from the *RowSetSelectionFactoryResponse*.

Input

- *RowSetSelectionFactoryRequest*
 - *RowSetSelectionNumber* - (1) – the number of the required *RowSet*.

Output

- *RowSetSelectionFactoryResponse*
 - *EndPointReference* - (1) – to *RowSetAccess* operation which provides access to the requested *RowSet*.

Faults

- *InvalidRowSetSelectionFactoryRequest* - XML syntax error or XML schema non-compliance.
- *InvalidRowSetFactoryNumber* - not a valid *RowSetFactoryNumber*.
- *OtherFault* – any other fault.

7. DataManagement

See the relevant section in the *Web Services Data Access and Integration* document [WS-DAI] for details.

8. Mapping to WSDL

For a mapping to the Web Services Resource Framework (WSRF) proposal see the following Sections:

- *SQLAccess*
 - WSDL Port Types – Appendix A.1
 - XML Schema – Appendix A.2
 - WSDL - Appendix A.3
- *SQLResponseAccess*
 - WSDL Port Types – Appendix B.1
 - XML Schema – Appendix B.2
 - WSDL - Appendix B.3
- *RowSetAccess*
 - WSDL Port Types – Appendix C.1
 - XML Schema – Appendix C.2
 - WSDL - Appendix C.3

9. Security Considerations

The relational realization of a Grid Data Service will use standard Grid Security mechanisms as specified by OGSA Security working group combined with standard ways of relating Grid credentials and authorities to resource access rights. The assumption is that these standards will also indicate how to make information related to authentication, authorization security etc available.

10. Conclusion

This document has discussed a specialization of the interfaces defined in the *Web Services Data Access and Integration* document [WS-DAI] and the additional capabilities required to properly address relational Data Resources. This is work in progress and feedback is welcomed on this document.

Editor Information

Mario Antonioletti,
EPCC,
University of Edinburgh,
James Clerk Maxwell Building,
Mayfield Road,
Edinburgh EH9 3JZ,
United Kingdom.

Brian M Collins
IBM United Kingdom Limited,
Hursley Park,
Winchester,
Hampshire, SO21 2JN,
United Kingdom.

Shannon Hastings,
Ohio State University,
333 W. Tenth Ave.,
Columbus OH, 43210,
USA.

Amy Krause,
EPCC,
University of Edinburgh,
James Clerk Maxwell Building,
Mayfield Road,
Edinburgh EH9 3JZ,
United Kingdom.

Stephen Langella,
Ohio State University,
333 W. Tenth Ave.,
Columbus OH, 43210,
USA.

Simon Laws,
IBM United Kingdom Limited,
Hursley Park,
Winchester,
Hampshire, SO21 2JN,

United Kingdom.

James Magowan,
IBM United Kingdom Limited,
Hursley Park,
Winchester,
Hampshire, SO21 2JN,
United Kingdom.

Susan Malaika,
IBM Corporation,
Silicon Valley Laboratory,
555 Bailey Avenue,
San Jose, CA 95141,
USA.

Norman W. Paton,
Department of Computer Science,
University of Manchester,
Oxford Road,
Manchester M13 9PL,
United Kingdom.

Contributor Information

Vijay Dialani, University of Southampton.
Greg Riccardi, Florida State University.
Shannon Hastings, Ohio State University.
Stephen Langella, Ohio State University.

Acknowledgements

The DAIS Working Group of the Global Grid Forum is active, and many people have contributed to discussions within the group in recent months, including but not limited to: Bill Allcock, Vijay Dialani , Dieter Gawlick, Allen Luniewski , Sastry Malladi, Inderpal Narang, Steve Tuecke , Jay Unger, Paul Watson, Martin Westhead, Patrick Dantressangle.

Intellectual Property Statement

The GGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the GGF Secretariat.

The GGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights, which may cover technology that may be required to practice this recommendation. Please address the information to the GGF Executive Director.

Full Copyright Notice

Copyright (C) Global Grid Forum (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the GGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the GGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the GGF or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE GLOBAL GRID FORUM DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE."

References

[Data Services]

I. Foster, A. Luniewski, S. Tuecke and J. Unger, *OGSA Data Services*, DAIS-WG Informational Draft, 11th Global Grid Forum, 21st May 2004.

[WS-DAI]

M. Antonioletti, M. Atkinson, S. Laws, S. Malaika, N. W. Paton D. Pearson and G. Riccardi. *Web Services Data Access and Integration (WS-DAI)*. DAIS-WG Informational Draft, 11th Global Grid Forum, 21st May 2004.

[RFC2199]

S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, Internet Engineering Task Force, RFC 2119, <http://www.ietf.org/rfc/rfc2119.txt>, March 1997.

[SQL2003]

Information technology -- Database languages -- SQL -- Part 14: XML-Related Specifications (SQL/XML), ISO/IEC 9075-14:2003,
<http://www.iso.ch/iso/en/stdsdevelopment/tc/tclist/TechnicalCommitteeStandardsListPage.TechnicalCommitteeStandardsList?COMMID=160&printable=true>

[JSR114]

J. Bruce, JSR-000114 JDBC RowSet Implementations, Final Release, 07 April 2004.
<http://jcp.org/aboutJava/communityprocess/final/jsr114/index.html>

[WS-DM MUWS]

A. Dharmawan and W. Vambenepe, *Web Services Distributed Management: Management Using Web Services (WSDM-MUWS 0.5)*, Committee Draft 2 April 2004
<http://www.oasis-open.org/committees/download.php/6234/cd-wsdm-muws-0.5.pdf>

[WS-DM MOWS]

J. DeCarlo and I. Sedukhin, *Web Services Distributed Management: Management Of Web Services (WSDM-MOWS 0.5)*, Committee Draft 2 April 2004
<http://www.oasis-open.org/committees/download.php/6255/cd-wsdm-mows-0.5-20040402.pdf>

Appendix A.1 – SQLAccess WSDL Port Types

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdair"
    targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
        xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
        xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR">

<!-- WSDL IMPORTS ##### -->
    <wsdl:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR" location="./wsdair-types-
0.3.wsdl" />

<!-- WSDL PORT TYPES ##### -->
    <wsdl:portType name="SQLDataService">

        <wsdl:operation name="SQLExecute">
            <wsdl:input message="wsdair:SQLExecuteRequest" />
            <wsdl:output message="wsdair:SQLExecuteResponse" />
        </wsdl:operation>

        <wsdl:operation name="SQLExecuteFactory">
            <wsdl:input message="wsdair:SQLExecuteFactoryRequest" />
            <wsdl:output message="wsdair:SQLExecuteFactoryResponse" />
        </wsdl:operation>

    </wsdl:portType>
</wsdl:definitions>
```

Appendix A.2 – SQLAccess XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR">

    <xsd:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAI" schemaLocation="./wsdai-types-
0.3.xsd" />
```

```
<!-- sql description -->
<!-- the following properties are examples only and are subject to change -->
<!-- the CGS working group is build models that describe the properties -->
<!-- that go here -->
  <xsd:complexType name="RelationalSchemaType">
    <xsd:sequence>
      <xsd:element name="Content?" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

<xsd:element name="RelationalSchema" type="wsdair:RelationalSchemaType" />

  <xsd:complexType name="StoredProcedureListType">
    <xsd:sequence>
      <xsd:element name="Content" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

<xsd:element name="StoredProcedures" type="wsdair:StoredProcedureListType" />

  <xsd:complexType name="UserDefinedTypesListType">
    <xsd:sequence>
      <xsd:element name="Content" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

<xsd:element name="UserDefinedTypes" type="wsdair:UserDefinedTypesListType" />

  <xsd:complexType name="UserDefinedFunctionListType">
    <xsd:sequence>
      <xsd:element name="Content" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

<xsd:element name="UserDefinedFunctions" type="wsdair:UserDefinedFunctionListType" />

  <xsd:complexType name="TriggersListType">
    <xsd:sequence>
      <xsd:element name="Content" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
```

```

<xsd:element name="Triggers" type="wsdair:TriggersListType" />

  <xsd:complexType name="LanguageCapabilitiesType">
    <xsd:sequence>
      <xsd:element name="Content" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:element name="LanguageCapabilities" type="wsdair:LanguageCapabilitiesType" />

<!-- sql access -->
<!-- the terms that control the behaviour of the sql access operations -->
<xsd:complexType name="SQLAccessTermsType">
  <xsd:complexContent>
    <xsd:extension base="wsdai:DataAccessTermsType">
      <xsd:sequence>
        <xsd:element name="LanguageCapabilities" ref="wsdair:LanguageCapabilities" />
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

  <xsd:element name="SQLAccessTerms" type="wsdair:SQLAccessTermsType"/>

  <!-- the term document to be used when creating a -->
<xsd:complexType name="SQLAccessTermDocumentType">
  <xsd:complexContent>
    <xsd:restriction base="wsdai:TermDocumentType">
      <xsd:sequence>
        <xsd:element name="PortType" >
          <xsd:simpleType>
            <xsd:restriction base="xsd:QName">
              <xsd:enumeration value="wsdair:SQLDataService"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="Terms" type="wsdair:SQLAccessTermsType"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>

<xsd:element name="SQLAccessTermDocument"

```

```

        type="wsdair:SQLAccessTermDocumentType"
        substitutionGroup="wsdai:TermDocument" />

<!-- the list of response types validly returned by the sql execute operation -->
<xsd:element name="SQLExecuteResponseTypeList" type="wsdai:ResponseTypeListType" />

<xsd:complexType name="SQLCommunicationsAreaType">
    <xsd:sequence>
        <xsd:element name="SQLState" type="xsd:string" />
        <xsd:element name="VendorCode" type="xsd:string" />
        <xsd:element name="MessageText" type="xsd:string" />
    </xsd:sequence>
</xsd:complexType>

<xsd:element name="SQLCommunicationsArea" type="wsdair:SQLCommunicationsAreaType" />

<!-- sql factory -->
<!-- the list of terms constructs that are valid - this implies the service type -->
<xsd:element name="SQLExecuteFactoryTermDocumentTypeList" type="wsdai:TermDocumentTypeListType" />

<!-- sql management -->
<!-- TBD -->

</xsd:schema>

```

Appendix A.3 – SQLAccess WSDL

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdair"
    targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wrs="http://java.sun.com/xml/ns/jdbc"
    xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR">

<!-- WSDL IMPORTS ##### -->

<!-- WSDL TYPES ##### -->
<xsd:schema targetNamespace="http://java.sun.com/xml/ns/jdbc"

```



```

        elementFormDefault="qualified">
        <xsd:include schemaLocation="./webrowset-jdbc150.xsd" />
</xsd:schema>

<xsd:schema targetNamespace="http://schemas.xmlsoap.org/ws/2004/03/addressing"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsa-0304.xsd" />
</xsd:schema>

<xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAI"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdai-types-0.3.xsd" />
</xsd:schema>

<xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdair-types-0.3.xsd" />

<!-- ##### -->
<!-- ### Common Message Types ### -->
<!-- ##### -->

    <!-- general request types -->
    <xsd:complexType name="SQLExecuteRequestParameterType">
        <xsd:sequence>
            <xsd:element name="Name" type="xsd:string" />
            <xsd:element name="Value" type="xsd:string"/>
            <xsd:element name="Type" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:complexType name="SQLExpressionType">
    <xsd:complexContent>
        <xsd:extension base="wsdai:ExpressionType">
            <xsd:sequence>
                <xsd:element name="Expression" type="xsd:string" minOccurs="1" maxOccurs="1"/>
                <xsd:element name="SQLExecuteRequestParameter"
type="wsdair:SQLExecuteRequestParameterType" minOccurs="0" maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>

```

```

</xsd:complexType>

<xsd:element name="SQLExpression" type="wsdair:SQLExpressionType" abstract="true" />

<!-- general response types -->
<xsd:element name="SQLUpdateCount" type="xsd:int" />
<xsd:element name="SQLOutputParameter" type="xsd:string" />
<xsd:element name="SQLReturnValue" type="xsd:string" />

<xsd:complexType name="SQLDatasetType">
<xsd:complexContent>
  <xsd:extension base="wsdai:DatasetType">
    <xsd:sequence>
      <xsd:element ref="wrs:WebRowSet" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="wsdair:SQLUpdateCount" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="wsdair:SQLOutputParameter" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="wsdair:SQLReturnValue" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="wsdair:SQLCommunicationsArea" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>

<xsd:element name="SQLDataset" type="wsdair:SQLDatasetType" substitutionGroup="wsdai:Dataset"/>

<!-- ##### -->
<!-- ### sqlExecute Message Types ### -->
<!-- ##### -->

  <xsd:element name="SQLExecuteRequest">
    <xsd:complexType >
      <xsd:sequence>
        <xsd:element ref="wsdair:SQLExpression" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="wsdai:ResponseFormat" minOccurs="0" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="SQLExecuteResponse">

```

```

        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="wsdai:Dataset" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

<!-- ##### -->
<!-- ### sqlExecuteFactory Message Types ### -->
<!-- ##### -->
    <xsd:element name="SQLExecuteFactoryRequest">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element ref="wsdair:SQLExpression" minOccurs="1" maxOccurs="1"/>

                <xsd:element ref="wsdai:TermDocument" minOccurs="0" maxOccurs="1" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <!-- assumes that these messages result in a service/resource that contains all of -->
    <!-- the possible responses from a SQL execute (rowset, count, value, parameter etc) -->
    <xsd:element name="SQLExecuteFactoryResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="wsa:EndPointReference" minOccurs="0" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

<!-- ##### -->
<!-- ### Resource Properties ### -->
<!-- ##### -->
    <xsd:element name="RelationalDescription">
        <xsd:complexType>
            <xsd:sequence>
                <!-- from wsdai - data description - properties of the data resource -->
                <xsd:element ref="wsdai:Name" minOccurs="0" maxOccurs="1" />
                <xsd:element ref="wsdai:Description" minOccurs="0" maxOccurs="1"/>

                <!-- from wsdaair - sql description - properties of the data resource -->

```

```

        <xsd:element ref="wsdair:RelationalSchema" minOccurs="1" maxOccurs="1" />
        <xsd:element ref="wsdair:StoredProcedures" minOccurs="1" maxOccurs="1" />
        <xsd:element ref="wsdair:UserDefinedTypes" minOccurs="1" maxOccurs="1" />
        <xsd:element ref="wsdair:UserDefinedFunctions" minOccurs="1" maxOccurs="1" />
        <xsd:element ref="wsdair:Triggers" minOccurs="1" maxOccurs="1" />

        <!-- from wsdaire - sql access - properties controlling access behaviour -->
        <xsd:element ref="wsdair:SQLAccessTerms" minOccurs="1" maxOccurs="1" />

        <!-- from wsdaire - sql access - properties controlling valid response formats -->
        <xsd:element ref="wsdair:SQLExecuteResponseTypeList" minOccurs="1" maxOccurs="1" />

        <!-- from wsdaire - sql factory - properties controlling valid term document
types-->
        <xsd:element ref="wsdair:SQLExecuteFactoryTermDocumentTypeList" minOccurs="1"
maxOccurs="1" />
    </xsd:sequence>
</xsd:complexType>
</xsd:element>

</xsd:schema>

<!-- WSDL MESSAGES ##### -->

<!-- ##### -->
<!-- ### sqlExecute Messages ### -->
<!-- ##### -->
<message name="SQLExecuteRequest">
    <part name="SQLExecuteRequest" element="wsdair:SQLExecuteRequest" />
</message>

<message name="SQLExecuteResponse">
    <part name="SQLExecuteResponse" element="wsdair:SQLExecuteResponse" />
</message>

<!-- ##### -->
<!-- ### sqlExecuteFactory Messages ### -->
<!-- ##### -->
<message name="SQLExecuteFactoryRequest">
    <part name="SQLExecuteFactoryRequest" element="wsdair:SQLExecuteFactoryRequest" />
</message>

```

```

        <message name="SQLExecuteFactoryResponse">
            <part name="SQLExecuteFactoryResponse" element="wsdair:SQLExecuteFactoryResponse" />
        </message>

</wsdl:definitions>

```

Appendix B.1 – SQLResponseAccess WSDL Port Types

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdairs"
    targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAISR"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:wsdaisr="http://www.ggf.org/namespaces/2004/05/WS-DAISR">

<!-- WSDL IMPORTS ##### -->
    <wsdl:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
        location="./wsdaisr-types-0.3.wsdl" />

<!-- WSDL PORT TYPES ##### -->
    <wsdl:portType name="SQLResponseDataService">

        <wsdl:operation name="GetRowSet">
            <wsdl:input message="wsdaisr:GetRowSetRequest" />
            <wsdl:output message="wsdaisr:GetRowSetResponse" />
        </wsdl:operation>

        <wsdl:operation name="RowSetSelectionFactory">
            <wsdl:input message="wsdaisr:RowSetSelectionFactoryRequest" />
            <wsdl:output message="wsdaisr:RowSetSelectionFactoryResponse" />
        </wsdl:operation>

        <wsdl:operation name="GetUpdateCount">
            <wsdl:input message="wsdaisr:GetUpdateCountRequest" />
            <wsdl:output message="wsdaisr:GetUpdateCountResponse" />
        </wsdl:operation>

        <wsdl:operation name="GetReturnValue">

```

```

        <wsdl:input message="wsdaisr:GetReturnValueRequest" />
        <wsdl:output message="wsdaisr:GetReturnValueResponse" />
    </wsdl:operation>

    <wsdl:operation name="GetOutputParameter">
        <wsdl:input message="wsdaisr:GetOutputParameterRequest" />
        <wsdl:output message="wsdaisr:GetOutputParameterResponse" />
    </wsdl:operation>

    <wsdl:operation name="GetSQLCommunicationsArea">
        <wsdl:input message="wsdaisr:GetSQLCommunicationsAreaRequest" />
        <wsdl:output message="wsdaisr:GetSQLCommunicationsAreaResponse" />
    </wsdl:operation>

</wsdl:portType>

</wsdl:definitions>

```

Appendix B.2 – SQLResponseAccess XML Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAISR"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wrs="http://java.sun.com/xml/ns/jdbc"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdaisr="http://www.ggf.org/namespaces/2004/05/WS-DAISR"

    <xsd:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAI" schemaLocation="./wsdai-types-0.3.xsd" />

    <!-- sql response description -->
    <xsd:element name="NumberOfRowSets" type="xsd:int" />
    <xsd:element name="NumberOfUpdateCounts" type="xsd:int" />
    <xsd:element name="NumberOfReturnValues" type="xsd:int" />
    <xsd:element name="NumberOfOutputParameters" type="xsd:int" />
    <xsd:element name="NumberOfSQLCommunicationsAreas" type="xsd:int" />

    <!-- sql response access -->

    <xsd:complexType name="SQLResponseAccessTermsType">
        <xsd:complexContent>
            <xsd:extension base="wsdai:DataAccessTermsType">

```

```

        <xsd:sequence>
            <xsd:element ref="TBD" minOccurs="1" maxOccurs="1"/>
        </xsd:sequence>
    </xsd:extension>
</xsd:complexContent>
</xsd:complexType>

<xsd:element name="SQLResponseAccessTerms" type="SQLResponseAccessTermsType"/>

<xsd:complexType name="SQLResponseAccessTermDocumentType">
    <xsd:complexContent>
        <xsd:restriction base="wsdai:TermDocumentType">
            <xsd:sequence>
                <xsd:element name="PortType" >
                    <xsd:simpleType>
                        <xsd:restriction base="xsd:QName">
                            <xsd:enumeration value="wsdaisr:SQLResponseDataService"/>
                        </xsd:restriction>
                    </xsd:simpleType>
                </xsd:element>
                <xsd:element name="Terms" type="wsdaisr:SQLResponseAccessTermsType"/>
            </xsd:sequence>
        </xsd:restriction>
    </xsd:complexContent>
</xsd:complexType>

<xsd:element name="SQLResponseAccessTermDocument"
    type="wsdaisr:SQLResponseAccessTermDocumentType"
    substitutionGroup="wsdai:TermDocument"/>

<!-- the list of response types validly returned by the get tuples operation -->
<xsd:element name="GetRowSetResponseTypeList" type="wsdai:ResponseTypeListType"/>

<!-- sql response factory -->

<!-- sql response management -->
</xsd:schema>

```

Appendix B.3 – SQLResponseAccess WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdairs"
    targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAISR"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:wsdairs="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
    xmlns:wsdairsr="http://www.ggf.org/namespaces/2004/05/WS-DAISR">

<!-- WSDL IMPORTS ##### -->

<!-- WSDL TYPES ##### -->
    <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAI"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdai-types-0.3.xsd" />
    </xsd:schema>

    <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdair-types-0.3.xsd" />
    </xsd:schema>

    <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdairs-types-0.3.xsd" />
    </xsd:schema>

    <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAISR"
        elementFormDefault="qualified">
        <xsd:include schemaLocation="./wsdairsr-types-0.3.xsd" />
    </xsd:schema>

<!-- ##### -->
<!-- ### Common Message Types ### -->
<!-- ##### -->
    <xsd:complexType name="RowSetDatasetType">
        <xsd:complexContent>
            <xsd:extension base="wsdai:DatasetType">
                <xsd:sequence>
                    <xsd:element ref="wrs:WebRowSet" minOccurs="0" maxOccurs="unbounded"/>
                </xsd:sequence>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>

```



```

        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>

<!-- ##### -->
<!-- ### GetRowSet Message Types ### -->
<!-- ##### -->
    <xsd:element name="GetRowSetRequest">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element name="RowSetNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
                <xsd:element ref="wsdai:ResponseFormat" minOccurs="0" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="GetRowSetResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="wsdai:Dataset" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

<!-- ##### -->
<!-- ### RowSetSelectionFactory Message Types ### -->
<!-- ##### -->
    <xsd:element name="RowSetSelectionFactoryRequest">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element name="RowSetNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
                <xsd:element ref="wsdai:TermDocument" minOccurs="0" maxOccurs="1" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="RowSetSelectionFactoryResponse">
        <xsd:complexType>
            <xsd:sequence>

```

```

        <xsd:element ref="wsa:EndPointReference" minOccurs="0" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ##### -->
<!-- ### GetUpdateCount Message Types ### -->
<!-- ##### -->
  <xsd:element name="GetUpdateCountRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="UpdateCountNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="GetUpdateCountResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="UpdateCount" type="xsd:int" minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

<!-- ##### -->
<!-- ### GetReturnValue Message Types ### -->
<!-- ##### -->
  <xsd:element name="GetReturnValueRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ReturnValueNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

  <xsd:element name="GetReturnValueResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="ReturnValue" type="?" minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>

```

```

        </xsd:complexType>
    </xsd:element>

<!-- ##### -->
<!-- ### GetOutputParameter Message Types ### -->
<!-- ##### -->
    <xsd:element name="GetOutputParameterRequest">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element name="OutputParameterNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="GetOutputParameterResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="OutputParameter" type="?" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

<!-- ##### -->
<!-- ### GetSQLCommunicationsArea Message Types ### -->
<!-- ##### -->
    <xsd:element name="GetSQLCommunicationsAreaRequest">
        <xsd:complexType >
            <xsd:sequence>
                <xsd:element name="SQLCommunicationsAreaNumber" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:element name="GetSQLCommunicationsAreaResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element ref="wsdair:SQLCommunicationsArea" minOccurs="1" maxOccurs="1"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

```

```

<!-- ##### -->
<!-- ### Resource Properties      ### -->
<!-- ##### -->
    <xsd:element name="SQLResponseDescription">
        <xsd:complexType>
            <xsd:sequence>
                <!-- from wsdai - data description - properties of the data resource -->
                <xsd:element ref="wsdai:Name" minOccurs="0" maxOccurs="1" />
                <xsd:element ref="wsdai:Description" minOccurs="0" maxOccurs="1"/>

                <!-- from wsdaistr - sql response description - properties of the data resource -
->
                <xsd:element ref="wsdaistr:NumberOfRowSets" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="wsdaistr:NumberOfUpdateCounts" minOccurs="1" maxOccurs="1"/>

                <xsd:element ref="wsdaistr:NumberOfReturnValues" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="wsdaistr:NumberOfOutputParameters" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="wsdaistr:NumberOfSQLCommunicationsAreas" minOccurs="1"
maxOccurs="1"/>

                <!-- from wsdaistr - sql response access - properties controlling access behaviour --
>
                <xsd:element ref="wsdaistr:SQLResponseAccessTerms" minOccurs="1" maxOccurs="1"/>

                <!-- from wsdaistr - sql response access - properties controlling valid response
formats -->
                <xsd:element ref="wsdaistr:RowSetSelectionFactoryResponseTypeList" minOccurs="1"
maxOccurs="1" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

</xsd:schema>

<!-- WSDL MESSAGES ##### -->

<!-- ##### -->
<!-- ### GetRowSet Messages ### -->
<!-- ##### -->
<message name="GetRowSetRequest">
    <part name="GetRowSetRequest" element="wsdaistr:GetRowSetRequest" />

```

```

</message>

<message name="GetRowSetResponse">
  <part name="GetRowSetResponse" element="wsdaisr:GetRowSetResponse" />
</message>

<!-- ##### -->
<!-- ### RowSetSelectionFactory Messages ### -->
<!-- ##### -->
<message name="RowSetSelectionFactoryRequest">
  <part name="RowSetSelectionFactoryRequest" element="wsdaisr:RowSetSelectionFactoryRequest" />
</message>

<message name="RowSetSelectionFactoryResponse">
  <part name="RowSetSelectionFactoryResponse" element="wsdaisr:RowSetSelectionFactoryResponse" />
</message>

<!-- ##### -->
<!-- ### GetUpdateCount Messages ### -->
<!-- ##### -->
<message name="GetUpdateCountRequest">
  <part name="GetUpdateCountRequest" element="wsdaisr:GetUpdateCountRequest" />
</message>

<message name="GetUpdateCountResponse">
  <part name="GetUpdateCountResponse" element="wsdaisr:GetUpdateCountResponse" />
</message>

<!-- ##### -->
<!-- ### GetReturnValue Messages ### -->
<!-- ##### -->
<message name="GetReturnValueRequest">
  <part name="GetReturnValueRequest" element="wsdaisr:GetReturnValueRequest" />
</message>

<message name="GetReturnValueResponse">
  <part name="GetReturnValueResponse" element="wsdaisr:GetReturnValueResponse" />
</message>

<!-- ##### -->
<!-- ### GetOutputParameter Messages ### -->
<!-- ##### -->

```

```

    <message name="GetOutputParameterRequest">
      <part name="GetOutputParameterRequest" element="wsdair:GetOutputParameterRequest" />
    </message>

    <message name="GetOutputParameterResponse">
      <part name="GetOutputParameterResponse" element="wsdair:GetOutputParameterResponse" />
    </message>

    <!-- ##### -->
    <!-- ### GetSQLCommunicationsArea Messages ### -->
    <!-- ##### -->
    <message name="GetSQLCommunicationsAreaRequest">
      <part name="GetSQLCommunicationsAreaRequest" element="wsdair:GetSQLCommunicationsAreaRequest"
/>
    </message>

    <message name="GetSQLCommunicationsAreaResponse">
      <part name="GetSQLCommunicationsAreaResponse"
element="wsdair:GetSQLCommunicationsAreaResponse" />
    </message>

</wsdl:definitions>

```

Appendix C.1 – RowSetAccess WSDL Port Types

```

<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdairs"
  targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
  xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
  xmlns:wsdairs="http://www.ggf.org/namespaces/2004/05/WS-DAIRS">

  <!-- WSDL IMPORTS ##### -->
  <wsdl:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
    location="./wsdairs-types-0.3.wsdl" />

  <!-- WSDL PORT TYPES ##### -->
  <wsdl:portType name="RowSetDataService">

    <wsdl:operation name="GetTuples">

```

```

        <wsdl:input message="wsdairs:GetTuplesRequest" />
        <wsdl:output message="wsdairs:GetTuplesResponse" />
    </wsdl:operation>

</wsdl:portType>

</wsdl:definitions>

```

Appendix C.2 – RowSetAccess XML Schema

```

<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wrs="http://java.sun.com/xml/ns/jdbc"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdairs="http://www.ggf.org/namespaces/2004/05/WS-DAIRS">

    <xsd:import namespace="http://www.ggf.org/namespaces/2004/05/WS-DAI" schemaLocation="./wsdai-types-
0.3.xsd" />

    <!-- rowset description -->
    <xsd:complexType name="RowSetSchemaType">
        <xsd:sequence>
            <xsd:element ref="wrs:metadata"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="RowSetSchema" type="wsdairs:RowSetSchemaType" />

    <xsd:element name="NumberOfRows" type="xsd:int" />

    <!-- rowset access -->
    <xsd:element name="CursorDirection">
        <xsd:simpleType>
            <xsd:restriction base="xsd:token">
                <xsd:enumeration value="ForwardOnly"/>
                <xsd:enumeration value="ForwardAndReverse"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>

    <xsd:element name="CursorHeldOverTxnBoundary" type="xsd:boolean" />

```

```

<xsd:complexType name="RowSetAccessTermsType">
  <xsd:complexContent>
    <xsd:extension base="wsdai:DataAccessTermsType">
      <xsd:sequence>
        <xsd:element ref="wsdairs:CursorDirection" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="wsdairs:CursorHeldOverTxnBoundary" minOccurs="1" maxOccurs="1"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name="RowSetAccessTerms" type="RowSetAccessTermsType"/>

<xsd:complexType name="RowSetAccessTermDocumentType">
  <xsd:complexContent>
    <xsd:restriction base="wsdai:TermDocumentType">
      <xsd:sequence>
        <xsd:element name="PortType" >
          <xsd:simpleType>
            <xsd:restriction base="xsd:QName">
              <xsd:enumeration value="wsdairs:RowSetDataService"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element name="Terms" type="wsdairs:RowSetAccessTermsType"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>

<xsd:element name="RowSetAccessTermDocument"
  type="wsdairs:RowSetAccessTermDocumentType"
  substitutionGroup="wsdai:TermDocument"/>

<!-- the list of response types validly returned by the get tuples operation -->
<xsd:element name="GetTuplesResponseTypeList" type="wsdai:ResponseTypeListType"/>

<!-- rowset factory -->

<!-- rowset management -->

```



```
</xsd:schema>
```

Appendix C.3 – RowSetAccess WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="wsdairs"
  targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:wsdai="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    xmlns:wsdair="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    xmlns:wsdairs="http://www.ggf.org/namespaces/2004/05/WS-DAIRS">

<!-- WSDL IMPORTS ##### -->

<!-- WSDL TYPES ##### -->
  <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAI"
    elementFormDefault="qualified">
    <xsd:include schemaLocation="./wsdai-types-0.3.xsd" />
  </xsd:schema>

  <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIR"
    elementFormDefault="qualified">
    <xsd:include schemaLocation="./wsdair-types-0.3.xsd" />
  </xsd:schema>

  <xsd:schema targetNamespace="http://www.ggf.org/namespaces/2004/05/WS-DAIRS"
    elementFormDefault="qualified">
    <xsd:include schemaLocation="./wsdairs-types-0.3.xsd" />

  <!-- ##### -->
  <!-- ### Common Message Types ### -->
  <!-- ##### -->

  <!-- ##### -->
  <!-- ### GetTuples Message Types ### -->
  <!-- ##### -->
    <xsd:element name="GetTuplesRequest">
      <xsd:complexType>
        <xsd:sequence>
```

```

        <xsd:element name="StartPosition" type="xsd:int" minOccurs="1"
maxOccurs="1"/>
        <xsd:element name="Count" type="xsd:int" minOccurs="1" maxOccurs="1"/>
        <xsd:element ref="wsdai:ResponseFormat" minOccurs="0" maxOccurs="1"/>

    </xsd:sequence>
</xsd:complexType>
</xsd:element>

<xsd:element name="GetTuplesResponse">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element ref="wsdai:Dataset" minOccurs="1" maxOccurs="1"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<!-- ##### -->
<!-- ### Resource Properties ### -->
<!-- ##### -->
    <xsd:element name="RowSetDescription">
        <xsd:complexType>
            <xsd:sequence>
                <!-- from wsdai - data description - properties of the data resource -->
                <xsd:element ref="wsdai:Name" minOccurs="0" maxOccurs="1" />
                <xsd:element ref="wsdai:Description" minOccurs="0" maxOccurs="1"/>

                <!-- from wsdaairs - rowset description - properties of the data resource -->

                <xsd:element ref="wsdaairs:RowsSchema" minOccurs="1" maxOccurs="1"/>
                <xsd:element ref="wsdaairs:NumberOfRows" minOccurs="1" maxOccurs="1" />

                <!-- from wsdaairs - rowset access - properties controlling access behaviour -->
                <xsd:element ref="wsdaairs:RowSetAccessTerms" minOccurs="1" maxOccurs="1"/>

                <!-- from wsdaairs - rowset access - properties controlling valid response formats --
>
                <xsd:element ref="wsdaairs:GetTuplesResponseTypeList" minOccurs="1" maxOccurs="1" />
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

```

```
</xsd:schema>

<!-- WSDL MESSAGES ##### -->

    <!-- ##### -->
    <!-- ### GetTuples Messages ### -->
    <!-- ##### -->
    <message name="GetTuplesRequest">
        <part name="GetTuplesRequest" element="wsdairs:GetTuplesRequest" />
    </message>

    <message name="GetTuplesResponse">
        <part name="GetTuplesResponse" element="wsdairs:GetTuplesResponse" />
    </message>

</wsdl:definitions>
```