

Open Data Analytics for z/OS
Version 1 Release 1

Solutions Guide



Note

Before using this information and the product it supports, read the information in [“Notices” on page 149](#).

This edition applies to Version 1 Release 1 of IBM® Open Data Analytics for z/OS® (5655-OD1) and to all subsequent releases and modifications until otherwise indicated in new editions.

Last updated: 2019-02-18

© **Copyright International Business Machines Corporation 2016, 2019.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

© **Rocket Software, Inc. 2016, 2019.**

Contents

List of Figures.....	v
List of Tables.....	vii
About this information.....	ix
How to send your comments to IBM.....	xi
If you have a technical problem.....	xi
Summary of changes for IBM Open Data Analytics for z/OS Solutions Guide.....	xii
Chapter 1. The Data Service SQL solution.....	1
Connectivity overview.....	1
Configuring access to data in Adabas.....	2
Configuring the server started task JCL.....	2
Modifying the server configuration member.....	3
Configuring Adabas security.....	4
Configuring access to data in relational database management systems (RDBMS).....	5
Configuring access to IBM DB2 for z/OS.....	6
Configuring access to distributed databases.....	20
Controlling display and access for native Db2 subsystems	33
Configuring access to CA IDMS data.....	34
Configuring the server started task JCL.....	34
Modifying the server configuration member for CA IDMS.....	34
Verifying access to CA IDMS data.....	35
Configuring access to data in IBM IMS databases.....	36
Configuring the server started task JCL.....	37
Modifying the server configuration member for DBCTL.....	37
Modifying the server configuration member for IMS Direct.....	38
Configuring access to IBM MQ.....	42
Configuring the server started task JCL.....	42
Modifying the server configuration member for IBM MQ.....	43
Configuring virtual table rules for IBM MQ.....	43
Configuring access to sequential files.....	44
Reading ahead tracks for sequential file access.....	45
Configuring access to native VSAM.....	45
Verifying access to native VSAM.....	46
Modifying the data and index buffer values for VSAM files.....	46
Configuring access to zFS files.....	48
Configuring access to SMF data for IT Operational Analytics.....	48
Configuring access to System Management Facility (SMF) files.....	49
Configuring access to SYSLOG files.....	52
Configuring access to OPERLOG files.....	53
Configuring access to ADDI.....	53
Installing virtual tables and virtual target maps for ADDI access.....	54
Modifying the configuration member for ADDI access.....	55
Configuring virtual table rules for ADDI.....	58
Configuring authentication for ADDI.....	59
Configuring access to RAA.....	60
Installing virtual tables and virtual target maps for RAA access.....	60
Modifying the configuration member for RAA access.....	61
Configuring virtual table rules for RAA.....	64
Configuring authentication for RAA.....	64

Using Db2 for z/OS to access multiple data source types.....	65
--	----

Chapter 2. Spark SQL data access: JDBC..... 69

Connecting to a data source using JDBC	70
JDBC connection properties.....	70
Error handling.....	85
Debugging and tracing.....	85
Connecting to a DRDA database server.....	87
JDBC performance management.....	88
Buffering data.....	89
Parallel IO.....	90
MapReduce.....	93
JDBC driver APIs.....	95

Chapter 3. JDBC Gateway..... 101

Using the JDBC Gateway administrative console.....	102
Configuring access to data sources using the JDBC Gateway.....	103
Adding JDBC driver information for a data source.....	103
Creating a data source definition entry.....	105
Configuring the Data Service server for JDBC Gateway sources.....	106
Example: Configuring access to Oracle data.....	109
Example: Configuring access to Hadoop data.....	110
Setting preferences.....	112
Setting JDBC driver preferences.....	112
Setting log preferences.....	112
Setting output preferences.....	113
Troubleshooting.....	113

Chapter 4. Creating your first Apache Spark program to access your z/OS data....115

Chapter 5. Accessing data from Python applications..... 117

Appendix A. DFStor support.....119

Appendix B. DVS_MAP API.....121

Syntax.....	121
XML schema for DMF	123
Example: SAMPLE-REC.....	142
Example: STAFF-RECORD.....	144

Accessibility.....147

Notices.....149

Trademarks.....	150
-----------------	-----

Index..... 151

List of Figures

- 1. JDBC driver sends data requests using multiple buffers..... 89
- 2. Server returns data one buffer at a time..... 90
- 3. Parallel IO pre-fetches buffers..... 91
- 4. Materializing prefetched row data..... 92

List of Tables

- 1. Partial list of supported data sources..... 2
- 2. SQL parameters in group PRODSQL..... 19
- 3. SQL parameters in group PRODTRACE..... 19
- 4. CA IDMS EMPDEMO database maps..... 35
- 5. CA IDMS network schema IDMSNTWK maps..... 36
- 6. JDBC connection properties..... 71

About this information

This information supports IBM Open Data Analytics for z/OS (5655-OD1) and contains information about the Mainframe Data Service (Data Service) SQL solution.

The Data Service is available with IBM Open Data Analytics for z/OS.

Purpose of this information

This information describes how to configure the Data Service for ANSI SQL access to your mainframe data.

Who should read this information

This information is intended for system and database administrators.

How to send your comments to IBM

We invite you to submit comments about the z/OS product documentation. Your valuable feedback helps to ensure accurate and high-quality information.

Important: If your comment regards a technical question or problem, see instead [“If you have a technical problem”](#) on page xi.

Submit your feedback by using the appropriate method for your type of comment or question:

Feedback on z/OS function

If your comment or question is about z/OS itself, submit a request through the [IBM RFE Community](http://www.ibm.com/developerworks/rfe/) (www.ibm.com/developerworks/rfe/).

Feedback on IBM Knowledge Center function

If your comment or question is about the IBM Knowledge Center functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Knowledge Center Support at ibmkc@us.ibm.com.

Feedback on the z/OS product documentation and content

If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to mhvrfs@us.ibm.com. We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The following deliverable title and order number: Open Data Analytics for z/OS Solutions Guide, SC27-9036-00
- The section title of the specific information to which your comment relates
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

If you have a technical problem

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the [IBM Support Portal](http://support.ibm.com) (support.ibm.com).
- Contact your IBM service representative.
- Call IBM technical support.

Summary of changes for IBM Open Data Analytics for z/OS Solutions Guide

The following changes are made to Version 1 Release 1.

March 2019

- *Db2 Virtualization* is a new feature that provides single-point access to various data source types. See [“Using Db2 for z/OS to access multiple data source types”](#) on page 65.
- The *JDBC Gateway* is a Data Service distributed application server that allows direct connectivity to JDBC data sources. See [Chapter 3, “JDBC Gateway,”](#) on page 101.
- Generation data group (GDG) data set names are now supported when processing SYSLOG files. See [“Configuring access to SYSLOG files”](#) on page 52.
- Information has been added about verifying that OPERLOG is active in a system logger log stream. See [“Configuring access to OPERLOG files”](#) on page 53.
- You can use a virtual table rule to set the value used by the server for the SET CURRENT QUERY ACCELERATION command to any of the CURRENT QUERY ACCELERATION valid values. See [“Setting CURRENT QUERY ACCELERATION”](#) on page 14.

September 2018

- You can control whether native Db2 database subsystems appear in ISPF and the Data Service Studio and if attempts to connect to native Db2 subsystems are allowed. See [“Controlling display and access for native Db2 subsystems”](#) on page 33.
- Virtual table rule support is provided for specifying the number of tracks to read ahead (MULTACC) when reading sequential data sets for individual requests. See [“Reading ahead tracks for sequential file access”](#) on page 45.
- Virtual table rule support is provided for overriding data buffer and index buffer values for VSAM files for individual requests. See [“Modifying the data and index buffer values for VSAM files”](#) on page 46.
- When streaming SMF data, the requester can use a SQL SELECT statement to stream SMF data in real time, directly from the SMF in-memory buffer. The connection to the SMF in-memory resource is made at the time of the request, and the SQL statement does not reach end of data until the server is stopped or the request is canceled. See [“Configuring access to SMF data for IT Operational Analytics”](#) on page 48 and [“Configuring access to System Management Facility \(SMF\) files”](#) on page 49.
- Dynamic data set names are now supported when processing SYSLOG files. See [“Configuring access to SYSLOG files”](#) on page 52.
- The DVS_MAP API can be used to call the Data Mapping Facility (DMF) to map proprietary data as well as sample maps. See [Appendix B, “DVS_MAP API,”](#) on page 121.

June 2018

- Db2 Direct is a new Data Service server access method used to access Db2 data by reading the data in the underlying Db2 VSAM linear data sets directly. See [“Db2 for z/OS data access methods”](#) on page 16 and [“Configuring Db2 Direct”](#) on page 18.
- The process of creating maps to access VSAM and sequential data has been simplified by support of the following methods:
 - Querying information in the IBM Application Discovery and Delivery Intelligence (ADDI) dictionary. See [“Configuring access to ADDI”](#) on page 53.
 - Querying information in the IBM Rational Asset Analyzer (RAA) dictionary. See [“Configuring access to RAA”](#) on page 60.

April 2018

- The following enhancements are available when configuring access to IBM DB2 for z/OS:
 - Using a virtual table rule, you can disable the SET CURRENT QUERY ACCELERATION command for DB2 virtual tables. See [“Setting CURRENT QUERY ACCELERATION”](#) on page 14.
 - SQL query access to DB2 unload data sets is now provided. See [“Configuring access to DB2 unload data sets”](#) on page 15.
- DRDA authentication has been enhanced to support encrypted passwords and create a global default user ID, as described in the following topics:
 - [“Configuring rules and authentication for Big SQL”](#) on page 26
 - [“Configuring rules and authentication for dashDB”](#) on page 27
 - [“Configuring rules and authentication for LUW databases”](#) on page 28
 - [“Configuring rules and authentication for Microsoft SQL Server”](#) on page 29
 - [“Configuring rules and authentication for Oracle DRDA”](#) on page 30
 - [“Configuring rules and authentication for QMF DRDA Server”](#) on page 31
- IMS Direct now supports access to multiple IMS subsystems and calls to compression exits and Guardium encryption and decryption exits. See [“Modifying the server configuration member for IMS Direct”](#) on page 38.
- SQL access to IBM MQ is now provided. See [“Configuring access to IBM MQ”](#) on page 42.
- SMF data in compressed log streams is now supported. See [“Configuring access to System Management Facility \(SMF\) files”](#) on page 49.
- Applications executing on the same sysplex as the server are not required to specify logon credentials when invoking the JDBC driver. See [“Connecting to a data source using JDBC ”](#) on page 70.
- When connecting to the Data Service server using the JDBC driver, password phrase authentication is supported. User ID encoding is also supported between the driver and the Data Service server. See [“JDBC connection properties”](#) on page 70.

Chapter 1. The Data Service SQL solution

To enable ANSI SQL access to mainframe data sources for IBM Open Data Analytics for z/OS, configure Mainframe Data Service (Data Service).

Data Service provides the following features:

- A high-speed conduit between Spark and z/OS data sources and a highly-optimized data flow, which occurs by increasing data agility between Spark and z Systems mainframe environments. Data Service also enables Spark clusters to consume data from mainframe and non-mainframe data sources while supporting simultaneous threads from Spark on IBM z Systems, Spark on Linux on z Systems, and Spark on Linux.
- An entry point for Python-based applications to access z/OS data, providing data scientists with a comprehensive solution for bringing the compute to the data. Accessing z/OS data from Python applications requires the IzODA Anaconda component and use of the Python dsdbc module. For more information, see [Chapter 5, “Accessing data from Python applications,” on page 117](#).

Data Service connectivity options for data consumers includes ANSI 92-SQL.

You must configure one or more data sources to use the Data Service solution.

Configuring a solution can include one or more of the following tasks:

- Configure the started task JCL by modifying the AZKS (subsystem default ID) member that is in the *hlq.SAZKCNTL* library.
- Configure the server member that is included in *hlq.AZKS.SAZKEEXEC(AZKSIN00)*. The server configuration member is shipped in data set member *hlq.SAZKEEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications. See "Creating system data sets" in the *Installation and Customization Guide*.
- Make definition changes in the data provider interface.

Before configuring the SQL solution, the Data Service server installation must be successfully completed.

After configuring the SQL solution, SQL data can be accessed in real time by using the JDBC driver or the Python dsdbc module, depending on your application needs.

You can also use the Data Service Studio to get SQL access to your data. See the *IBM Open Data Analytics for z/OS User's Guide* for details about accessing data using SQL.

For information about configuring the Data Service SQL solution, see the following topics.

Connectivity overview

Data Service supports a broad range of data sources, including mainframe relational and non-relational databases, and distributed databases running on Linux, UNIX, and Windows platforms.

Data Service connectivity options for data consumers includes ANSI 92-SQL (JDBC).

Table 1: Partial list of supported data sources

Data support	Data source
Mainframe relational/non-relational databases and file structures	• IBM® DB2 • IBM® Information Management System (IMS/DB) • Native VSAM files • Sequential files • Software AG Adabas
Mainframe applications and screens	• IDMS • IBM® Information Management System • Software AG Natural
Distributed data stores running on Linux, UNIX, and Windows platforms	• IBM® BigInsights Hadoop • IBM® DB2 • Apache Derby • IBM® Informix • Oracle • Microsoft SQL Server

SQL connectivity

SQL connectivity provides ANSI 92-SQL access through mainframe data virtualization. You can use it to transform mainframe artifacts into a familiar, easy-to-use relational format. With industry standard ANSI 92-SQL access, Data Service makes critical mainframe data readily available for analytic purposes.

Configuring access to data in Adabas

To access Adabas, you need to configure the started task JCL and the server configuration member.

Before you begin

The server must be installed.

Procedure

To configure access to data in Adabas, perform the following tasks:

- Configure the server started task. See [“Configuring the server started task JCL”](#) on page 2.
- Modify the server configuration member. See [“Modifying the server configuration member”](#) on page 3.
- Configure security access to Adabas data. See [“Configuring Adabas security”](#) on page 4.

Configuring the server started task JCL

Make the ADALNKR module available.

Before you begin

All LOAD library data sets allocated to the Data Service server in the server started task JCL must be APF-authorized.

About this task

Note: You can skip this task if the ADALNKR module is in the z/OS linklist.

Procedure

1. Add the Adabas LOAD library to the server started task JCL. Uncomment the ADALOAD parameter and set it to the correct Adabas load library name.

```
ADALOAD= ' ADABAS . LOAD '
```

2. Uncomment the reference to the LOAD library in the STEPLIB.

Modifying the server configuration member

Enable the Adabas parameters in the server configuration member.

About this task

The server configuration member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. In the AZKSIN00 member, locate the comment “ENABLE ADABAS DATABASE SERVER SUPPORT.”
2. Enable the Adabas parameters by changing the syntax `if DontDoThis` to `if DoThis`.

Set the ADABAS parameter to YES. The following example shows the section in the configuration member to enable:

```
if DoThis then
do
    "MODIFY PARM NAME(ADABAS)VALUE(YES)"
    "MODIFY PARM NAME(ADABASUBINFOSIZE)VALUE(256)"
    "MODIFY PARM NAME(ADABASAUTOCOMMITBIND)VALUE(YES)"

    "MODIFY PARM NAME(ACIMAPREDUCEADAB)VALUE(64000)"
    "MODIFY PARM NAME(ACIMAPREDUCEADAISN)VALUE(64000)"
end
```

The following table lists the parameters for configuring support for Adabas data stores:

Parameter	Description	Valid values
ACIMAPREDUCEADAB	Map Reduce Adabas Record Buffer Size - Allows Adabas Multi-Fetch used to read records via L1 commands. If the Adabas ADARUN limits are exceeded, an Adabas response code 53 is issued.	Buffer size in bytes. 64000 (default value)
ACIMAPREDUCEADAISN	Map Reduce Adabas ISN Buffer Size - When a Key Descriptor is used in a Search query, an Adabas S1 search is performed. The resulting internal sequence number (ISN) Record number list is divided up into separate Map Reduce threads.	Buffer size in bytes. 64000 (default value)

Parameter	Description	Valid values
ADABAS	Activates support for Adabas data stores.	NO Support is not active. (default value) YES Activate support.
ADABASAUTOCOMMITBIND	Activates support for the AUTOCOMMIT BIND option.	YES Activate support. (default value) NO Support is not active.
ADABASUBINFOSIZE	Specifies the total amount of space to allocate for user information and review information in the Adabas user block. Review the maximum user information size in the ADALNKR, and increase the value of this parameter to be equal to or greater than the maximum user information size.	256 KB (default value)

Configuring Adabas security

Configure security to access Adabas data at a DBID or file number level.

About this task

Securing Adabas files at a DBID or file number level requires the use of the following Data Service server parameters:

- RESOURCETYPE
- SQLVTRESOURCETYPE
- ADABASSECURITY

The following sample jobs for defining Adabas security-related definitions are provided in the *hlq.SAZKCNTL* library:

- AZKRAADA for RACF

Note: When using job AZKRAADA, make the following changes for file ID security:

```
RDEFINE FACILITY ADAxxxxx.FILyyyyy UACC(NONE)
PERMIT ADAxxxxx.FILyyyyy CLASS(FACILITY) ID(<USERID>)
ACCESS(aaaa)
SETROPTS REFRESH RACLIST(FACILITY)
```

- Change xxxxx to the Adabas database ID.
- Change yyyyyy to the Adabas file ID.
- AZKA2ADA for CA ACF2
- AZKTSADA for CA Top Secret

Procedure

1. Locate the server configuration member. The server initialization member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and may have been copied to a new data set for customization in the step "Copying target libraries" in the *Installation and Customization Guide*.
2. Ensure the following settings are set in the AZKIN00 file:

```
MODIFY PARM NAME(RESOURCE TYPE) VALUE(RAVZ)
MODIFY PARM NAME(SQLVTRESOURCE TYPE) VALUE(RAVZ)
MODIFY PARM NAME(ADABAS SECURITY) VALUE(YES)
```

Parameter name	Parameter description	Value
RESOURCE TYPE	RESOURCE TYPE FOR RESOURCE RULES Specify the name of the security server's class (or resource type for ACF2) that is used to perform resource access authorization checks. When using RACF, the corresponding class name within RACF must start with R, for example, RAZK.	For RACF: RAZK
SQLVTRESOURCE TYPE	RESOURCE TYPE FOR SQL ACCESS TO VIRTUAL TABLES Specify the name of the security server's class (or resource type for ACF2) that is used to perform authorization checks for SQL access to metadata and virtual tables in the SQL Engine. When using RACF, the corresponding class name within RACF must start with R, for example, RAZK.	For RACF: RAZK
ADABAS SECURITY	ADABAS SECURITY ACTIVATED Set this parameter to indicate that a resource rule is to be constructed consisting of DBID and file. Note: Both RESOURCE TYPE and SQLVTRESOURCE TYPE must be set in order for ADABAS SECURITY to be in effect.	YES

Configuring access to data in relational database management systems (RDBMS)

You can access data on DB2 for z/OS and distributed databases IBM Big SQL, IBM dashDB, DB2 LUW (Linux, UNIX, and Windows), Microsoft SQL Server, Oracle, and QMF DRDA.

Before you begin

The server and relational database management system (RDBMS) must already be installed.

About this task

To configure and verify access to data in a RDBMS, perform the following tasks.

Procedure

1. Enable the RDBMS access method in the Data Service configuration member.
See [“Modifying the server configuration member for DRDA” on page 7](#).
2. Configure access to the database.

- IBM DB2 for z/OS

Configure DB2 to use the Distributed Relational Database Architecture (DRDA) access method.

See [“Configuring access to IBM DB2 for z/OS” on page 6](#).

- Distributed databases, including Big SQL, dashDB, DB2 LUW, Microsoft SQL Server, Oracle, and QMF DRDA.

Configure the RDBMS to use the Distributed Relational Database Architecture (DRDA) access method.

See [“Configuring access to distributed databases” on page 20](#).

Configuring access to IBM DB2 for z/OS

Configure DB2 for z/OS by using the server started task JCL, the Distributed Relational Database Architecture (DRDA) access method, and security.

About this task

Configure access to DB2 for z/OS databases as follows.

Procedure

1. [“Configuring security” on page 6](#)
2. Configure for DRDA (Distributed Relational Database Architecture) or for RRSF (Resource Recovery Services attachment facility) access method.
 - If you are using a zIIP specialty engine, enable the RDBMS access method for DRDA:
 - a. [“Modifying the server configuration member for DRDA” on page 7](#)
 - b. [“Configuring DB2 for DRDA” on page 12](#)
 - If you are not using a zIIP specialty engine, enable the RDBMS access method for RRSF:
 - a. [“Modifying the server configuration member for RRSF” on page 12](#)
 - b. [“Configuring DB2 for RRSF” on page 14](#)

Configuring security

Configure security to provide user access to DB2.

About this task

If the DB2 being accessed does not have the DSNZPARM DDF option TCPALVER set to either YES or CLIENT, then a passticket is needed for certain DB2 on z/OS DRDA operations. These operations may include:

- Refreshing in-memory metadata catalog information at server startup for DB2 on z/OS defined virtual tables. Catalog information is refreshed at every server startup by the server connecting to each DB2 where virtual tables have been defined.
- Any SQL statement coming from the dsClient interface, dsSpufi or application APIs using the dsClient interface. This may also include running administrative tasks in batch using dsClient that accesses DB2 on z/OS such as updating MapReduce information using the DRDARange command.

Procedure

1. This step only applies to DB2 for z/OS. To grant users access to the DB2 subsystem and to enable passticket logon processing, you must define one RACF PTKTDATA resource for each unique DRDA APPLNAME. To define each PTKTDATA resource, customize and run the appropriate job.
 - AZKRADB2 is for IBM Resource Access Control Facility (RACF) security.
 - AZKA2DB2 is for CA ACF2 (Access Control Facility) security.

- AZKTSDDB2 is for CA Top Secret Security (TSS).
2. Assign users READ authority.
 - For DRDA, assign users READ authority to the *ssid*.DIST profile.

Configuring the server started task JCL

If you use Db2 z/OS, add the Db2 load library to the server started task JCL.

Before you begin

All LOAD library data sets allocated to the Data Service server in the server started task JCL must be APF-authorized.

Procedure

Edit the JCL in the *hlq*.SAZKCNTL (AZK1PROC) member to include in the PROC statement the DB2LIB parameter with the Db2 library name assigned, as shown in the following example:

```
DB2LIB= 'DSNX10'
```

The Db2 library must contain the Db2 interface modules, such as DSNALI and DSNHLI, and must be in uppercase and enclosed in quotation marks.

Modifying the server configuration member for DRDA

If you are using a zIIP specialty engine, enable the RDBMS access method for Distributed Relational Database Architecture (DRDA) in the server configuration member.

About this task

Configure the server to use Distributed Relational Database Architecture (DRDA) when accessing a RDBMS.

Modify the server configuration member in data set *hlq*.AZKS.SAZKEXEC(AZKSIN00). The server configuration member is shipped in data set member *hlq*.SAZKEXEC(AZKSIN00) and copied to *hlq*.AZKS.SAZKEXEC(AZKSIN00) by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. Verify that the Unicode translation of the Coded Character Set Identifier (CCSID) used in the DEFINE DATABASE statement and the CCSID used by the target RDBMS are defined for your z/OS environment.

- a) You should identify the CCSID of the RDBMS.

For example, Oracle may use *ccsid1*. In your DEFINE DATABASE statement in the configuration member for the RDBMS you have *ccsid2*. For this example, where Oracle is using *ccsid1*, you need to verify that you have *ccsid1-ccsid2* and *ccsid2-ccsid1* defined in your Unicode translation table on z/OS using the command `D UNI,ALL`.

- b) If the entry is not present, you need to add the entry to your Unicode translation table and refresh.

Please refer to the IBM z/OS documentation on how to add the entry.

Note: As an alternative, the Unicode table can be appended within the server by using the following statement examples in the server configuration member:

```
"DEFINE CONV SOURCE(ccsid1) TARGET(ccsid2) TECH(RE)"
"DEFINE CONV SOURCE(ccsid2) TARGET(ccsid1) TECH(RE)"
```

2. In the AZKSIN00 member, locate the section that contains the comment `Enable DRDA access to DB2 database subsystems`.

3. Enable the DRDA parameters by changing the syntax `if DontDoThis` to `if DoThis`, and then set the `DRDASKIPZSERVICES` parameter to YES. The following example shows the section in the configuration member to enable:

```

/*-----*/
/* Enable DRDA access to DB2 database subsystems */
/*-----*/
if DoThis then
do
    "MODIFY PARM NAME(TRACEOEDRDARW) VALUE(YES)"
    "MODIFY PARM NAME(CLIENTMUSTELECTDRDA) VALUE(NO)"
    "MODIFY PARM NAME(DRDASKIPWLMSETUP) VALUE(NO)"
    "MODIFY PARM NAME(DRDAFORLOGGINGTASK) VALUE(NO)"
    "MODIFY PARM NAME(DRDASKIPZSERVICES) VALUE(YES)"

```

The following table describes these parameters:

Parameter	Description	Valid values
TRACEOEDRDARW	If set to YES (recommended), TCP/IP communications via DRDA are traced. If set to NO, DRDA receive and send operations are not traced.	YES NO Default value.
CLIENTMUSTELECTDRDA	If set to YES, JDBC clients must explicitly opt in for DRDA to be used by setting the user parameter connection variable to 'DRDA'. Note: JDBC clients can always opt out of DRDA processing by setting the user parameter to 'NODRDA'. If set to NO, DRDA processing is used for access all configured RDBMSs.	YES NO Default value.
DRDASKIPWLMSETUP	If set to YES, WLM information is not collected and sent to DRDA during JDBC logon processing. If captured, the DRDA equivalent to SET_CLIENT_ID calls is issued after logon to establish these values on the DRDA connection. If not captured, the transmission that is used to set these WLM-related values is bypassed. If set to NO, the client user ID, application name, workstation name, and accounting token that were sent in the initial client buffer are collected and sent separately after logon processing to DRDA.	YES NO Default value.

Parameter	Description	Valid values
DRDAFORLOGGINGTASK	If set to YES, DRDA processing is used for the Db2 on z/OS logging subtask. If set to NO, SAF or RRSAP connections are used. Note: Passticket support must be enabled for the target DDF server. If passticket support is not configured, set the parameter to NO.	YES NO Default value.
DRDASKIPZSERVICES	Prevents DRDA from being used for z/Service Db2 processing. If set to YES, z/Services client tasks do not use DRDA processing for Db2 requests. If set to NO, DRDA will be used when configured for a particular Db2 connection. Note: Passticket support must be enabled for all target DDF servers.	YES NO Default value.

4. Define DRDA RDBMSs by entering a definition statement. Provide your local environment values for all the parameters. The following example shows the section in the configuration member to enable:

```
"DEFINE DATABASE TYPE(type_selection)"
      "NAME(name)"
      "LOCATION(location)"
      "DDFSTATUS(ENABLE)"
      "DOMAIN(your.domain.name)"
      "PORT(port)"
      "IPADDR(1.1.1.1)"
      "CCSID(37)"
      "APPLNAME(DSN1LU)"
      "IDLETIME(110)"
```

Where *type_selection* is either GROUP, MEMBER, or ZOSDRDA.

The previous example shows only a subset of the available parameters. The following table lists all available parameters for defining DDF endpoints:

Parameter	Description	Valid values
APPLNAME	Application name. The APPLNAME used by the target endpoint for passticket generations. <i>(Optional)</i>	A valid value is 1 - 8 characters. If APPLNAME is not specified in the definition statement, no default value is provided and passticket access is disabled. Note: APPLNAME is not required when connecting from the JDBC driver.

Parameter	Description	Valid values
AUTHTYPE	Authentication type. This can be either DES (Diffie Hellman Encryption Standard) or AES (Advanced Encryption Standard). When AUTHTYPE is not supplied, the default is DES. To force AES, the option must be added to the DEFINE DATABASE statement. Each server can be different in what is supported as to AES/DES. For this setting to have effect, you must specify a security mechanism (SECMEC) that requests encryption.	DES Diffie Hellman Encryption Standard (default value) AES Advanced Encryption Standard.
CCSID	Specify the EBCDIC single-byte application CCSID (Coded Character Set Identifier) configured for this RDBMS subsystem on the RDBMS installation panel DSNTIPF, option 7. <i>(Optional)</i>	Refer to the RDBMS vendor documentation for a list of valid CCSID.
DDFSTATUS	The DDF activation status can be altered online by using the ISPF 4-DB2 dialog panels. <i>(Required)</i>	ENABLE To make this DDF definition active within server. DISABLE DDF endpoint is not used.
DOMAIN	The part of a network address that identifies it as belonging to a particular domain.	No default value.
IDLETIME	If Db2 ZPARM parameter IDTHTOIN is set to a non-zero value set IDLETIME to a value slightly less (10 secs.) than IDTHTOIN. This will also allow product DRDA threads to become inactive. <i>(Db2 for z/OS only)</i>	0-9999 seconds.
IPADDR	Specify the dot-notation IPV4 address of the DDF endpoint. <i>(Optional)</i>	If this parameter is not specified, the value 127.0.0.1 (local host) is the default. For group director definitions, use the DVIPA IP address of the group director.

Parameter	Description	Valid values
LOCATION	For Db2: The Db2 location name. For LUW: The LUW database. For Oracle: The Oracle SSID as defined to the Oracle Database Provider (Gateway) <i>(Required)</i>	A valid value is a string 1 - 16 characters.
NAME	The database name as known to the server. <i>(Required)</i>	A valid value consists of 1 - 4 characters. Clients use this ID when they request access to a specific Db2 subsystem.
PORT	The TCP/IP port at which the server is listening. <i>(Required)</i>	If this keyword is not entered, the default DRDA port number 443 is used.
SECMEC	The DRDA security mechanism in force. <i>(For GROUP and MEMBER types.)</i>	USERIDPWD User ID and password are sent as is. No encryption is used. USRIDONL User ID is sent as is. No encryption is used for the user ID only (client security). USRENCPWD Encrypt the password only. EUSRIDPWD Encrypt the user ID and password.
SYSTEMVCAT	The VCATNAME for the Db2 system catalog tables (in the DSNDB06 database). The VCATNAME for system catalog tables is a system bootstrap value and not available using the data discovery query. Use this parameter if you intend to access the system catalog tables using Db2 Direct or if the VCATNAME for database DSNDB06 is different from the subsystem name.	A valid value is 1 - 8 characters. If this parameter is not specified, the 4-character Db2 subsystem name is used by default as the high-level qualifier for Db2 data sets.

Parameter	Description	Valid values
TYPE	For Db2 for z/OS: GROUP DDF endpoint is a Db2 group director. MEMBER DDF endpoint is a Db2 instance or group member for z/OS. ZOSDRDA DDF endpoint is a remote z/OS Db2 on another LPAR. This setting allows you to use SEF ATH rules when z/OS Pass Ticket passwords cannot be used or the server administrator has the requirement to manage the authentication credentials for remote z/OS systems.	For Db2 for z/OS: GROUP MEMBER ZOSDRDA

Configuring DB2 for DRDA

If you are using a zIIP specialty engine, configure DB2 to use DRDA.

About this task

Before you can successfully issue DRDA requests, you must bind IBM Open Data Analytics for z/OS DBRMs into packages within each target DB2 subsystem.

Procedure

1. Set the DEFAULTDB2SUBSYS parameter in the server configuration member to a valid DB2 subsystem name.
2. Edit the AZKBINDD job that is supplied in the *hlq.SAZKCNTL* data set.
Follow the instructions that are provided in the JCL.
3. Run the AZKBINDD job.

Modifying the server configuration member for RRSF

If you are not using a zIIP specialty engine, enable the RDBMS access method for Resource Recovery Services attachment facility (RRSAF) in the server configuration member.

About this task

This task is only applicable for DB2 for z/OS.

Modify the server configuration member in data set *hlq.AZKS.SAZKEXEC(AZKSIN00)*. The server configuration member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. Verify that the Unicode translation of the Coded Character Set Identifier (CCSID) used in the DEFINE DATABASE statement and the CCSID used by the target RDBMS are defined for your z/OS environment.
 - a) You should identify the CCSID of the RDBMS.

For example, Oracle may use *ccsid1*. In your DEFINE DATABASE statement in the configuration member for the RDBMS you have *ccsid2*. For this example, where Oracle is using *ccsid1*, you need to verify that you have *ccsid1-ccsid2* and *ccsid2-ccsid1* defined in your Unicode translation table on z/OS using the command D UNI,ALL.

- b) If the entry is not present, you need to add the entry to your Unicode translation table and refresh.

Please refer to the IBM z/OS documentation on how to add the entry.

Note: As an alternative, the Unicode table can be appended within the server by using the following statement examples in the server configuration member:

```
"DEFINE CONV SOURCE(ccsid1) TARGET(ccsid2) TECH(RE)"
"DEFINE CONV SOURCE(ccsid2) TARGET(ccsid1) TECH(RE)"
```

- Set the DEFAULTDB2SUBSYS parameter in the server configuration member AZKSIN00 to a valid DB2 subsystem name.
- In the AZKSIN00 member, locate the comment "ENABLE DB2 RRSF SUPPORT" section.
- Enable the RRSF parameters by changing the syntax if DontDoThis to if DoThis. The following example shows the section in the configuration member to enable:

```
if DoThis then
do
  "MODIFY PARM NAME(RRS) VALUE(YES)"
  "MODIFY PARM NAME(DB2ATTACHFACILIT) VALUE(RRS)"
  "MODIFY PARM NAME(TRACERSSDATA) VALUE(NO)"
  "MODIFY PARM NAME(TRACERSSEVENTS) VALUE(YES)"
  "MODIFY PARM NAME(TRACERSSAF) VALUE(YES)"
end
```

The following table lists the parameters for configuring support for RRSF:

Parameter	Description	Valid values
DB2ATTACHFACILITY	Specifies the DB2 attach facility. The Resource Recovery Services attachment facility (RRSAF) uses the DSNRLI interface module and allows for 2-phase commit actions. The Call Attach Facility (CAF) uses the DSNALI interface module.	The default value is RRS. Valid values are RRS and CAF.
RRS	Activates RRS support. This parameter must be set to YES to activate RRS.	YES Default value. NO
TRACERSSDATA	Specifies whether to trace RRS data.	YES Default value. NO
TRACERSSEVENTS	Specifies whether to trace RRS events.	YES Default value. NO
TRACERSSAF	Creates an entry in the server trace for each call to DSNRLI for RRSF requests.	YES Default value. NO

Configuring DB2 for RRSAP

If you are not using a zIIP specialty engine, configure RRSAP for access to local DB2.

About this task

This task only applies to DB2 for z/OS.

Procedure

1. Run the AZKBINDC member of the *hlq*.SAZKCNTL data set to bind the following server product plans:
 - AZKC1010 is bound using cursor stability.
 - AZKR1010 is bound using repeatable read.
 - AZKS1010 is bound using read stability.
 - AZKU1010 is bound using uncommitted read.

Use AZKC1010 as the default server plan, and use the other product plans for operations that require those levels of isolation. To change the default plans, edit the BIND member and replace the default plan names with new names. You must run the BIND job of the *hlq*.SAZKCNTL data set against each DB2 subsystem that you want to access. Use the instructions in the JCL to customize the job.

2. Install the DSN3@SGN exit in the DB2 master task (normally placed in the SDSNEXIT data set). Installing this exit enables the server to use DB2 authority that was granted through secondary DB2 authorization IDs.

Setting CURRENT QUERY ACCELERATION

You can use a Server Event Facility (SEF) rule to set the value used by the server for the SET CURRENT QUERY ACCELERATION command.

About this task

The Data Service server sends the command SET CURRENT QUERY ACCELERATION to a DRDA server when the data source is Db2 for z/OS. By default, the command uses the setting ENABLE WITH FAILBACK, which allows access to accelerator tables but does not prevent access to non-accelerator tables. Using a virtual table rule, you can use the OPTBCUQA field to specify any CURRENT QUERY ACCELERATION valid value as the setting that the server will use when sending the command. The following table lists the CURRENT QUERY ACCELERATION values and the corresponding OPTBCUQA settings:

OPTBCUQA value	CURRENT QUERY ACCELERATION value
0	ENABLE WITH FAILBACK
1	NONE When using this setting, if the table is an accelerator only table, the query will fail.
2	ENABLE
3	ELIGIBLE
4	ALL When using this setting, if the table is not an accelerator table, the query will fail.

The OPTBCUQA setting has effect only when the table is owned by a Db2 for z/OS subsystem; otherwise, there is no impact to the processing.

Procedure

To set the CURRENT QUERY ACCELERATION value that the server will use when sending the SET CURRENT QUERY ACCELERATION command, in a VTB rule, set the variable VTB.OPTBCUQA to a valid CURRENT QUERY ACCELERATION value. For more information, see the generic sample rule AZKMDTBL.

Note: If both OPTBCUQA and OPTBNOQA are specified in a rule, OPTBCUQA takes precedence. OPTBNOQA will eventually be deprecated.

Configuring access to DB2 unload data sets

To be able to access a DB2 unload data set directly with an SQL query, you must configure a virtual table rule to define the DB2 unload data set name to the DB2 virtual table.

About this task

To configure access to a DB2 unload data set, you must add the DB2 unload data set name to the DB2 virtual table in a Server Event Facility (SEF) virtual table rule. With this access, you can issue SQL queries directly against DB2 unload data sets using existing DB2 virtual tables.

Switching a DB2 virtual table to read an unload data set is done by assigning a data set name to the table in a virtual table rule. The VTB variable vtb.optbdsna is used to redirect access from DB2 to reading the sequential file named in the variable. The named sequential file must contain unload data created by the DB2 UNLOAD utility. A model VTB rule, AZKMDLDU, is provided to demonstrate redirecting a DB2 virtual table to a DB2 unload data set.

As an example, consider a virtual table named DSNA_EMPLOYEES that maps the EMPLOYEES table in DB2 subsystem DSNA. By activating the model rule AZKMDLDU, you can query an unload sequential dataset named EMPLOYEE.UNLOAD.SEQ by issuing the following query:

```
SELECT * FROM MDLDU_DSNA_EMPLOYEES__EMPLOYEE_UNLOAD_SEQ
```

The AZKMDLDU rule performs the following steps:

1. Extracts the table name DSNA_EMPLOYEES and sets the VTB variable vtb.optbmtna.
2. Extracts the data set name EMPLOYEE_UNLOAD_SEQ, converts the underscores to periods, and sets the VTB variable vtb.optbdsna.

The following restrictions and considerations apply when using this feature:

- SQL access to DB2 unload files is limited to SQL queries only.
- The columns in DB2 virtual table definition must exactly match the table unloaded in DB2.

Use the following procedure to configure the sample rule AZKMDLDU.

Note: Sample rule AZKMDLDU is intended to be used as a model and may require customization. When customizing this rule, additional logic may need to be added if different unload data sets require different VTB variable settings for CCSID or internal/external format.

Procedure

1. Customize the server configuration member (AZKSIN00) to enable virtual table rule events by configuring the SEFVTBEVENTS parameter in the member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES) "
```

2. Access the VTB rules, as follows:
 - a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
 - b) Specify option 2, **SEF Rule Management.**
 - c) Enter VTB for **Display Only the Ruleset Named.**
3. Customize the AZKMDLDU rule, as follows:
 - a) Specify S next to AZKMDLDU to edit the rule.

- b) Find the vtb.optbdsna variable and specify the name of the DB2 unload data set to process.
- c) Update additional rule options as needed. The following table describes the VTB rule options that support DB2 unload data set access.

VTB variable	Description
vtb.optbdlcv	If the data was unloaded with a DELIMITED statement, set vtb.optbdlcv to 1 to declare the data is in delimited format. It may also be necessary to declare the delimiters if the default column delimiter (,) and character string delimiter (") were overridden when the data was unloaded.
vtb.optbdsna	Specifies the name of the sequential unload data set created by the DB2 UNLOAD utility to access.
vtb.optbduif	By default, the DB2 unload utility writes data in external format. If FORMAT INTERNAL is used when unloading data, vtb.optbduif must be set to 1 to declare that the data was unloaded in internal format.
vtb.optbmtna	Specifies the map name of the DB2 virtual table describing the unload file.
vtb.optbtbcc	If the table CCSID is not compatible with the CCSID defined for the SQL engine (AZKSIN00 SQLENGDFLTCCSID parameter), vtb.optbtbcc can be used to declare the CCSID of the data. This is particularly important for Unicode tables and tables containing GRAPHIC columns.

- d) Save your changes and exit the editor.
 4. Enable the rule by specifying E next to AZKMDLDU and pressing Enter.
 5. Set the rule to Auto-enable by specifying A next to AZKMDLDU and pressing Enter.
- Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

Db2 for z/OS data access methods

Db2 for z/OS data can be accessed by the Data Service server using different data access methods.

The following Db2 for z/OS data access methods are available:

- Traditional Db2 access. This method accesses Db2 data through traditional Db2 APIs. This access method allows for reading and writing of the data and provides transactional integrity.
- Db2 Direct. This method accesses Db2 data by reading the underlying Db2 VSAM linear data sets directly. This access method allows read-only access to the data and provides high performance, bulk data access.

The Db2 data access method is specified when creating virtual tables in the Data Service Studio for access to Db2 data.

The following topics provide more information about the Db2 for z/OS data access methods.

Using traditional Db2 access

Traditional Db2 access methods access Db2 data through APIs such as Distributed Relational Database Architecture (DRDA), Call Attachment Facility (CAF), and Resource Recovery Services attachment facility (RRSAF). Using traditional Db2 access allows for reading and writing of the data and provides transactional integrity.

Traditional DB2 access methods provide MapReduce and Virtual Parallel Data support. MapReduce is an algorithm that enables the Data Service server to streamline how it accesses Db2 data, thereby reducing the processing time required to virtualize Db2 data. Statistics about the Db2 database are collected and stored within a metadata repository from which the SQL engine optimizes the MapReduce process.

In order to exploit MapReduce for Db2 when using traditional Db2 access, the Data Service server must collect information about the Db2 database. This information is collected using the DRDARange command and is stored within the Data Service server metadata repository.

Traditional Db2 access is used automatically when Db2 Direct access is not available.

Using Db2 Direct

Db2 Direct is a Data Service server access method that reads the data in the Db2 VSAM linear data sets directly instead of accessing the data through traditional Db2 APIs. Using Db2 Direct, large data pulls can be performed in service request block (SRB) mode, and MapReduce and Virtual Parallel Data features can be exploited without any prerequisite processing, such as the collection of statistics using the DRDARange command. Db2 Direct access provides a significant increase in performance and reduced elapsed time in processing analytical type queries.

Db2 Direct allows read-only access to the data. When using Db2 Direct, there is no locking involved when accessing the data, so updates may not be captured and deleted records may have been captured. Results from Db2 Direct queries may be out of sync with the current state of a Db2 table due to recent table updates not being flushed to the linear data sets.

Security is managed using Db2 table authorization.

Restrictions and considerations:

Consider the following points when using Db2 Direct:

- The Db2 subsystem hosting a Db2 table must be active when Db2 Direct-enabled tables are loaded or refreshed in the data server. The map build process requires Db2 system access to identify data set information in the Db2 system catalog.
- The Data Service server requires read access to the Db2 VSAM linear data sets. The linear data sets containing the Db2 rows must be available to the data server processing SQL requests for Db2 data. If the data sets are unavailable or archived, Db2 Direct will be disabled during map load or refresh for the virtual table.
- Virtual tables enabled for Db2 Direct must include all the columns defined in the base Db2 table. This is necessary because the columns describe the internal format of the Db2 data.
- If Db2 is not available or some other error occurs during map build or map refresh processing, Db2 Direct is automatically disabled for the table and a message is written to the trace log:

```
DB2 direct processing disabled for map map-name
```

- If Db2 Direct processing is disabled, processing will continue with traditional Db2 APIs when possible.
- To determine if Db2 Direct is active, the following messages appear in the server trace:
 - At startup and map refresh, the following message is issued:

```
DB2 direct processing enabled for map map-name
```

- When DB2 Direct is used in a query, the following message is issued:

```
Processing table map-name using DB2 direct
```

- If Db2 Direct table security is enabled, the Db2 subsystem must be available to check security at SQL query time.
- If Db2 Direct table security is disabled, unauthorized users who would normally receive a -551 SQLCODE attempting to access data through traditional APIs may gain access to Db2 data.
- Db2 Direct does not support tables with edit procedures or SQL statements containing joins, LOB columns, or key columns.
- If Db2 Direct security is disabled, the CCSIDs of table columns will be assumed based on the ENCODING_SCHEME (EBCDIC, Unicode, ASCII) of the table.

Configuring Db2 Direct

Configure Db2 Direct options or disable Db2 Direct.

Before you begin

Review the restrictions and considerations when using Db2 Direct. See [“Using Db2 Direct” on page 17](#).

About this task

By default, Db2 Direct is enabled in the Data Service server. Use the information in this topic to perform the following optional tasks:

- Disable the Db2 Direct feature for a virtual table by using a Virtual Table (VTB) rule.
- Define the VCATNAME for the DB2 system catalog tables (in the DSNDB06 database) by modifying the `DEFINE DATABASE` statement. The VCATNAME for system catalog tables is a system bootstrap value and is not available using the data discovery query. This task is required only in the following situations:
 - Access to system catalog tables using Db2 Direct is intended.
 - The VCATNAME for database DSNDB06 is different from the subsystem name.
- Configure Db2 Direct options, such as the number of pages to allocate for Db2 segment information, whether to enforce Db2 SQL table security authorizations, and disabling Db2 Direct for the server, by modifying server parameters.
- Specify what Db2 Direct information to display in the server trace by modifying server parameters.

Procedure

1. To disable the Db2 Direct feature for a virtual table, in a VTB rule, set the variable `OPTBDIDD` to 1. For additional information, see the generic sample rule `AZKMDTBL`.
2. To define the VCATNAME for the Db2 system catalog tables, perform the following steps:
 - a) Locate the server configuration member. The server initialization member is shipped in data set member `hlq.SAZKEXEC(AZKSIN00)` and may have been copied to a new data set for customization.
 - b) In the `DEFINE DATABASE` statement, use the `SYSTEMVCAT` parameter to define the VCATNAME for the system catalog tables, as shown in the following example:

```
"DEFINE DATABASE TYPE(MEMBER)"
      "NAME(DBA9)"
      "LOCATION(RS28DDS9)"
      "DDFSTATUS(ENABLE)"
      "PORT(3725)"
      "IPADDR(127.0.0.1)"
      "CCSID(37)"
      "APPLNAME(DBA9DB2)"
      "SYSTEMVCAT(DDS9)"
      "IDLETIME(110)"
```

3. To modify server parameters, perform the following steps:
 - a) Locate the server configuration member. The server initialization member is shipped in data set member `hlq.SAZKEXEC(AZKSIN00)` and may have been copied to a new data set for customization.
 - b) Use the `MODIFY PARM` command to change a parameter value. For example, the following command disables Db2 Direct for the Data Service server:

```
"MODIFY PARM NAME(DISABLEDB2DIRECT) VALUE(YES)"
```

The parameters in the following tables are available for use with Db2 Direct.

Table 2: SQL parameters in group PRODSQL		
Parameter name	Parameter description	Default value
DB2DIRECTSEGTLBPAGES	DB2-DIRECT SEGMENT TABLE PAGES Defines the number of 4K pages to be allocated for Db2 segment information. The default value is 8, which should be enough for most Db2 Direct queries. This parameter should only be changed if a query fails because the Db2 Direct segment table was exhausted.	8
DISABLEDB2DIRECT	DISABLE DB2-DIRECT PROCESSING Disables Db2 Direct processing in the server.	NO
DISABLEDB2DIRSEC	DISABLE DB2-DIRECT TABLE SECURITY Disables SQL table security checking when Db2 Direct is selected to process Db2 data. Disabling table security checking will allow access to Db2 data when the target Db2 subsystem is not active. Important: Unauthorized users who would normally receive a -551 SQLCODE attempting to access data through traditional APIs like DRDA may gain access to Db2 data.	NO

Table 3: SQL parameters in group PRODTRACE		
Parameter name	Parameter description	Default value
TRACEDB2DIRSTATS	TRACE DB2-DIRECT STATISTICS Enables tracing of a summary report to the system trace after each Db2 Direct query. Included in the trace are statistics about read and point operation in the Db2 linear data set(s) processed.	NO
TRACEDB2DIOPEN	TRACE DB2-DIRECT OPEN CONTROL BLOCKS Enables tracing of control blocks created at the open of each linear data set for Db2 Direct processing.	NO

Table 3: SQL parameters in group PRODTRACE (continued)		
Parameter name	Parameter description	Default value
TRACEDB2DIRSEGP	TRACE DB2-DIRECT SEGMENT PAGES Enables tracing if Db2 pages containing segmented map information.	NO
TRACEDB2DIRDICTP	TRACE DB2-DIRECT DICTIONARY PAGES Enables tracing of the compression dictionary used to compress and expand rows stored in Db2 linear data sets.	NO
TRACEDB2DIRDATAP	TRACE DB2-DIRECT DATA PAGES Enables tracing of data pages in a linear data set containing Db2 rows.	NO
TRACEDB2DIRROWS	TRACE DB2-DIRECT ROWS Enables tracing of rows extracted from data pages in a Db2 linear data set. If rows are compressed, an additional trace is created of the uncompressed row data.	NO

Configuring access to distributed databases

You can configure access to data on Big SQL, dashDB, DB2 LUW (Linux, UNIX, and Windows), Microsoft SQL Server, Oracle, and QMF DRDA.

Before you begin

If you are connecting to a Big SQL or DB2 LUW database, then you must install and configure the IBM DB2 Federated Server. For additional information, refer to the documentation on the IBM website.

If you are connecting to an Oracle database, then you must install and configure the Oracle Database Provider for DRDA. For additional information, refer to the documentation on the Oracle website.

If you are connecting to a 2016 Microsoft SQL Server database, then you must install and configure the Host Integration Server for HIS DRDA Service. For additional information, refer to the documentation on the Microsoft website. The SYSIBM Views from Microsoft must be installed.

About this task

Configure access to distributed databases by modifying the configuration member, configuring Server Event Facility (SEF) rules, and optionally setting up alternate authentication information.

Procedure

1. [“Modifying the server configuration member”](#) on page 21.
2. Configure the Server Event Facility rules and set up authentication for the appropriate database.
 - [“Configuring rules and authentication for Big SQL”](#) on page 26.
 - [“Configuring rules and authentication for dashDB”](#) on page 27.
 - [“Configuring rules and authentication for LUW databases”](#) on page 28.

- [“Configuring rules and authentication for Microsoft SQL Server” on page 29.](#)
- [“Configuring rules and authentication for Oracle DRDA” on page 30.](#)
- [“Configuring rules and authentication for QMF DRDA Server” on page 31.](#)

Modifying the server configuration member

Enable the RDBMS access method in the Data Service server configuration member.

About this task

Configure the server to use Distributed Relational Database Architecture (DRDA) when accessing a RDBMS.

Modify the server configuration member in data set *hlq*.AZKS.SAZKEEXEC(AZKSIN00). The server configuration member is shipped in data set member *hlq*.SAZKEEXEC(AZKSIN00) and copied to *hlq*.AZKS.SAZKEEXEC(AZKSIN00) by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. Verify that the Unicode translation of the Coded Character Set Identifier (CCSID) used in the DEFINE DATABASE statement and the CCSID used by the target RDBMS are defined for your z/OS environment.
 - a) Identify the CCSID of the RDBMS.

For example, Oracle may use *ccsid1*. In your DEFINE DATABASE statement in the configuration member for the RDBMS you have *ccsid2*. For this example, where Oracle is using *ccsid1*, you need to verify that you have *ccsid1-ccsid2* and *ccsid2-ccsid1* defined in your Unicode translation table on z/OS using the command D UNI,ALL.

- b) If the entry is not present, add the entry to your Unicode translation table and refresh.

Refer to the IBM z/OS documentation on how to add the entry.

Note: As an alternative, the Unicode table can be appended within the server by using the following statement examples in the server configuration member:

```
"DEFINE CONV SOURCE(ccsid1) TARGET(ccsid2) TECH(RE)"
"DEFINE CONV SOURCE(ccsid2) TARGET(ccsid1) TECH(RE)"
```

2. In the AZKSIN00 member, locate the section that contains the comment “Enable DRDA access to DB2 database subsystems.”
3. Enable the DRDA parameters by changing the syntax `if DontDoThis` to `if DoThis` and then set the DRDASKIPZSERVICES parameter to YES. The following example shows the section in the configuration member to enable:

```
if DoThis then
do
  "MODIFY PARM NAME(TRACEOEDRDARW)      VALUE(YES)"
  "MODIFY PARM NAME(CLIENTMUSTELECTDRDA) VALUE(NO)"
  "MODIFY PARM NAME(DRDASKIPWLMSETUP)    VALUE(NO)"
  "MODIFY PARM NAME(DRDAFORLOGGINGTASK)  VALUE(NO)"
  "MODIFY PARM NAME(DRDASKIPZSERVICES)   VALUE(YES)"
```

The following table lists the parameters for configuring support for DRDA:

Parameter	Description	Valid values
CLIENTMUSTELECTDRDA	If set to YES, JDBC clients must explicitly opt in for DRDA to be used by setting the user parameter connection variable to 'DRDA'. Note: JDBC clients can always opt out of DRDA processing by setting the user parameter to 'NODRDA'. If set to NO, DRDA processing is used for access to all configured RDBMSs.	YES NO Default value.
DRDAFORLOGGINGTASK	If set to YES, DRDA processing is used for the DB2 on z/OS logging subtask. If set to NO, SAF or RRSF connections are used. Note: Passticket support must be enabled for the target DDF server. If passticket support is not configured, set the parameter to NO.	YES NO Default value.
DRDASKIPWLMSETUP	If set to YES, WLM information is not collected and sent to DRDA during JDBC logon processing. If captured, the DRDA equivalent to SET_CLIENT_ID calls is issued after logon to establish these values on the DRDA connection. If not captured, the transmission that is used to set these WLM-related values is bypassed. If set to NO, the client user ID, application name, workstation name, and accounting token that were sent in the initial client buffer are collected and sent separately after logon processing to DRDA.	YES NO Default value.

Parameter	Description	Valid values
DRDASKIPZSERVICES	Prevents DRDA from being used for z/Service DB2 processing. If set to YES, z/Services client tasks do not use DRDA processing for DB2 requests. If set to NO, DRDA will be used when configured for a particular DB2 connection. Note: Passticket support must be enabled for all target DDF servers.	YES NO Default value.
TRACEOEDRDARW	If set to YES (recommended), TCP/IP communications via DRDA are traced. If set to NO, DRDA receive and send operations are not traced.	YES NO Default value.

4. Define DRDA RDBMSs by entering a definition statement. Provide your local environment values for all the parameters. The following example shows the section in the configuration member to enable:

```
"DEFINE DATABASE TYPE(type_selection) "
      "NAME(name) "
      "LOCATION(location) "
      "DDFSTATUS(ENABLE) "
      "DOMAIN(your.domain.name) "
      "PORT(port) "
      "IPADDR(1.1.1.1) "
      "CCSID(37) "
      "APPLNAME(DSN1LU) "
      "IDLETIME(110) "
```

This is an example for dashDB:

```
"DEFINE DATABASE TYPE(DASHDB) "
      "NAME(name) "
      "LOCATION(location) "
      "AUTHTYPE(AES) "
      "SECMEC(EUSRIDPWD) "
      "DDFSTATUS(ENABLE) "
      "DOMAIN(your.domain.name) "
      "PORT(port) "
      "CCSID(37) "
```

The following table lists the parameters for defining DDF endpoints:

Parameter	Description	Valid values
APPLNAME	Application name. The APPLNAME used by the target endpoint for passticket generations. <i>(Optional)</i>	A valid value is 1 - 8 characters. If APPLNAME is not specified in the definition statement, no default value is provided and passticket access is disabled. Note: APPLNAME is not required when connecting from the JDBC driver.

Parameter	Description	Valid values
AUTHTYPE	Authentication type. This can be either DES for Diffie Hellman Encryption Standard or AES for Advanced Encryption Standard. When AUTHTYPE is not supplied, the default is DES. To force AES, the option must be added to the DEFINE DATABASE statement. Each server can be different in what is supported as to AES/DES.	DES Diffie Hellman Encryption Standard (default value) AES Advanced Encryption Standard.
CCSID	Specify the EBCDIC single-byte application CCSID (Coded Character Set Identifier) configured for this RDBMS subsystem on the RDBMS installation panel DSNTIPF, option 7. (<i>Optional</i>)	Refer to the RDBMS vendor documentation for a list of valid CCSIDs.
DDFSTATUS	The DDF activation status can be altered online by using the ISPF 4-DB2 dialog panels. (<i>Required</i>)	ENABLE Make this DDF definition active. DISABLE DDF endpoint is not used.
DOMAIN	The part of a network address that identifies it as belonging to a particular domain.	No default value.
IDLETIME	If DB2 ZPARM parameter IDTHTOIN is set to a non-zero value set IDLETIME to a value slightly less (10 secs.) than IDTHTOIN. This will also allow product DRDA threads to become inactive. (<i>DB2 for z/OS only</i>)	0-9999 seconds.
IPADDR	Specify the dot-notation IPV4 address of the DDF endpoint. (<i>Optional</i>)	If this parameter is not specified, the value 127.0.0.1 (local host) is the default. For group director definitions, use the DVIPA IP address of the group director.

Parameter	Description	Valid values
LOCATION	For DB2: The DB2 location name. For dashDB: This is the database name of the dashDB database or alias name for the database. For LUW: The LUW database. For Oracle: The Oracle SSID as defined to the Oracle Database Provider (Gateway). <i>(Required)</i>	A valid value is a string 1 - 16 characters.
NAME	The database name as known to the server. <i>(Required)</i>	A valid value consists of 1 - 4 characters. Clients use this ID when they request access to a specific DB2 subsystem.
PORT	The TCP/IP port at which the server is listening. <i>(Required)</i>	A valid 1-5 numeric string.
SECMEC	The DRDA security mechanism in force for standard dashDB services requires an authentication method setting. Define as either USRENCPWD, which informs the server to encrypt the PASSWORD or EUSRIDPWD, which informs the server to encrypt the USERID and PASSWORD during the initial connection to dashDB. <i>(Except QMFDRDA)</i>	USRENCPWD Encrypt password only. EUSRIDPWD Encrypt userid and password.

Parameter	Description	Valid values
TYPE	For distributed databases: BIGSQL DDF endpoint is a Big SQL engine. DASHDB DDF endpoint is a dashDB database. LUW DDF endpoint is a DB2 instance or group member for Linux, UNIX, or Windows. MSSQL DDF endpoint is a DB2 instance or group member for Microsoft SQL Server. ORACLE DDF endpoint is an Oracle instance. The parameter informs DRDA AR and supportive tooling that the remote server is an Oracle Database Provider which supports DRDA AS. The Oracle DRDA AS must be in z/OS simulation mode. QMFDRDA DDF endpoint is a QMF DRDA AS Object Server instance.	For distributed databases: BIGSQL DASHDB LUW MSSQL ORACLE QMFDRDA

Configuring rules and authentication for Big SQL

Configure Server Event Facility (SEF) rules and set up authentication to provide access to Big SQL databases.

About this task

To complete configuration for access to Big SQL databases, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to Big SQL. By default, the server will attempt to log on to Big SQL with the same user ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing Big SQL.

This task uses the following tools:

AZKSBIGC

An SQL rule that allows Meta discovery on Big SQL databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEBIGG

An ATH rule that switches credentials when connecting to a Big SQL database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSBIGC) to allow Data Service Studio Meta discovery on Big SQL databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your Big SQL database, as instructed in the JCL, make sure to specify the correct DBTYPE. For Big SQL, specify DBTYPE=BIGSQL.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEBIGG) to provide the logon credentials to each Big SQL instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Configuring rules and authentication for dashDB

Configure Server Event Facility (SEF) rules and set up authentication to provide access to IBM dashDB databases.

About this task

To complete configuration for access to dashDB databases, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to dashDB. By default, the server will attempt to log on to dashDB with the same user ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing dashDB.

This task uses the following tools:

AZKSDDBC

An SQL rule that allows Meta discovery on dashDB databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEDDBG

An ATH rule that switches credentials when connecting to a dashDB database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSDDBC) to allow Data Service Studio Meta discovery on dashDB databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your dashDB database, as instructed in the JCL, make sure to specify the correct DBTYPE. For dashDB, specify DBTYPE=DASHDB.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEDDBG) to provide the logon credentials to each dashDB instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Configuring rules and authentication for LUW databases

Configure Server Event Facility (SEF) rules and set up authentication to provide access to LUW (Linux, UNIX, and Windows) databases, including databases connected via IBM Federated Server.

About this task

To complete configuration for access to LUW databases, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to LUW databases. By default, the server will attempt to log on to the LUW database with the same user ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing an LUW database.

This task uses the following tools:

AZKSLUWC

An SQL rule that allows Meta discovery on LUW databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKELUWG

An ATH rule that switches credentials when connecting to an LUW database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSLUWC) to allow Data Service Studio Meta discovery on LUW databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your LUW database, as instructed in the JCL, make sure to specify the correct DBTYPE. For LUW databases, specify DBTYPE=LUW.

 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKELUWG) to provide the logon credentials to each LUW instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Configuring rules and authentication for Microsoft SQL Server

Configure Server Event Facility (SEF) rules and set up authentication to provide access to Microsoft SQL Server via the 2016 Host Integration Server for HIS DRDA Service.

About this task

To complete configuration for access to Microsoft SQL Server, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to SQL Server. By default, the Data Service server will attempt to log on to SQL Server with the same user ID that was presented for logon to z/OS. A facility is provided in the Data Service server to optionally change the logon credentials for a user when accessing SQL Server.

This task uses the following tools:

AZKSMSSC

An SQL rule that allows Meta discovery on SQL Server databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEMSSG

An ATH rule that switches credentials when connecting to a SQL Server database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSMSSC) to allow Data Service Studio Meta discovery on SQL Server databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your Microsoft SQL Server database, as instructed in the JCL, make sure to specify the correct DBTYPE. For SQL Server databases, specify DBTYPE=MSSQL.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEMSSG) to provide the logon credentials to each SQL Server instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Configuring rules and authentication for Oracle DRDA

Configure Server Event Facility (SEF) rules and set up authentication to provide access to Oracle databases via the Oracle Database Provider for DRDA.

About this task

To complete the configuration for access to Oracle databases via the Oracle Database Provider for DRDA, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to Oracle AS. By default, the Data Service server will attempt to log on to Oracle with the same user ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing Oracle.

This task uses the following tools:

AZKSORAC

An SQL rule that allows Meta discovery on Oracle databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEORAG

An ATH rule that switches credentials when connecting to an Oracle database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSORAC) to allow Data Service Studio Meta discovery on Oracle databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your Oracle database, as instructed in the JCL, make sure to specify the correct DBTYPE. For Oracle, specify DBTYPE=ORACLE.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEORAG) to provide the logon credentials to each Oracle instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Configuring rules and authentication for QMF DRDA Server

Configure Server Event Facility (SEF) rules and set up authentication to provide access to QMF DRDA Server databases.

About this task

To complete the configuration for access to QMF DRDA Server databases, you must activate SEF rules and optionally set up authentication.

It is common for data centers to assign different user IDs for access to z/OS and for access to QMF DRDA Server. By default, the Data Service server will attempt to log on to QMF DRDA Server with the same user

ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing QMF DRDA Server.

This task uses the following tools:

AZKSQMFC

An SQL rule that allows Meta discovery on Oracle databases.

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEQMFG

An ATH rule that switches credentials when connecting to a QMF DRDA Server database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. Auto-enable the SQL rule SAZKXSQL(AZKSQMFC) to allow Data Service Studio Meta discovery on QMF DRDA Server databases.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or SQL to display only SQL rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq*.SAZKCNTL data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your QMF DRDA Server database, as instructed in the JCL, make sure to specify the correct DBTYPE. For QMF DRDA Server databases, specify DBTYPE=QMFDRA.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEQMFG) to provide the logon credentials to each QMF DRDA Server database. Global variables are used to define alternate authentication credential mapping for the SEF ATH Rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is restarted.

Controlling display and access for native Db2 subsystems

You can control whether native Db2 database subsystems appear in ISPF and the Data Service Studio and if attempts to connect to native Db2 subsystems are allowed.

About this task

The server parameter `DISABLEATTACH` controls whether native Db2 database subsystems appear in the ISPF and Data Service Studio applications and if attempts to connect to native Db2 subsystems are allowed.

The following table describes the settings for this parameter:

Parameter	Description	Valid values
DISABLEATTACH	Controls whether native Db2 database subsystems appear in the ISPF and Data Service Studio applications and if attempts to connect to native Db2 subsystems are allowed. YES Only data sources defined as DRDA endpoints appear in the ISPF DB2 Interface Facility (Database Control) and the Data Service Studio interface. An attempt to connect to a subsystem that does not have a DRDA configuration will be rejected. Trace Browse will show the following message: <pre>DB SUBSYSTEM xxxx IS NOT DEFINED</pre> For an attempt to connect to a DRDA data source that is disabled, Trace Browse will show the following message: <pre>DB SUBSYSTEM xxxx IS NOT OPERATIONAL</pre> NO (Default) All Db2 subsystems appear in the ISPF and Data Service Studio interfaces.	YES NO

The default setting for server parameter `DISABLEATTACH` is `NO`; however, the following statement is included in the server configuration file, which changes the setting to `YES`:

```
"MODIFY PARM NAME(DISABLEATTACH) VALUE(YES)"
```

If this override is omitted from the server configuration file, the setting will default to `NO`.

To review or update the `DISABLEATTACH` parameter setting, use the following procedure:

Procedure

1. Locate the server configuration member. The server initialization member is shipped in data set member `hlq.SAZKEXEC(AZKSIN00)` and may have been copied to a new data set for customization in the step "Copying target libraries" in the *Installation and Customization Guide*.
2. Review the following statement in your `AZKSIN00` member, and update the setting if necessary:

```
"MODIFY PARM NAME(DISABLEATTACH) VALUE(YES)"
```

Configuring access to CA IDMS data

To access CA IDMS data, you must configure the Data Service server started task JCL. You can then optionally verify access to the data.

Data Service server started task JCL changes are required to access CA IDMS software and define default CA IDMS settings.

Restrictions

The following restrictions and considerations apply when accessing CA IDMS data:

- SELECT-only support is provided.
- CA IDMS Logical Record Facility (LRF) is not supported. Virtual views provide many of the same capabilities as LRF and can be used in place of LRF.
- Data access uses CA IDMS network DML only. The CA IDMS SQL product is not required.

Note:

Server configuration parameters control the following behaviors and can be modified if necessary:

- CA IDMS run-unit management, specifically maximum run-units and a timeout value for inactive run-units
- CA IDMS access tracing

Configuring the server started task JCL

Modify the server started task JCL to access CA IDMS and define default CA IDMS settings.

Before you begin

All LOAD library data sets allocated to the Data Service server in the server started task JCL must be APF-authorized.

About this task

Modify the server started task JCL to access CA IDMS and define default IDMS settings.

Procedure

1. Add the CA IDMS load libraries to the STEPLIB, which are required for CA IDMS central version access.
2. Add the SYSCTL DD statement identifying the CA IDMS central version to access.
3. Add the SYSIDMS statement with additional environment parameters. Minimally, this data set should include a CVRETRY=OFF statement to prevent an WTOR message when the CA IDMS central version is not active.
4. Add the CA IDMS system message data set to DCMSG.

Modifying the server configuration member for CA IDMS

To optionally configure server parameters for CA IDMS, you can update your Data Service server configuration file.

About this task

The CA IDMS server parameters can assist you in configuring CA IDMS data access. In most typical environments, the default settings for these parameters will not need modification.

Procedure

1. Locate the server configuration member. The server initialization member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and may have been copied to a new data set for customization.

2. Add the following statements to your AZKSIN00 member:

The following table lists the parameters for configuring CA IDMS data access:

Parameter	Description	Valid values
MAXIDMSRUNUNITS	MAXIMUM IDMS RUN UNITS This parameter limits the number of concurrent IDMS run units that a server will start to access a CA IDMS central version. Limiting concurrent IDMS run units will prevent storage related user 3134 abends when creating run units with CA IDMS.	Positive numeric value. Default value is 4.
SQLENGIDMSRUTIMOUT	IDMS RUN UNIT INACTIVITY TIMEOUT Specifies the length of time in seconds to keep a run unit active for reuse by subsequent SQL queries in a client connection.	Positive numeric value. Default value is 60 seconds.

Verifying access to CA IDMS data

To verify access to CA IDMS data, you can optionally install a set of maps to the sample database EMPDEMO and run queries using the installed maps.

Before you begin

The CA IDMS sample database EMPDEMO must be installed in the central version you plan to access.

About this task

You can customize and run the provided IVP job AZKISIV1 to install maps to the EMPDEMO database and network schema maps to the SYSTEM database.

The following maps are installed for verification testing using the sample EMPDEMO database:

Table 4: CA IDMS EMPDEMO database maps	
Map	Description
EMPSS01_EMPLOYEE	Enables SQL access to EMPLOYEE record.
EMPSS01_OFFICE	Enables SQL access to the OFFICE record.
EMPSS01_DEPARTMENT	Enables SQL access to the DEPARTMENT record.
EMPSS01_OFFICE_EMPLOYEE	Enables SQL access to the OFFICE-EMPLOYEE set for joining the EMPSS01_OFFICE and EMPSS01_EMPLOYEE tables.
EMPSS01_DEPT_EMPLOYEE	Enables SQL access to the DEPT-EMPLOYEE set for joining the EMPSS01_DEPARTMENT and EMPSS01_EMPLOYEE tables.

The network schema maps can be used for verification purposes if the EMPDEMO database is not installed in your central version. These maps access records and sets in the CA IDMS network schema IDMSNTWK, providing SQL access to application metadata. The following table provides a subset of the installed network schema maps that can be used for verification purposes:

Table 5: CA IDMS network schema IDMSNTWK maps	
Map	Description
IDMSNWKA_S_010	Enables SQL access to the S-010 network schema record. S-010 records describe application schemas defined to your IDMS central version.
IDMSNWKA_SS_026	Enables SQL access to the SS-026 network schema record. SS-026 records describe application subschemas defined to your IDMS central version.
IDMSNWKA_SSR_032	Enables SQL access to the SSR-032 network schema record. SSR-32 records describe application subschema records defined to your IDMS central version.
IDMSNWKA_S_SS	Enables SQL access to the S-SS set for joining the IDMSNWKA_S_010 and IDMSNWKA_SS_026 tables.
IDMSNWKA_SS_SSR	Enables SQL access to the SS-SSR set for joining the IDMSNWKA_SS_026 and IDMSNWKA_SSR_032 tables.

Procedure

1. Locate the AZKISIV1 member in the *hlq.SAZKCNTL* data set.
2. Modify the JCL according to the instructions provided in the AZKISIV1 member.
3. Submit the job.
4. If the server is active, use the following instructions to refresh maps and make the maps available for use:
 - a) From the Primary Option Menu, specify option D, **Data Mapping**, and press Enter.
 - b) From the Data Mapping Facility menu, specify option 3, **Map Refresh**, and press Enter.

Results

AZKISIV1 installs CA IDMS EMPDEMO and network schema maps into the server map data set.

Configuring access to data in IBM IMS databases

To access an IMS database, you need to configure the server started task JCL and the server configuration member.

Before you begin

The server must already be installed.

About this task

Mainframe Data Service provides seamless, real-time controlled access to IMS database data. It allows JDBC access from Spark programs to IMS database data in a relational model without host programming.

Procedure

To configure and verify access to data in an IMS database, complete the following tasks.

Configuring the server started task JCL

Add IMS.SDFSRESL to the server started task JCL.

Before you begin

All LOAD library data sets allocated to the Data Service server in the server started task JCL must be APF-authorized.

About this task

You can omit this task if the IMS resident library (SDFSRESL) module is in the z/OS linklist.

Procedure

Modify the server started task JCL. If the IMS SDFSRESL is not already in the link pack area or linklist, add it to the STEPLIB.

Modifying the server configuration member for DBCTL

Enable the IMS database control (DBCTL) parameters in the server configuration member.

About this task

In order to exploit MapReduce for DBCTL, the server must have information regarding the IMS database to be used by the SQL engine optimizer. This is done by the following command query:

```
SELECT IMSRange('IMS database name')
```

Note: This command should be periodically run if the size of the database changes significantly or the index on the database changes. When this command is issued, data is gathered for each segment in the database for which a virtual table exists and is stored within the server metadata repository.

You can use a batch job to schedule this command to refresh the statistics on a specified schedule. For example:

```
//DSCLIENT EXEC PGM=xxxXMAPD,PARM='SSID=VDBS'  
//STEPLIB DD DISP=SHR,DSN=loadlibrary  
//OUT DD SYSOUT=*  
//IN DD *  
SELECT IMSRANGE('<IMS DBD Name>');
```

The server configuration member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. In the AZKSIN00 member, locate the comment “Enable IMS CCTL/DBCTL support.”
2. Enable the IMS DB parameters by changing the syntax `if DontDoThis` to `if DoThis`, and then set the parameter DBCTL to YES. The following example shows the section in the configuration member to enable:

```
if DoThis then  
do  
"MODIFY PARM NAME(DBCTL) VALUE(YES) "  
"MODIFY PARM NAME(IMSID) VALUE(IVP1) "  
"MODIFY PARM NAME(IMSNAME) VALUE(IMSX10.SDFSRESL) "  
"MODIFY PARM NAME(IMSINTHEADS) VALUE(5) "  
"MODIFY PARM NAME(IMSMAXTHEADS) VALUE(10) "  
"MODIFY PARM NAME(IMSNBABUFFERS) VALUE(0) "  
"MODIFY PARM NAME(IMSFPBUFFERS) VALUE(0)
```

```
"MODIFY PARM NAME(IMSFP OVERFLOW) VALUE(0)"
"MODIFY PARM NAME(TRACEIMSDLIEVENTS) VALUE(NO)"
```

The following table lists the parameters for configuring support for IMS DB data stores:

Parameter	Description	Valid values
DBCTL	Initialize DBCTL support.	YES NO (default value)
IMSID	IMS SSID of the DBCTL region.	Four-character name
IMSDSNAME	The name of the data set for the IMS residence library.	Data set name
IMSMINTHREADS	Minimum number of threads.	Numeric value. Default is 5.
IMSMAXTHREADS	Maximum number of threads.	Numeric value. Default is 10.
IMSNBABUFFERS	Total number of NBA buffers.	Numeric value. Default is 0.
IMSFPBUFFERS	Fast path buffers per thread.	Numeric value. Default is 0.
IMSFP OVERFLOW	Fast path overflow buffers.	Numeric value. Default is 0.
TRACEIMSDLIEVENTS	Trace IMS DLI events.	YES NO (default value)

Modifying the server configuration member for IMS Direct

Enable and configure the IMS Direct parameters in the server configuration member.

About this task

The IMS Direct feature provides map reduce and parallelism support for accessing native IMS files. This support bypasses the requirement of having to use native IMS API calls by reading the IMS database files directly, similar to how an unload utility may work. This method provides a significant improvement in performance and reduced elapsed time in processing analytical type queries.

When an IMS SQL query is run, the SQL engine for the server will determine if the request is best executed using IMS Direct (native file support) or if IMS APIs are required. The determination is based on database and file types supported as well as the size of the database. Virtual tables of the IMS segments are required.

The following types of IMS databases are currently supported by IMS Direct:

- Hierarchical direct access method (HDAM) - VSAM and OSAM
- Hierarchical indexed direct access method (HIDAM) - VSAM and OSAM
- Partitioned HDAM (PHDAM) - VSAM and OSAM
- Partitioned HIDAM (PHIDAM) - VSAM and OSAM
- Fast Path data entry database (DEDB)

When using IMS Direct, there is no locking involved when accessing the data, so updates may not be captured and deleted records may have been captured. Security is managed on the IMS native data set itself when IMS Direct is used. The user ID of the client connection must have the necessary security permissions for reading the IMS database data set(s).

IMS Direct supports access to multiple IMS subsystems and calls to compression exits and Guardium encryption and decryption exits.

Using exits

If you use compression exits or Guardium encryption and decryption exits, you can configure the server to call these exits, providing optimization.

For compression exits, the default mode of operation is to call them in TCB mode with a serialization latch held and a PST address of 0. This can be inefficient since most of the IMS Direct processing takes place in SRB mode on a zIIP. If you know enough about your compression exit, you can optimize performance of the exit by specifying it in either the `IMSDIRCMPTCBn`, or `IMSDIRCMPSRBn` statements, which are described in the procedure below. All exits are called for INIT and TERM in TCB mode.

- Decompression calls may be made in TCB mode, without serialization by specifying the name in an `IMSDIRCMPTCBn` statement. This will allow parallel threads to run without serialization, improving performance.
- Decompression calls may also be made in SRB mode, without serialization, by specifying the name in an `IMSDIRCMPSRBn` statement. This will avoid a task switch for each compressed segment, improving performance. Note that the supplied IMS compression `DFSCMPX0` exits and `DFSKMPX0` will run in SRB mode.

Guardium decryption exits require a PST and PST work area. A dummy PST with a PST work area is passed to these exits when they are specified in an `IMSDIRDECXITn` statement, which is described in the procedure. Guardium decryption exits can run in SRB mode, without serialization.

Procedure

1. Locate the server configuration member. The server initialization member is shipped in data set member `hlq.SAZKEXEC(AZKSIN00)` and may have been copied to a new data set for customization.
2. In the `AZKSIN00` member, locate the comment "Enable IMS Direct Map Reduce."
3. Enable the IMS Direct parameters by changing the syntax `if DontDoThis` to `if DoThis`, and then set the parameter `IMSDIRECTENABLED` to YES. The following example shows the section in the configuration member to enable:

```
if DoThis then
do
"MODIFY PARM NAME(IMSDIRECTENABLED) VALUE(YES) "
"MODIFY PARM NAME(IMSDIRECTBUFFERSIZE) VALUE(1024) "
"MODIFY PARM NAME(ACIINTSEGMP256) VALUE(200) "
"MODIFY PARM NAME(TRACEIMSDBREFRESH) VALUE(YES) "
"MODIFY PARM NAME(TRACEIMSDIRSTATS) VALUE(YES) "

"DEFINE IMSDBINFO",
.
.
.
end
```

The following table lists the parameters for configuring support for IMS Direct:

Parameter	Description	Valid values
ACIINTSEGMP256	The 256K ACI buffer pool. Required for IMS Direct.	Numeric value. Default is 200.
IMSDIRECTBUFFERSIZE	Specified in KB, and should be greater than the size of the largest complete IMS database record (root + all dependent segments).	Numeric value.
IMSDIRECTENABLED	Enable IMS Direct support.	YES NO (default value)

Parameter	Description	Valid values
TRACEIMSDBREFRESH	Generate trace message when IMS Direct map reduce discovery processing is performed.	YES NO (default value)
TRACEIMSDIRSTATS	Produce runtime statistics at the end of IMS Direct processing of a data set.	YES NO (default value)

4. Define your IMS subsystem using the DEFINE IMSDBINFO statement. Provide one statement for each IMS subsystem that will be used by IMS Direct.

```

"DEFINE IMSDBINFO",
  "IMSID(xxxx)",
  "SUFFIX(x)",
  "MODBLKS(your.MODBLKS)",
  "ACBLIB(your.ACBLIB)",
  "DFSRESLB(your.SDFSRESL)",
  "IMSDALIB(your.dynamic.allocation.lib)",
  "RECON1(your.RECON1)",
  "RECON2(your.RECON2)",
  "RECON3(your.RECON3)"
end

```

The following table lists the parameters used to define the IMS database:

Parameter	Description	Valid values
IMSID	The IMS subsystem identification.	Up to 4-character ID.
SUFFIX	The setting of the SUF= keyword used in the IMS Control Region.	One character. Default value is I.
ACBLIB	ACBLIB data sets contain the application control blocks (ACBs), which describe IMS applications, and data management blocks (DMBs), which describe databases and the applications that can access them.	your.ACBLIB
DFSRESLB	Load library that contains the major IMS modules.	your.SDFSRESL
IMSDALIB	Dynamic Allocation Library for IMSDBs and RECONs.	your.dynamic.allocation.lib
MODBLKS	Used to support dynamic resource definition. Contains the APPLCTN, DATABASE, RTCODE, and TRANSACT macros.	your.MODBLKS
RECON1	Primary RECONciliation dataset, which holds all of the resource information and event tracking information that is used by IMS.	your.RECON1
RECON2	An active copy of RECON1.	your.RECON2

Parameter	Description	Valid values
RECON3	Spare RECON to be used when RECON1 or RECON2 are not useable.	your.RECON3

5. (Optional) Add the following statements to configure additional IMS Direct parameters:

```
"MODIFY PARM NAME(IMSDIRECTCYLBUF) VALUE(3)"
"MODIFY PARM NAME(IMSDIRECTOSAMRECSRD) VALUE(2)"
```

Parameter	Description	Valid values
IMSDIRECTCYLBUF	Specifies the number of cylinders of data to buffer for each file processed in an IMS Direct task.	1-50. Default value is 3.
IMSDIRECTOSAMRECSRD	Specifies the number of records to read in each OSAM I/O operation. For random reads, a large number may lead to unnecessary blocks read. For sequential reads, small numbers may give decreased performance.	1-50. Default value is 2.

6. To call a compression exit, perform one of the following steps as appropriate:

- If your compression exit must be called in TCB mode but can run properly without serialization, specify your exit name in the following statement:

```
"MODIFY PARM NAME(IMSDIRCMPXITTCBn) VALUE(exitname)"
```

where *n* is a number from 1 to 10 and *exitname* is the name of the compression exit routine.

- If your exit can run properly in SRB mode without serialization, specify your exit name in the following statement:

```
"MODIFY PARM NAME(IMSDIRCMPXITSRBn) VALUE(exitname)"
```

where *n* is a number from 1 to 10 and *exitname* is the name of the compression exit routine.

If neither of these conditions apply, do not specify the name of your compression exit.

Note: Review ["Using exits"](#) for more information about configuring calls to compression exits.

Parameter	Description	Valid values
IMSDIRCMPXITTCBn	Specifies the name of a compression exit that can be safely called without serialization. Up to 10 exit names can be specified, where <i>n</i> is a number from 1 to 10. Since the server runs multiple threads in parallel, this feature provides optimization by eliminating the possible serialization conflicts between threads.	Name of compression exit routine

Parameter	Description	Valid values
IMSDIRCMPXITSRB <i>n</i>	Specifies the name of a compression exit that can be safely called without serialization and in SRB mode. Up to 10 exit names can be specified, where <i>n</i> is a number from 1 to 10. Since multiple exit names can be called without serialization and without switching off the zIIP (SRB mode) into TCB mode (GP processor), this feature provides optimization by eliminating the need to switch tasks for each exit call. The IBM supplied compression exits DFSCMPX0 and DFSKMPX0 will run safely in SRB mode. They can be specified in IMSDIRCMPXITSRB1 and IMSDIRCMPXITSRB2.	Name of compression exit routine

7. To call Guardium encryption and decryption exits, add the following statement:

```
"MODIFY PARM NAME(IMSDIRDECXITSRBn) VALUE(exitname)"
```

where *n* is a number from 1 to 20 and *exitname* is the name of the Guardium exit routine.

Note: Review ["Using exits"](#) for more information about configuring calls to Guardium encryption and decryption exits.

Parameter	Description	Valid values
IMSDIRDECXITSRB <i>nn</i>	Specifies the name of the Guardium encryption and decryption exit routine. Up to 20 exit names can be specified, where <i>nn</i> is a value from 1 to 20.	Name of Guardium exit routine

Configuring access to IBM MQ

For access to IBM MQ (MQ) data, you must modify the server started task, configure the server configuration member, and set virtual table options.

Data Service provides SQL-only query access to MQ queues using virtual tables. Data in MQ queues is described using COBOL or PLI data descriptions taken from copybooks or programs.

IBM MQ for z/OS Versions 7.5 and newer are supported.

Note: Server configuration parameters control MQ tracing and can be modified if necessary.

Configuring the server started task JCL

Modify the server started task JCL to access IBM MQ data.

Before you begin

All data sets that you add to the server started task JCL STEPLIB must be APF-authorized.

About this task

Modify the server started task JCL to access IBM MQ data. You can skip this task if the IBM MQ load module is in the z/OS linklist or link pack area.

Procedure

Add the IBM MQ load library to the server started task JCL STEPLIB.

Modifying the server configuration member for IBM MQ

To enable support for MQ data, you must update your Data Service server configuration file.

About this task

To be able to access MQ data in virtual tables, enable the feature in the server configuration file, as described in the following procedure.

Procedure

1. Locate the server configuration member. The server initialization member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and may have been copied to a new data set for customization.
2. Add the following statement to your AZKSIN00 member:

The following table describes this parameter:

Parameter	Description	Valid values
MQACTIVE	Initialize IBM MQ support. This parameter must be set to YES to access MQ queues.	YES NO (default value)

Configuring virtual table rules for IBM MQ

Configure Server Event Facility (SEF) rules to support IBM MQ data.

About this task

You can configure VTB rule options to control the MQ data access feature. These options control inclusion of the MQ message descriptor meta data fields in the virtual tables, how to handle truncated messages, and whether to perform destructive reads. Sample VTB rule AZKMDLMQ documents these settings.

When accessing MQ data with sample rule AZKMDLMQ (or equivalent options) enabled, tables prefixed with MDLMQ_* are filtered, and the map name is extracted by removing the MDLMQ_ prefix. For example, the following query will execute the rule and query virtual table MQ_CSQ7_TRADE:

```
SELECT * FROM MDLMQ_MQ_CSQ7_TRADE
```

Use the following procedure to configure the sample rule AZKMDLMQ.

Note: Sample rule AZKMDLMQ is intended to be used as a model and may require customization. When customizing this rule, additional logic may need to be added if different VTB variable settings are required for different MQ queues.

Procedure

1. Customize the server configuration member (AZKSIN00) to enable virtual table rule events by configuring the SEFVTBEVENTS parameter in the member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES) "
```

2. Access the VTB rules, as follows:
 - a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
 - b) Specify option 2, **SEF Rule Management.**

- c) Enter VTB for **Display Only the Ruleset Named**.
3. Customize the AZKMDLMQ rule, as follows:
- Specify S next to AZKMDLMQ to edit the rule.
 - Update the rule options as needed. The following table describes the VTB rule options that support MQ data access.

VTB variable	Description	Valid values
vtb.optbmqdg	Delete messages during retrieval. When set to 1, SQL queries will remove messages from the queue if ALL messages in the queue are successfully retrieved by the server. Retrieval of MQ messages will use non-browse (destructive) MQGET calls with syncpoint control. Once all messages are delivered to the server, they will be deleted from the queue. If a failure occurs before all messages are retrieved, an MQBACK call will be issued to restore messages to the queue that have been retrieved so far. Note that an MQCMIT will be issued and messages deleted if the IBM MQ syncpoint limit is reached. A failure after MQCMIT will not be able to restore messages as they have been permanently deleted.	0 (Default) 1
vtb.optbmqim	When set to 1 for an MQ virtual table, the MQ Series Message Descriptor (MQMD) meta data fields will be added to the virtual table as columns and returned with each result row. These columns are prefixed with the value MQMD_.	0 (Default) 1
vtb.optbmqtc	By default, a truncation error reading an IBM MQ message will result in a query failure. When set to 1, MQ Series access ignores truncated message warnings and returns data received.	0 (Default) 1

- c) Save your changes and exit the editor.
4. Enable the rule by specifying E next to AZKMDLMQ and pressing Enter.
5. Set the rule to Auto-enable by specifying A next to AZKMDLMQ and pressing Enter.
Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

Configuring access to sequential files

No modifications are needed to configure the SQL interface to access sequential files. However, you should verify access to sequential files. Optionally, you can specify the number of tracks to read ahead when reading sequential data sets for individual requests.

Reading ahead tracks for sequential file access

You can use a Server Event Facility (SEF) rule to specify the number of tracks to read ahead (MULTACC) when reading sequential data sets for individual requests.

About this task

Using a virtual table (VTB) rule, you can specify the number of tracks to read ahead (the MULTACC parameter value) for MapReduce sequential file access for individual requests. This support overrides the value in the server parameter ACIMAPREDUCETRACKS (NUMBER OF MAP REDUCE TRACKS TO READ) for individual requests. Sample VTB rule AZKMLTAC is provided.

To override the MULTACC value, you must enable the AZKMLTAC rule and use the MACC_*nn*_ prefix for table names in your SQL statement.

The AZKMLTAC rule is invoked every time a table with the prefix MACC_*nn*_ is found in the SQL statement. The following format is expected:

```
MACC_nn_virtualtablename
```

Where:

- *nn* is the number of tracks to read ahead (the MULTACC value) when reading sequential data sets
- *virtualtablename* is the name of the virtual table

For example:

```
SELECT * from MACC_15_STAFF_SSEQ ;
```

The following message is displayed in the Server Trace:

```
AZK1000I VTB.OPTBMACC set to 15
```

Use the following procedure to set up the rule.

Procedure

1. Customize the server configuration member (AZKSIN00) to enable virtual table rule events by configuring the SEFVTBEVENTS parameter in the member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES) "
```

2. Access the VTB rules, as follows:
 - a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
 - b) Specify option 2, **SEF Rule Management.**
 - c) Enter VTB for **Display Only the Ruleset Named.**
3. Enable the rule by specifying E next to AZKMLTAC and pressing Enter.
4. Set the rule to Auto-enable by specifying A next to AZKMLTAC and pressing Enter.

Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

Configuring access to native VSAM

No modifications are required to configure the SQL interface for native VSAM. However, you should verify that the server has access to native VSAM. Optionally, you can control the data buffer (BUFND) and the index buffer (BUFNI) values for VSAM files either globally or for individual requests.

Before you begin

The server must already be installed.

About this task

Mainframe Data Service provides seamless, real-time controlled access to native VSAM. It allows JDBC access from Spark programs to native VSAM data in a relational model without host programming.

Verifying access to native VSAM

Verify native VSAM data access by creating a sample VSAM file and a corresponding virtual table and running a query that accesses the VSAM data.

Procedure

1. Create the sample VSAM file on the mainframe that hosts the server.
Run the AZKGNSTF member in the *hlq.SAZKCNTL* data set to allocate and load the sample VSAM file.
The job should complete with a condition code of 0.
2. Create the *staffvs* virtual table, and run a query that returns a result set.
Run the AZKIVVS1 member in the *hlq.SAZKCNTL* data set to perform a batch extract of the sample VSAM file listing and create a virtual table that is used to format the result set that is returned from the VSAM file.
The job should complete with a condition code of 0.
3. Verify that the SQL results contained in the AZKIVVS1 member are valid.

Modifying the data and index buffer values for VSAM files

You can change the data and index buffer values for VSAM files.

About this task

You can control the data buffer (BUFND) and the index buffer (BUFNI) values for VSAM files either globally or for individual requests, as follows:

- To change the values globally, you must add the required parameters to your Data Service server configuration file. The following table lists these parameters:

Parameter	Description	Valid values
SQLENGVSAMDATABUFF	Specifies the number of data buffers for VSAM files. Default: 20	Numeric value.
SQLENGVSAMINDEXBUFF	Specifies the number of index buffer for VSAM files. Default: 30	Numeric value.

- To change the values for individual requests, you can use virtual table (VTB) rules. Sample VTB rules AZKBUFND and AZKBUFNI are provided.

To override your index buffer or data buffer values, you must enable the respective rule and use the appropriate BUF prefix for table names in your SQL statement, as follows.

– To override the data buffer (BUFND) value:

Use sample rule AZKBUFND. The AZKBUFND rule is invoked every time a table with the prefix BUFND_ is found in the SQL statement. The following format is expected:

```
BUFND_nn_virtualtablename
```

Where:

- *nn* is the number of data buffers (BUFND) for the VSAM data sets
- *virtualtablename* is the name of the virtual table

For example:

```
SELECT * from BUFND_30_STAFF_VSAM ;
```

The following message is displayed in the Server Trace:

```
AZK1000I VTB.OPTBVSND set to 30
```

– **To override the index buffer (BUFNI) value:**

Use sample rule AZKBUFNI. The AZKBUFNI rule is invoked every time a table with the prefix BUFNI_ is found in the SQL statement. The following format is expected:

```
BUFNI_nn_virtualtablename
```

Where:

- *nn* is the number of index buffers (BUFNI) for the VSAM data sets
- *virtualtablename* is the name of the virtual table

For example:

```
SELECT * from BUFNI_30_STAFF_VSAM ;
```

The following message is displayed in the Server Trace:

```
AZK1000I VTB.OPTBVSNI set to 30
```

Procedure

1. To change the values globally, perform the following steps:

- a) Locate the server configuration member. The server initialization member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and may have been copied to a new data set for customization in the step "Copying target libraries" in the *Installation and Customization Guide*.
- b) Add the following statements to your AZKSIN00 member:

```
"MODIFY PARM NAME(SQLENGVSAMDATABUFF) VALUE(20)"  
"MODIFY PARM NAME(SQLENGVSAMINDEXBUFF) VALUE(30)"
```

2. To change the values for individual requests, perform the following steps:

- a) Customize the server configuration member (AZKSIN00) to enable virtual table rule events by configuring the SEFVTBEVENTS parameter in the member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES)"
```

- b) Access the VTB rules, as follows:

- 1) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
- 2) Specify option 2, **SEF Rule Management.**
- 3) Enter VTB for **Display Only the Ruleset Named.**

- c) Enable each rule as follows:

- Specify E next to AZKBUFND and press Enter.
- Specify E next to AZKBUFNI and press Enter.

- d) Set each rule to Auto-enable as follows:

- Specify A next to AZKBUFND and press Enter.
- Specify A next to AZKBUFNI and press Enter.

Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

- e) Use the appropriate BUF prefix for table names in your SQL statement.

Configuring access to zFS files

The Data Service server is already configured to support zFS files. No modifications are needed to configure access to zFS files.

Configuring access to SMF data for IT Operational Analytics

IT Operational Analytics (ITOA) allows you to retrieve, analyze, and report data for IT operations. System information can be logged using the IBM System Management Facility (SMF) and the native Data Service server logging feature. Logging allows you to collect various system and operations-related information.

Before you begin

Verify that the following IBM APARs have been applied:

- APAR OA49263 (www-01.ibm.com/support/docview.wss?uid=isg1OA49263). This APAR provides real-time SMF support and is a requirement for the configuration of real-time SMF data access. (The closed date for this APAR is 2016-08-31.)
- APAR OA48933 (www-01.ibm.com/support/docview.wss?uid=isg1OA48933). This APAR is required to address accessing log streams. SMF log stream configuration is required for in-memory resource support. (The closed date for this APAR 2015-11-24.)

About this task

Virtual tables for SMF are provided in the *hlq.SAZKSMAP* data set.

The following options are available to access the SMF data:

- Reading data from SMF data sets - SMF information is recorded in MANx data sets. When a data set gets full, the data is processed via IFASMFDP. When defining global variables for accessing SMF data in data sets, the output of IFASMFDP is used.
- Reading data from log streams - SMF information is recorded in multiple log streams and data can be read directly from the log streams. Log stream recording is determined by the data set name beginning with IFASMF that is used in the VTB rule for SMF.
- Reading SMF data from in-memory (real-time) - SMF information is read directly from the system buffer. SMF information is read in real time. There are two interfaces to real-time SMF data, which connect to the in-memory resource at different times, as follows:
 - At product initialization. This interface connects to the in-memory resource at product initialization and continuously reads from the API to maintain a buffer of recent SMF activity. This buffer can be queried, and its contents will be returned, followed by an end-of-data indication.
 - At the time of the request. This interface connects to the in-memory resource at the time of the request and streams the SMF data to the requester in real time. A request to this named stream is considered non-ending, and data will continue to flow until the request is canceled or the server is stopped.

When defining the global variables for SMF, the data set can be either a log stream or a SMF dump data set from IFASMFDP. The log stream data set is recommended for access to near real-time data.

To configure access to IT Operational Analytics data, see the following topics:

- [“Configuring access to System Management Facility \(SMF\) files” on page 49](#)
- [“Configuring access to SYSLOG files” on page 52](#)
- [“Configuring access to OPERLOG files” on page 53](#)

Configuring access to System Management Facility (SMF) files

To configure access to System Management Facility (SMF) files, you need to configure the server started task JCL, the server configuration member, and the server virtual table member. To enable reading SMF data real-time using log streams, you must have the SMFPRMxx member in the system PARMLIB data set configured to use both log streams and in-memory resources. Follow the steps in this section to use SMF GDG data set names, or to use dynamic data set names.

About this task

SMF data set names are dynamic in local environments and require SEF rules enablement and optionally Global Variables set to specific values to provide data set names to the virtual tables and views when using SMF data set or log stream configurations.

You can choose either GDG data set name support or dynamic data set name support, or both, to quickly access your SMF data. These two options are provided for your convenience to help you start accessing your SMF data. Custom rules may need to be developed to use your local naming convention to access your SMF files.

Procedure

1. Configure the server started task JCL by concatenating the *hlq.SAZKSMAP* data set to the AZKMAPP DD statement to add all maps for SMF.

2. Customize the server configuration member.

To enable virtual table rule events, configure the SEFVTBEVENTS parameter in the AZKSIN00 member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES) "
```

Verify the VTB ruleset name:

```
"DEFINE RULESET NAME(VTB) "  
"RULETYPE(VTB) "  
"DSNAME(' '||SHLQ2|||.SAZXVTB') "
```

If there were any changes to AZKSIN00, recycle the server started task.

3. To enable real-time access to SMF data, add the following statements to the AZKSIN00 member after the GLOBAL PRODUCT OPTIONS statement.

```
IF DoThis  
  THEN DO  
    "DEFINE SMF NAME(IFASMF.INMEM) ",  
    "STREAM(IFASMF.INMEM.STREAM) ",  
    "BUFSIZE(500) ",  
    "TIME(0) "  
END
```

Note: You must have the SMFPRMxx member in the system PARMLIB data set configured to use log streams and in-memory resources.

Parameter	Description	Valid values
NAME	Specifies the name of the in-memory resource. This value must match the name of a resource defined to SMF with the INMEM parameter. If this parameter is included, the in-memory API will be read continuously and a buffer of the most recent records will be maintained. Either this parameter or the STREAM parameter, or both, must be specified.	This parameter must contain the name of an in-memory resource defined to SMF with the INMEM statement. The format of the name is defined by SMF configuration, which is 1-26 characters and must begin with IFASMF.
STREAM	Specifies the name of the streaming in-memory feature. If this name is specified on a SELECT statement, a dynamic connection will be made to the SMF in-memory API and records will be streamed to the caller in real time. Either this parameter or the NAME parameter, or both, must be specified.	If a NAME parameter is also supplied, the in-memory resource named in that parameter will be connected to and the value of this parameter can be any name, 1-26 characters. If the NAME parameter is not supplied, this parameter must contain the name of an in-memory resource defined to SMF with the INMEM parameter. If both NAME and STREAM are provided, the names must be different.
BUFSIZE	Indicates how much SMF data (megabytes) will be retained in memory for queries. If the buffer fills up, the oldest data will be discarded. In parallel, SMF is recording these records to a log stream. This parameter applies to the resource named in the NAME parameter.	1-10,000
TIME	Indicates how long (in minutes) to keep SMF data in memory. Older data will be discarded. Specifying 0 indicates no time limit and data will be retained until the buffer fills up. This parameter applies to the resource named in the NAME parameter.	0-1440

4. To use SMF data in compressed log streams, add the following statement to the AZKSIN00 member:

```
"MODIFY PARM NAME(ZEDCCOMPRESSION) VALUE(YES) "
```

Note: You must have the SMFPRMxx member in the system PARMLIB data set configured to use compressed log streams, and the zEDC Express hardware feature must be installed. For more information about enabling zEDC, see the *Administrator's Guide*.

5. Enable reading SMF data from GDG data sets and access to SMF data using dynamic data set names by enabling Server Event Facility rule AZKSMFT1 in the VTB ruleset. You can select from a GDG data set, any SMF dump data set, a log stream data set, or the in-memory stream. Activate your options by customizing the rule.

a) Use the following steps to enable rule AZKSMFT1 in the VTB ruleset:

- 1) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
- 2) Specify option 2, **SEF Rule Management.**
- 3) Enter VTB for **Display Only the Ruleset Named.**
- 4) Enable the rule by specifying E and pressing Enter.
- 5) Set the rule to Auto-enable by specifying A and pressing Enter.

Setting the rule to Auto-enable activates the rule automatically when the server is re-started.

b) Configure the access method using one or more of the following methods:

- Review the information in the rule for the instructions on setting Global Variables that will be used by the rule. Navigate one screen back on the ISPF panel, or start over by going to option E, **Rules Mgmt.**, and then option 1, **Global Variables.** In the Global Variables display, perform the following steps:
 - 1) Change Global Prefix to GLOBAL2.
 - 2) Select SMFTBL2 by entering S next to the SMFTBL2 data set.
 - 3) Configure the SMF data access option. DEFAULT should have corresponding SMF dump data set names if used. This option can be used to specify the source SMF, such as GDGBASE, INMEM, and LOGSTREAM.

Note:

VTB rules and global variables may be used to reference a GDG data set, any SMF dump data set, a log stream data set, or the in-memory stream. For example:

```
GLOBAL2.SMFGBL2.YESTERDAY = "YOUR.DATASET.SMFDUMP(-1)"
GLOBAL2.SMFGBL2.M2 = "YOUR.DATASET.SMFDUMP(-2)"
GLOBAL2.SMFGBL2.M3 = "YOUR.DATASET.SMFDUMP(-3)"
GLOBAL2.SMFGBL2.M4 = "YOUR.DATASET.SMFDUMP(-4)"
GLOBAL2.SMFGBL2.M5 = "YOUR.DATASET.SMFDUMP(-5)"
GLOBAL2.SMFGBL2.IM = "IFASMF.INMEM"
GLOBAL2.SMFGBL2.IM2 = "IFASMF.INMEM2"
GLOBAL2.SMFGBL2.LOG = "LOGSTREAM.dataset.name"
```

- Pass a dynamic data set name for SMF tables using the following format for the table name in the SQL statement:

```
TableMapName__DataSetName
```

Where DataSetName is prefixed by two underscores (__) and the periods in the data set name are replaced with single underscores (_).

For example, SELECT * FROM SMF_01400__DATA_SET_NAME would translate into an SQL query of SELECT * FROM SMF_14000 and access the data set DATA.SET.NAME.

- Pass a dynamic data set name for SMF virtual views using the following format for the virtual view name in the SQL statement:

```
ViewMapName__DataSetName
```

Where DataSetName is prefixed by two underscores (__) and the periods in the data set name are replaced with single underscores (_).

For example, SELECT * FROM SMFV_01400__DATA_SET_NAME would translate into an SQL query of SELECT * FROM SMFV_01400 and access the data set DATA.SET.NAME.

Configuring access to SYSLOG files

To configure access to system log (SYSLOG) files, you need to configure the server configuration member and the server virtual table rules.

About this task

Virtual table rules are provided that support the processing of SYSLOG files and vary based on the type of file name used for your SYSLOG data sets. Each of the rules for SYSLOG processing requires that the table names in the SQL begin with SYSLOG. The following rules are provided:

AZKSYSLG

This rule uses a global variable to specify the name of the data set to use for the SYSLOG data.

AZKSYSL2

This rule supports the use of generation data group (GDG) data set names. One of the following formats is expected:

- `SYSLOG_GDG_nnnn`

Where *nnnn* is a relative GDG number (between 0 and 9999) that is appended to the GDG base name value that is obtained from the GLOBAL2.SYSLOG.GDGBASE variable. For example, if the table name as specified in the SQL statement is `SYSLOG_GDG_1`, then the data set name returned by this rule is `HLQ.SYSLOG(-1)`, depending on the value in GLOBAL2.SYSLOG.GDGBASE.

- `SYSLOG_DSN_suffix`

Where *suffix* is used as the last part of a global variable of the form GLOBAL2.SYSLOG.*suffix* in order to look up the name of the data set to be used. If this variable does not exist, the data set name specified in GLOBAL2.SYSLOG.DEFAULT is used to read the SYSLOG records.

By using global variables, you do not need to modify the code in the rule. The following are some examples of global variables that can be set up to be used in conjunction with this rule:

Global Prefix: GLOBAL2.SYSLOG			
S	Subnode Name	Nodes	Subnode Value

	GDGBASE	0	HLQ.SYSLOG
	DEFAULT	0	HLQ.SYSLOG(0)
	TODAY	0	HLQ.SYSLOG(0)
	YESTERDAY	0	HLQ.SYSLOG(-1)

AZKSYSL3

This rule lets you dynamically specify in your SQL the name of the data set to use when processing SYSLOG files. In the SQL, the table name must begin with the prefix SYSLOG; the rest of the table name is used by the rule to determine the actual data set name to use for processing the SYSLOG records.

The following format is expected:

`SYSLOG__DataSetName`

Where *DataSetName* is preceded by two underscores (__) and the periods in the data set name are replaced with single underscores (_). For example, `SELECT * FROM SYSLOG__DATA_SET_NAME` would translate into an SQL query of `SELECT * FROM SYSLOG` and access the data set `DATA.SET.NAME`.

To use one of the rules, you must enable the rule and use the prefix SYSLOG for table names in your SQL statement. The enabled rules are invoked every time a table with the prefix SYSLOG is found in the SQL statement.

Use the following procedure to set up the rules.

Procedure

1. Customize the server configuration member.

To enable virtual table rule events, configure the SEFVTBEVENTS parameter in the AZKSIN00 member, as follows:

```
"MODIFY PARM NAME(SEFVTBEVENTS) VALUE(YES) "
```

2. Access the VTB rules, as follows:
 - a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
 - b) Specify option 2, **SEF Rule Management.**
 - c) Enter VTB for **Display Only the Ruleset Named.**
3. For AZKSYSLG, customize the rule, as follows:
 - a) Specify S next to AZKSYSLG to edit the rule.
 - b) Customize the rule with the SYSLOG data set name.
 - c) Save your changes and exit the editor.

Note: For AZKSYSL2 and AZKSYSL3, no customization of the rule is needed.

4. Enable each rule by specifying E next to the member name and pressing Enter.
5. Set each rule to Auto-enable by specifying A next to the member name and pressing Enter.

Setting a rule to Auto-enable activates the rule automatically when the server is re-started.
6. If global variables are needed, set up the SYSLOG global variable.

Configuring access to OPERLOG files

No modifications are needed to configure the Data Service server to access OPERLOG data; however, OPERLOG must be active in a system logger log stream.

About this task

Use the following procedure to verify that OPERLOG is active in a system logger log stream.

Procedure

To display the active medium where messages are recorded, enter the following command:

```
D C,HC
```

The following results are expected:

```
CNZ4100I 15.19.16 CONSOLE DISPLAY 056
CONSOLES MATCHING COMMAND: D C,HC
MSG:CURR=0      LIM=9000 RPLY:CURR=0      LIM=9999  SYS=P02      PFK=00
HARDCOPY LOG=(SYSLOG,OPERLOG) CMDLEVEL=CMDS
ROUT=(ALL)
LOG BUFFERS IN USE: 0      LOG BUFFER LIMIT: 9999
```

Configuring access to ADDI

To use IBM Application Discovery and Delivery Intelligence (ADDI) information for creating virtual maps that access VSAM and sequential data, you must configure the server for ADDI access.

System requirements

The following system requirements apply:

- IBM Application Discovery Suite Version 5.0 or newer

- Microsoft Host Integration Server (HIS) 2016 or higher. The SYSIBM views that are part of the Microsoft HIS Software Development Kit must be installed as part of the HIS installation.
- Microsoft SQL Server 2012 Enterprise or Express or higher

Restrictions

The following restrictions and considerations apply when using ADDI to access VSAM and sequential data sets:

- Virtual table creation is restricted to data sets in the ADDI project that are processed by COBOL programs using JCL. Data sets accessed using CICS as well as other databases (such as IMS, CA IDMS, or Adabas) are not supported.
- Virtual table mapping is only supported through the Data Service Studio. No batch utilities or ISPF interfaces are provided to map tables.

Configuration steps

The following configuration steps are required to use ADDI to access VSAM and sequential data:

1. Install virtual tables. See [“Installing virtual tables and virtual target maps for ADDI access”](#) on page 54.
2. Define ADDI project in the server configuration member. See [“Modifying the configuration member for ADDI access”](#) on page 55.
3. Activate virtual table rules. See [“Configuring virtual table rules for ADDI”](#) on page 58.
4. Define credentials for target database(s). See [“Configuring authentication for ADDI”](#) on page 59.

Installing virtual tables and virtual target maps for ADDI access

Install virtual tables and virtual target maps for IBM Application Discovery and Delivery Intelligence (ADDI) access.

About this task

The Data Service Studio reads the ADDI project using virtual tables and views installed as part of server set up. The following maps are distributed in XMIT format in the SAZKSAMP member AZKIAMPD:

ZIADTSPR

Virtual target system TSIAD_PROJECT1 for external subsystem named IAD1.

ZIADT001-ZIADT021

Virtual tables that map tables in the ADDI project. Each virtual table uses the name of the corresponding ADDI project table with the added prefix IAD_. For example, SQL Server table `dbo.Variables` has a virtual table name of `IAD_VARIABLES`.

ZIADV001-ZIADV002

Virtual views on the IAD_ virtual tables used by the Data Service Studio to read ADDI data. These views are all prefixed with IADV_ (for example, `IADV_DATASETS`). All data access from the studio is performed using virtual views.

These maps are not installed by default. Use the following procedure to install these maps.

Procedure

1. Locate the AZKIAMPD member in the *hlq*.SAZKCNTL data set.
2. Modify the JCL according to the instructions provided in the AZKIAMPD member.
3. Submit the job.

The virtual tables and virtual target maps are installed.

Modifying the configuration member for ADDI access

Enable and configure the parameters for IBM Application Discovery and Delivery Intelligence (ADDI) in the server configuration member.

About this task

The server configuration member contains a sample DATABASE definition that defines the first ADDI project. The initial definition is named IAD1 and is disabled.

When enabling the database definition for the first ADDI project, the LOCATION and IPADDR parameters must be set to the correct project name and IP address of the Microsoft HIS DRDA Provider Service for SQL Server. The LOCATION provides the name of the SQL Server project, and IPADDR(...) PORT(...) provide the TCP/IP information for the HIS DRDA Service. DOMAIN(...) can be used instead of IPADDR to provide the DNS of the HIS DRDA Service. The subsystem NAME(IAD1) should not be changed because a target subsystem map is configured to use this name for the virtual tables accessing the ADDI project.

For multiple ADDI projects, see [“Adding an ADDI project”](#) on page 57.

The server configuration member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

1. In the AZKSIN00 member, locate the comment “Sample IBM Application Discovery configuration”.
2. Enable the ADDI parameters by changing the syntax `if DontDoThis` to `if DoThis`. The following example shows the section in the configuration member to enable:

```
/*-----*/
/* Sample IBM Application Discovery configuration using DRDA to */
/* communicate with a Microsoft SQLServer database.          */
/*-----*/
if DoThis then do
  "DEFINE DATABASE TYPE(MSSQL)"
  "NAME(IAD1)"
  "LOCATION(EZ_Project1)"
  "DDFSTATUS(ENABLE)"
  "SECMEC(USRIDPWD)"
  "IPADDR(:.FFFF:0.0.0.0)"
  "PORT(446)"
  "CCSID(37)"
  "IDLETIME(0)"
end
```

The following table lists the parameters for configuring support for ADDI:

Parameter	Description	Valid values
TYPE	Database type. Because ADDI stores information in Microsoft SQL Server, this value must be MSSQL.	MSSQL

Parameter	Description	Valid values
NAME	The database name as known to the server. The first definition must be IAD1 because the target system map names this as the subsystem to access for ADDI. For additional ADDI projects, subsystems can have any name since you must also create a virtual target system to point to it; however, it recommended that the name start with IAD. <i>(Required)</i>	A valid value consists of 1 - 4 characters. For example, IAD1.
LOCATION	Name of the database for the ADDI project. The LOCATION parameter must be set to the correct database name of the target MSSQL server. <i>(Required)</i>	A valid value is a string 1 - 16 characters.
DDFSTATUS	The DDF activation status <i>(Required)</i>	ENABLE Make this DDF definition active within Data Service server. DDFSTATUS should always be ENABLE for TYPE(MSSQL). DISABLE DDF endpoint is not used. This value disables the MSSQL database. This value should only be used if the database is off-line or otherwise not available for access.
SECMEC	Security mechanism. The DRDA security mechanism for authentication with the HIS DRDA Service for SQL Server. The SECMEC setting for TYPE(MSSQL) must match the HIS DRDA Service configuration.	USRIDPWD User ID and password USRIDONL User ID only USRENCPWD Encrypt the password only EUSRIDPWD Encrypt the user ID and password

Parameter	Description	Valid values
IPADDR	Specify the IPV4 or IVP6 address of the target MSSQL server. Use DOMAIN instead of IPADDR to supply the DNS of the target HIS DRDA Server for SQL Server. Use DOMAIN if the IPADDR or the HIS DRDA Service Provider can change. Either DOMAIN or IPADDR is required, but not both.	A valid IPV4 or IVP6 address set to the correct remote IP address for the system running Microsoft SQL Server.
DOMAIN	The part of a network address that identifies it as belonging to a particular domain. Use DOMAIN instead of IPADDR to supply the DNS of the target HIS DRDA Server for SQL Server. Use DOMAIN if the IPADDR or the HIS DRDA Service Provider can change. Either DOMAIN or IPADDR is required, but not both.	No default value.
PORT	The TCP/IP port defined for Microsoft HIS DRDA Service Provider. For TYPE(MSSQL), the standard HIS default is 446. (Required)	A valid 1-5 numeric string.
CCSID	Specify the EBCDIC single-byte application CCSID (Coded Character Set Identifier). (Required)	Refer to the Microsoft SQL Server documentation for a list of valid CCSIDs. Refer to the ISV documentation on HIS DRDA Service to SQL Server. For USA, this value is 037.
IDLETIME	This setting is not used for TYPE(MSSQL).	0

Adding an ADDI project

Perform required configuration steps to add an ADDI project.

About this task

For multiple ADDI projects, you must perform configuration steps to define each additional ADDI project. The following requirements apply when maintaining multiple ADDI projects:

- For the first instance of an ADDI project:
 - The database name in the must be IAD1.
 - The target system for the name IAD1 is automatically installed with the ADDI maps, as described in [“Installing virtual tables and virtual target maps for ADDI access” on page 54.](#)

- For subsequent ADDI projects:
 - It is recommended that the database name start with IAD.
 - The target system must start with TSIAD.

Perform the following procedure for each additional ADDI project.

Procedure

1. Repeat the database definition in the configuration member and make the following changes:
 - a) Change the NAME value to a unique name (for example, IAD2).
 - b) Change the LOCATION value to match the Microsoft SQL Server project name containing the ADDI project you need to access.

For information about the database definition parameters, see [“Modifying the configuration member for ADDI access”](#) on page 55.
2. Define a new virtual target system using the studio. The name of the virtual target system must start with TSIAD. This can be done in the Data Service Studio by selecting the **Create Virtual Target System** in the **Server** tab under the **SQL > Target Systems > DBMS** node of the tree. The connection value in each definition must match the NAME value defined in the DATABASE definition in the configuration member.
3. If required, create authentication information using the AZKDRATH batch utility.

Configuring virtual table rules for ADDI

Configure Server Event Facility (SEF) rules to support multiple projects using common virtual table and view definitions.

About this task

To support multiple projects using common virtual table and view definitions, VTB rules AZKIADTB and AZKIADVW provide support to process tables starting with IAD_ and views starting with IADV_.

AZKIADTB

This table rule looks at the base view of a query for double underscores “__” and uses the data after the underscores to update the target subsystem for the query.

AZKIADVW

This view rule looks for the double underscores and removes them from the view name to process.

With the rules activated, the Data Service Studio can suffix the view names with __SSID for all calls and process multiple ADDI projects using a single set of maps.

These rules must be activated regardless of the number of ADDI projects to be enabled.

Use the following procedure to set up these rules.

Procedure

Use the following steps to enable rules AZKIADTB and AZKIADVW in the VTB ruleset:

- a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
- b) Specify option 2, **SEF Rule Management.**
- c) Enter VTB for **Display Only the Ruleset Named.**
- d) Enable the rules by specifying E and pressing Enter.
- e) Set the rules to Auto-enable by specifying A and pressing Enter.

Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

Configuring authentication for ADDI

Configure authentication for communicating with the IBM Application Discovery and Delivery Intelligence (ADDI) project.

About this task

It is common for data centers to assign different user IDs for access to z/OS and for access to SQL Server. By default, the server will attempt to log on to SQL Server with the same user ID that was presented for logon to z/OS. A facility is provided in the server to optionally change the logon credentials for a user when accessing SQL Server.

When communicating between the Data Service server and the ADDI project, you must define what credentials to use in MSSQL connections if z/OS users are not defined as users to SQL Server. To accomplish this, the following tools are provided:

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. Use this utility to define alternate logon information for the Data Service server started task and z/OS users. This utility places SQL Server authentication information in GLOBALU system variables for connecting to ADDI projects. You can also use this utility to list existing credential information.

AZKEMSSG

An ATH rule that swaps z/OS user information with SQL Server authentication information defined using the AZKDRATH utility. This rule uses AES encrypted passwords stored as GLOBALU system variables.

You can use any of the following options for authentication:

- Use z/OS IDs for authentication
- Add a global default user definition using sample job AZKDRATH and enable ATH rule AZKEMSSG
- Add authentication information for specific mainframe users using sample job AZKDRATH and enable ATH rule AZKEMSSG

Network administrators may need to open ports for DRDA communication between the z/OS host and the Microsoft SQL Server machine(s) hosting ADDI projects. The default port for Microsoft SQL Server access is 446.

If z/OS user IDs are not defined to Microsoft SQL Server, use the following procedure to define alternate authentication information for the started task and z/OS users requiring access to this feature:

Procedure

1. Use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq.SAZKCNTL* data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your ADDI project, as instructed in the JCL, make sure to specify the correct DBTYPE. For ADDI projects, specify DBTYPE=MSSQL.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
2. Auto-enable the SEF ATH rule SAZKXATH(AZKEMSSG) to switch credentials when connecting to ADDI using DRDA. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.

- d) Set Auto-Enable for the AZKEMSSG rule member by entering A and pressing Enter.

Configuring access to RAA

To use IBM Rational Asset Analyzer (RAA) information for creating virtual maps that access VSAM and sequential data, you must configure the server for RAA access.

System requirements

The following system requirement applies:

- IBM Rational Asset Analyzer for System z 6.1 PID5655-W57

Restrictions

The following restrictions and considerations apply when using RAA to access VSAM and sequential data sets:

- Virtual table creation is restricted to data sets in the RAA database that are processed by COBOL programs using JCL. Data sets accessed using CICS as well as other databases (such as IMS, CA IDMS, or Adabas) are not supported.
- Virtual table mapping is only supported through the Data Service Studio. No batch utilities or ISPF interfaces are provided to map tables.

Configuration steps

The following configuration steps are required to use RAA to access VSAM and sequential data:

1. Install virtual tables. See [“Installing virtual tables and virtual target maps for RAA access”](#) on page 60.
2. Define RAA database in the server configuration member. [“Modifying the configuration member for RAA access”](#) on page 61.
3. Activate virtual table rules. See [“Configuring virtual table rules for RAA”](#) on page 64.
4. Define credentials for target database(s). See [“Configuring authentication for RAA”](#) on page 64.

Installing virtual tables and virtual target maps for RAA access

Install virtual tables and virtual target maps for IBM Rational Asset Analyzer (RAA) access.

About this task

The Data Service Studio reads the RAA database using virtual tables and views installed as part of server set up. The following maps are distributed in XMIT format in the SAZKSAMP member AZKRAMPD.

ZRAATSPR

Virtual target system TSRAA_PROJECT1 for external subsystem named RAA1.

ZRAAT001-ZRAAT010

Virtual tables mapping tables in the RAA database. All tables use the same name as the corresponding RAA database table with a prefix of RAA_ (for example, “DMH”.DMH_DATA_RECORD” in DB2 has a virtual table name of RAA_DATA_RECORD).

ZRAAV001-ZRAAV003

Virtual views on the RAA_ virtual tables used by the Data Service Studio to read RAA data. These views are all prefixed with RAAV_ (for example, RAAV_DATASETS). All data access from the studio is performed using virtual views.

These maps are not installed by default. Use the following procedure to install these maps.

Procedure

1. Locate the AZKRAMPS member in the *hlq*.SAZKCNTL data set.

2. Modify the JCL according to the instructions provided in the AZKRAMPS member.
3. Submit the job.
The virtual tables and virtual target maps are installed.

Modifying the configuration member for RAA access

Enable and configure the parameters for IBM Rational Asset Analyzer (RAA) in the server configuration member.

About this task

The server configuration member contains a sample DATABASE definition that defines the first RAA database.

When enabling the database definition for the first RAA instance, the LOCATION and IPADDR parameters must be set to the database information for the DB2 on z/OS subsystem hosting the RAA database. The subsystem NAME(RAA1) should not be changed because a target subsystem map is configured to use this name for the virtual tables accessing the RAA database.

For multiple RAA databases, see [“Adding an RAA database”](#) on page 63.

The server member is shipped in data set member *hlq.SAZKEXEC(AZKSIN00)* and copied to *hlq.AZKS.SAZKEXEC(AZKSIN00)* by the job in the AZKGNMP1 member for you to make your local modifications.

Procedure

In the AZKSIN00 member, locate the comment “IBM Rational Asset Analyzer location”. The following example shows the section in the configuration member to locate:

```
/*-----*/
/* DRDA definition for IBM Rational Asset Analyzer location. RAA */
/* database definitions must have a NAME() starting with RAA    */
/*-----*/
"DEFINE DATABASE TYPE(ZOSDRDA) "
      "NAME(RAA1) "
      "LOCATION(DRDAZOS) "
      "DDFSTATUS(ENABLE) "
      "PORT(443) "
      "IPADDR(127.0.0.1) "
      "CCSID(37) "
      "APPLNAME(DSN1LU) "
      "IDLETIME(100) "

end
```

The following table lists the parameters for configuring support for RAA:

Parameter	Description	Valid values
TYPE	Database type. Because RAA stores information in DB2 for z/OS, this value must be ZOSDRDA.	ZOSDRDA

Parameter	Description	Valid values
NAME	The database name as known to the server. The first definition must be RAA1 because the target system map names this as the subsystem to access for RAA. For additional RAA databases, subsystems can have any name since you must also create a virtual target system to point to it; however, it recommended that the name start with RAA. <i>(Required)</i>	A valid value consists of 1 - 4 characters, starting with RAA. For example, RAA1.
LOCATION	Name of the database. The LOCATION parameter must be set to the database information for the DB2 on z/OS subsystem hosting the RAA database. <i>(Required)</i>	A valid value is a string 1 - 16 characters.
DDFSTATUS	The DDF activation status, which can be altered online by using the ISPF 4-DB2 dialog panels. <i>(Required)</i>	ENABLE Make this DDF definition active within Data Service server. DISABLE DDF endpoint is not used.
PORT	The TCP/IP port at which the server is listening. <i>(Required)</i>	A valid 1-5 numeric string.
IPADDR	Specify the dot-notation IPV4 address of the DDF endpoint. For the first RAA instance, the IPADDR parameter must be set to the database information for the DB2 on z/OS subsystem hosting the RAA database. <i>(Optional)</i>	If this parameter is not specified, the value 127.0.0.1 (local host) is the default. For group director definitions, use the DVIPA IP address of the group director.
CCSID	Specify the EBCDIC single-byte application CCSID (Coded Character Set Identifier) configured for this RDBMS subsystem on the RDBMS installation panel DSNTIPF, option 7. <i>(Optional)</i>	Refer to the RDBMS vendor documentation for a list of valid CCSIDs.

Parameter	Description	Valid values
APPLNAME	Application name. The APPLNAME used by the target endpoint for passticket generations. <i>(Optional)</i>	A valid value is 1 - 8 characters. If APPLNAME is not specified in the definition statement, no default value is provided and passticket access is disabled. Note: APPLNAME is not required when connecting from the JDBC driver.
IDLETIME	If DB2 ZPARM parameter IDTHTOIN is set to a non-zero value set IDLETIME to a value slightly less (10 secs.) than IDTHTOIN. This will also allow product DRDA threads to become inactive. <i>(DB2 for z/OS only)</i>	0-9999 seconds.

Adding an RAA database

Perform required configuration steps to add an RAA database.

About this task

For multiple RAA databases, you must perform configuration steps to define each additional RAA database. The following requirements apply when maintaining multiple RAA databases:

- For the first instance of an RAA database:
 - The database name in the must be RAA1.
 - The target system for the name RAA1 is automatically installed with the RAA maps, as described in [“Installing virtual tables and virtual target maps for RAA access” on page 60.](#)
- For subsequent RAA databases:
 - It is recommended that the database name start with RAA.
 - The target system must start with TSRAA.

Perform the following procedure for each additional RAA database.

Procedure

1. Repeat the database definition in the configuration member and make the following changes:
 - a) Change the NAME value to a unique name (for example, RAA2).
 - b) Change the LOCATION value to reference the DB2 subsystem hosting the RAA database.
 For information about the database definition parameters, see [“Modifying the configuration member for RAA access” on page 61.](#)
2. If the schema (table owner) used by RAA is not 'DMH', update the system global variable GLOBAL2.RAA.database-name.SCHEMA to the correct schema name for the RAA database tables.
3. Define a new virtual target system using the studio. The name of the virtual target system must start with TSRAA. This can be done in the Data Service Studio by selecting the **Create Virtual Target System** in the **Server** tab under the **SQL > Target Systems > DBMS** node of the tree. The connection value in each definition must match the NAME value defined in the DATABASE definition in the configuration member.
4. If required, create authentication information using the AZKDRATH batch utility.

Configuring virtual table rules for RAA

Configure Server Event Facility (SEF) rules to support multiple instances of the IBM Rational Asset Analyzer (RAA) schema using common virtual table and view definitions.

About this task

To support multiple instances of the RAA schema using common virtual table and view definitions, VTB rules AZKRAATB and AZKRAAVW provide support to process tables starting with RAA_ and views starting with RAAV_.

AZKRAATB

This table rule looks at the base view of a query for double underscores “__” and uses the data after the underscores to update the target subsystem for the query. This rule will also change the schema (or table owner) name of RAA tables from DMH to another value if the global system variable GLOBAL2.RAA.database-name.SCHEMA is set with an alternate schema name.

AZKRAAVW

This view rule looks for the double underscores and removes them from the view name to process.

With the rules activated, the Data Service Studio can suffix the view names with __SSID for all calls and process multiple instances of the RAA schema using a single set of maps.

These rules must be activated regardless of the number of RAA databases to be enabled.

Use the following procedure to set up these rules.

Procedure

Use the following steps to enable rules AZKRAATB and AZKRAAVW in the VTB ruleset:

- a) In the IBM Open Data Analytics for z/OS - Primary Option Menu, specify option E, **Rules Mgmt.**
- b) Specify option 2, **SEF Rule Management.**
- c) Enter VTB for **Display Only the Ruleset Named.**
- d) Enable the rule by specifying E and pressing Enter.
- e) Set the rules to Auto-enable by specifying A and pressing Enter.

Setting a rule to Auto-enable activates the rule automatically when the server is re-started.

Configuring authentication for RAA

Configure authentication for communicating with the IBM Rational Asset Analyzer (RAA) database.

About this task

Since RAA is hosted on a z/OS DB2 database, the z/OS credentials that are used to connect to Data Service should also be usable for the z/OS system where DB2 resides. By default, the Data Service server will attempt to use the same user ID that was presented for logon to z/OS for access to the RAA database. To use these credentials, the user ID must have SELECT access on the RAA tables in DB2.

If you choose to specify alternate credentials when communicating between the Data Service server and the RAA database, you must define what credentials to use. A facility is provided in the server to optionally change the logon credentials for a user when accessing the RAA database. To accomplish this, the following tools are provided:

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEDB2G

An ATH rule that switches credentials when connecting to an RAA database using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

You can use any of the following options for authentication:

- Use z/OS IDs for authentication
- Add a global default user definition using sample job AZKDRATH and enable ATH rule AZKEDB2G
- Add authentication information for specific mainframe users using sample job AZKDRATH and enable ATH rule AZKEDB2G

If z/OS user IDs and passwords used to connect to the Data Service server are not authorized for the Db2 database hosting the RAA tables, you must define the credentials to use. Use the following procedure.

Procedure

1. Use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq.SAZKCNTL* data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.
 When adding the SYSIN statements that define the alternate credentials for logging in to your RAA database, as instructed in the JCL, make sure to specify the correct DBTYPE. For RAA databases, specify DBTYPE=ZOSDRDA.
 - c) Submit the job.
 - d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
2. Auto-enable the SEF ATH rule SAZKXATH(AZKEDB2G) to switch credentials when connecting to RAA using DRDA. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Set Auto-Enable for the AZKEDB2G rule member by entering A and pressing Enter.

Using Db2 for z/OS to access multiple data source types

You can use a Db2 for z/OS subsystem to access all Data Service data, regardless of data source type.

Db2 Virtualization (DB2V) is a feature that provides single-point access to various data source types. Using a Db2 for z/OS subsystem as a primary access point to your data from various data source types provides the following benefits:

- Db2 can be used to standardize all federated access.
- Db2 security can be used to manage access to Data Service federated data.
- No additional drivers are required on platforms needing federated access.

To use a Db2 subsystem as a primary access point requires the following components:

- Virtualized data for each data source on the Data Service server.
- A Db2 subsystem, local to the Data Service server, which will be used as the primary access point to all virtualized data, regardless of data source type.
- A user-defined table function (UDTF) defined in the Db2 subsystem for each virtual table.
- (Optional) A view defined in the Db2 subsystem for each virtual table. You can use the view (which calls the UDTF) to query the data in the Data Service server. The view provides a convenient and familiar way to access the data instead of using the table function directly in SQL statements. Using the view, the virtualized data will appear as though it is local to the Db2 subsystem in which the view is defined.

System requirements

The following system requirements apply:

- Db2 for z/OS 11.1.0 (PID 5615-DB2) or newer

Restrictions

The following restrictions and considerations apply:

- The Db2 subsystem must be running on the same LPAR as the Data Service server.
- Data access using Db2 table functions is query only.

Configuration steps

To configure your system to use a Db2 subsystem as a primary access point, complete the following steps:

1. Configure access to data from various data source types. See the various sections in [Chapter 1, “The Data Service SQL solution,”](#) on page 1.
2. Create virtual tables for data sources. See "Creating virtual tables" in the *User's Guide*.
3. Designate the Db2 subsystem to use as the primary access point, and install the UDTF routines. See [Install the UDTF routines in Db2](#).
4. Create a UDTF and a view in the designated Db2 subsystem for each virtual table using the Data Service studio. See "Creating Db2 user-defined table functions" in the *User's Guide*.

After these steps are completed, you can use the view or UDTF to query your data.

Install the UDTF routines in Db2

Support for accessing Data Service tables using views in Db2 is implemented as a generic user-defined table function (UDTF) in Db2. The generic program is an IBM LE program compiled for z/OS ARCH levels 9-12. The name is in the form AZKUDTaN, where *a* is the z/OS architecture level (9, A, B, C).

The UDTF programs run in a Db2 WLM address space that is started when the UDTF table functions are called. These routines must be included in the STEPLIB of the Db2 WLM address space JCL so they are available when requested. The Db2 system administrator can either add the Data Service load library to the JCL started for Db2 WLM address spaces, or copy the following members from the SAKLOAD library into an existing STEPLIB allocated to Db2 (typically the SDSNEXIT library):

- AZKUDT9N
- AZKUDTAN
- AZKUDTBN
- AZKUDTCN
- AZKSXMAP
- AZKCLAPI
- AZKCMCP

SQLSTATE values issued by AZKUDTaN

SQLSTATE	Description
38601	The SCRATCHPAD size defined in the CREATE FUNCTION for the UDTF is too small. The corresponding message includes the minimum size needed by the UDTF.
38602	The format of the period delimited DVMNAME parameter is invalid. Generally, this means there are not enough period delimited tokens in the DVMNAME or one of the tokens exceeds the maximum size for its intended use.
38603	One of the runtime options is unrecognized or invalid. The option or value in question is included in the associated error message.

SQLSTATE	Description
38604	A call sequence error occurred. Either an OPEN request was made without initializing the environment or a FETCH request was made when there was no active query. This may indicate a program logic error in the UDTF requiring IBM Software Support assistance.
38605	A memory allocation error occurred in the UDTF program. The accompanying message indicates the amount of memory requested. If the memory requested was 256K or less, the region size of the WLM address space needs to be increased.
38606	This SQLSTATE indicates the XMPL control block was not returned in a high level DS Client API call. This error is extremely unlikely and requires IBM Software Support to diagnose.
38607	This SQLSTATE indicates that a connection request to the Data Service server failed. The accompanying message includes the name of the server taken from the DVMNAME parameter. If the name is incorrect, fix the name in the DB2 view or UDTF call. If the name is correct, check that the Data Service server is running on the same LPAR as the Db2 subsystem and that the IN00 for the Data Service server has the DSCIENTACTIVE system parameter set to YES.
38608	This SQLSTATE indicates a send request to Data Service with an SQL statement failed. Review the associated message and the Data Service server trace for diagnosing the cause of the error.
38609	This SQLSTATE indicates a receive request to Data Service for result set rows failed. Review the associated message and the Data Service server trace for diagnosing the cause of the error.
38610	This SQLSTATE indicates that the number of SQLVARS in the SQLDA returned from Data Service does not match the number of columns passed in the COLINFO parameter. This will usually indicate that the virtual table in Data Service has changed since the UDTF definition was created in Db2. Review the table columns in the UDTF and verify they match the columns defined in the corresponding Data Service virtual table.

Chapter 2. Spark SQL data access: JDBC

The JDBC driver is a Type 4 driver (written in Java) that is used to implement the network protocol for Mainframe Data Service.

The Java Virtual Machine manages the applications connection to Data Service. Java-based applications and tools use the JDBC driver to access Data Service applications and features. Clients connect directly to Data Service without translation.

Requirements

The driver requires Java 1.7 or higher and it is supplied as a .jar archive file. The components that following are included in the archive file.

The following runtime .jar files are required:

- `:dv-jdbc-[version #].jar`: The driver core implementation file.
- `log4j-api-[version #].jar`: The logging framework API file.
- `log4j-core-[version #].jar`: The logging framework implementation file.

The following sample logging configuration file is included:

- `log4j2.xml`: A sample logging configuration file.

The following command line utilities are included:

Note: Some system-specific environment utilities are available in two formats; Microsoft Windows command script (.cmd) and Bash shell script (.sh).

- `hashpassword.cmd`: Use this utility to generate a hashed format of your text password. You can include the hashed password in your applications connection string or .ini file. The script prompts the user for the plain text password. To avoid prompting, the password can be specified as a command-line argument.
- `helpdriver`: Displays the help text, including a detailed list of all supported driver properties.
- `sysinfo.cmd`: Displays local system information.
- `LICENSE.txt`: Provides product licensing information.
- `NOTICE.txt`: Lists notices related to this product.
- `RELEASE-NOTES.txt`: Provides details about optional folders.
- `optional-charsets` folder: Contains additional character sets. Depending on your set up you may need to place these jars either on you classpath or endorsed classpath.
- Java API documentation
- Optional runtime.jar
- `optional-charsets`: Depending on your environment, you may need to place these .jars on either your classpath or endorsed classpath.
- `optional-pooling` folder: Contains jar files to pool Data Service data sources. To pool a data source, use:

```
import org.apache.commons.dbcp2.cpdsadapter.DriverAdapterCPDS;
DriverAdapterCPDS driverAdapterCPDS = new DriverAdapterCPDS();
driverAdapterCPDS.setUrl("jdbc:rs:dv:HOST=...");
```

- `optional-logging` folder: Contains optional Log4j jar files for adding additional logging features.

Connecting to a data source using JDBC

A JDBC connection string is used to load the driver and to indicate the settings that are required to establish a connection to the data source. These settings are referred to as *connection properties*. The connection string is a URL with the following format:

```
jdbc:rs:dv://host:port;Key=Value;Key=value;...
```

For example, you can use the following connection string to access virtual tables on your Data Service server, where *host* is the network hostname or IP address:

```
jdbc:rs:dv://host:1200;DatabaseType=DVS;user=userid;password=xxxx
```

If you prefer not to use the `//host:port` syntax, you can use the following string:

```
jdbc:rs:dv:Host=host;Port=1200;DatabaseType=DVS;user=userid;password=xxxx
```

Additional connection properties can be added to influence the behavior of the driver-server communication.

Coding a Spark application

A Spark application can access a data source using the Spark SQL interface, which is defined in the `org.apache.spark.sql` package namespace. The `DataFrameReader` interface, obtained via `SparkSession.read`, is used to load data sets from the data source. Spark SQL uses the JDBC driver to connect to the Data Service server and access the virtualized data.

Sample Scala code fragment:

```
val url = "jdbc:rs:dv://host:1200; DatabaseType=DVS; user=userid; password=xxx"
val sql = "MY_VIRTUAL_TABLE"
val spark: SparkSession = ...
val df = spark.read
    .format("jdbc")
    .option("url", url)
    .option("dbtable", sql)
    .load()
// perform data analytics on the dataframe
```

Running on the same sysplex

Applications executing on the same sysplex as the server are not required to specify logon credentials when invoking the JDBC driver. Instead, the application can specify empty string settings for the user and password to indicate that the current application user ID is to be used. For example, empty values can be specified in the JDBC connection string as follows:

```
...; user=;password=; ...
```

When values are specified, the credentials supplied on the driver connection string will be used for server logon instead of the propagated application user ID information.

JDBC connection properties

The JDBC driver supports the following connection properties. If the same property occurs more than once in the connection string, the last entry takes precedence. Property names are not case sensitive.

Table 6: JDBC connection properties	
Property names	Description
AdabasColumnNameCorrelationIds Alias: ABCN	Support ADABAS column name correlation IDs. Required: false Default value: false Valid values: [true, false]
ApplicationName Alias: APNA	Application name that is sent to the host as part of logon for connection tracking. Required: false (maximum of 16 characters in length)
AuthenticationMechanism Alias: ATHM	Mechanism for encrypting passwords. Required: false Default value: DEFAULT Valid values: [DEFAULT, AES]
BindInetSocketAddressList Alias: BISAL	A single local name or IP address to bind a socket, or a comma-separated IP address list (used with MapReduce), or AutoDetect to detect NIC IP addresses automatically. Required: false Default value: No
CatalogPrefix Alias: CPMX	Database catalog prefix, SYSPROC for DB2, SLENGL for DVS, SDBMAP for the others. Required: false Default value: SYSPROC Valid values: [SYSIBM, SYSPROC, SDBMAP, SLENGL]
CertificateHostName Alias: HostNameInCertificate	Host name for certificate validation when SSL encryption and validation is enabled. Required: false

Table 6: JDBC connection properties (continued)

Property names	Description
Charset Alias: CS, CodePage, CP, Encoding, ENC	The database character encoding. To get a complete list the charsets that are available on a particular JVM, call the <code>Charset.availableCharsets()</code> API, where <code>charsetName</code> is one of the available Java character sets. The list of available character sets that are returned depends on the specific version and supplier of Java, as well as the availability of the ICU jar files on the classpath. Names with the 'x-' prefix indicate that a charset is not registered at the Internet Assigned Numbers Authority (IANA). For more information, see Class Charset (docs.oracle.com/javase/8/docs/api/java/nio/charset/Charset.html) and ICU Converter Explorer (ssl.icu-project.org/icu-bin/convexp). Use the IBM JRE for proper data translation when <code>Charset</code> specifies a Japanese Code page such as; 930, 939, 1390, 1399 or 5026. IBM JRE 1.8 or 1.7 is recommended when accessing IBM-1390 and IBM-1399 mainframe data, and using or exchanging that data in a Unicode environment. IBM JRE 1.8 is recommended when accessing IBM-930 and IBM-939 mainframe data, if you do not experience issues using the IBM-conversion-table-based conversion of the following EBCDIC characters: • X'4260' (Minus sign) • X'444A' (EM Dash) • X'43A1' (Wave Dash) • X'447C' (Double vertical line) • X'426A' (Broken bar) Required: false Default value: IBM037 Valid values: For a list of supported character sets, see "Character sets" on page 84.
CicsTransactionName Alias: TRNA	CICS transaction name. Required: false Default value: (maximum of eight characters in length)
CompressionLevel	Compression level (ZLib only), -1 (default), 1 (best speed) ... 9 (best compression). Required: false Default value: -1 Valid values: [-1, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Table 6: JDBC connection properties (continued)

Property names	Description
CompressionThresholdBytes	Compression threshold in bytes. The driver compresses data for buffers larger than this size. This value can be post-fixed with a unit like KB (K) or MB (M). For example, the following values are all equal: 1048576, 1024 K, 1024 KB, 1 MB. Required: false Default value: 0
CompressionType	The type of compression used. Required: false Default value: UNCOMPRESSED Valid values: [ZLIB_NO_WRAP, CMBU, CMBV, UNCOMPRESSED]
ConnectionType	Connection wire type. Required: false Default value: SOCKET Valid values: [SOCKET, SOCKET_CHANNEL]
CountTraceEnter	Counts JDBC API calls. Required: false Default value: false Valid values: [true, false]
DatabaseRequestModule Alias: DBRM	Database request module. Required: false Default value: OPRXSQ (maximum of eight characters in length)
DatabaseType Alias: DBTY	The database type to connect to after connecting to the server. Required: false Default value: DRDAorDB2 Valid values: • DRDAorDB2 (the data source type is DB2 on a z/OS subsystem or DB2 LUW database) • DVS (the data source type is determined by the virtual table map definition, and no value is specified for SUBSYS)

Table 6: JDBC connection properties (continued)

Property names	Description
EncodeUserName	When set to true, the user name is encoded when establishing the server connection. Support of this feature can be controlled using server parameters USERIDENCODEREQUIRE and USERIDENCODEALLOW. Required: false Default value: false Valid values: [true, false]
EncryptionMethod	Encryption method. Required: false Default value: NONE Valid values: [NOENCRYPTION, NONE, SSL]
GetTablesSchemaFilter Alias: DP	Filter to use for getTables() DB2 metadata schema. Required: false
HexDumpBytesPerLine	Bytes per line in the hexadecimal dump. Required: false Default value: 16
HexDumpBytesPerWord	Bytes per word in the hexadecimal dump. Required: false Default value: 4
Host	Host name or IP address. Required: true
IniFile Alias: INI	Loads properties from this INI file (overrides IniFileEnvVar). Required: false.
IniFileCharset Alias: INICS	The INI file Charset. Required: false Default value: UTF-8.
IniFileDataSourceName Alias: DSN	Loads properties from a section name to an INI file. Required: false.
IniFileEnvVar Alias: INIEV	Loads properties from the INI file to this environment variable. Required: false. Default value: DV_INI
InitialCurrentDegree Alias: SEDG	Initial current degree (DB2). Required: false Valid values: [ANY, 1]

Table 6: JDBC connection properties (continued)	
Property names	Description
InitialCurrentPackageSet SEPK	Initial current package set (DB2). Required: false
InitialCurrentRules Alias: SERL	Initial current rules (DB2). Required: false Valid values: [DB2, STD]
InitializationString	Initialization string, use ';' to separate statements, wrap the string in '(' and ')' when used in a connection string. Required: false
KeyPassword Alias: KP	The SSL key password. Required: false
KeyStore Alias: KS	The SSL keystore. Required: false
KeyStorePassword Alias: KSP	The SSL keystore password. Required: false
LegacySqlPrepareEnabled Alias: WRPR	This flag controls the behavior of SQLPrepare for non-DB2 data sources such as ADABAS, VSAM, IMSDB, and VSAM CICS. When this keyword is set to true (which is the default), a request is always sent to the host at SQLPrepare time to obtain metadata for the SQL statement. For applications that access non-DB2 data sources and do NOT require metadata after the SQLPrepare, it is recommended to set WRPR to false for better performance since this will eliminate a network roundtrip whenever a SQLPrepare is executed. Required: false Default value: true Valid values: [true, false]

Table 6: JDBC connection properties (continued)

Property names	Description
LGID Alias: LanguageID	This setting is for backward compatibility. It is recommended to use Charset instead. Each language code corresponds to a charset used for byte conversion. If a key has no value, the driver will use the default. The mappings are as follows: <pre>{ARB=IBM420, CHS=, CHT=, DAN=IBM01142, DEU=IBM01141, DFT=IBM037, ENC=IBM1047, ENG=IBM285, ENU=IBM037, ESN=IBM01145, ESP=IBM284, FIN=IBM01143, FRA=IBM01147, FRC=IBM037, ISL=IBM01149, ITA=IBM01144, JNL=IBM1390, JNX=IBM1399, JPL=IBM5026, JPX=IBM5035, KOR=IBM037, KRN=x-IBM833, MDI=, NGN=IBM01142, NLD=IBM037, NOR=IBM01142, PPS=, PTG=IBM037, SVE=IBM278, SWE=IBM01143, TUR=IBM1026}</pre> Required: false Default value: ENU Valid values (31): [ARB, CHS, CHT, DAN, DEU, DFT, ENC, ENG, ENU, ESN, ESP, FIN, FRA, FRC, ISL, ITA, JNL, JNX, JPL, JPX, KOR, KRN, MDI, NGN, NLD, NOR, PPS, PTG, SVE, SWE, TUR]
LogConfiguration Alias: LOGCONFIG	Sets the Log4j 2 configuration file. Required: false
LogThreadsState Alias: LTS	Logs the state of a thread. Required: false Default value: false Valid values: [true, false]
LogThreadsStatePeriodMillis	Logs the state period for a thread in milliseconds. Required: false Default value: 1000
LoginTimeoutMillis	Login timeout in milliseconds (0 = system default if there is one, or no limit.) Required: false Default value: 0
LoginTimeoutSeconds Alias: LOGINTIMEOUT	Login timeout in seconds. Required: false Default value: 0

Table 6: JDBC connection properties (continued)

Property names	Description
MapReduceClient Alias: MRC, MapReduce	Use MapReduceClient (MRC) to read query results in parallel from different connections. When enabled, the driver creates one master connection and N worker connections (instances of JDBC connections). Required: false Default value: false Valid values: [true, false, list] To distribute MapReduce on a single server, select from the following methods: • Set MapReduceClient to true (MRC=true). The MapReduceConnectionCount defaults to the number of CPU cores discovered (for example: MRCC=10) • Set MRC=(host, port, taskCount) To distribute MapReduceClient over multiple servers, set MapReduceClient: MCR=(host1, port1, taskCount1), (host2, port2, taskCount2),... If you are using MapReduceClient with RDBMS or IMS, you must complete the metadata repository configuration requirements. See "MapReduce" in the <i>Administrator's Guide</i>.
MapReduceClientCount Alias: MRCC	The single-connection MapReduceClientCount. This value is only set for use with MRCN. The MRCC default setting is 0. Required: false Default value: 0
MapReduceClientNumber Alias: MRCN	To enable highly-parallel client applications to control concurrent MapReduce connections, from which queries are executed as a single thread, set the MapReduceClientCount (MRCN) value for each MapReduceClientCount (MRCC). The following example shows a connection string used to connect to the first of four available connections: <pre>jdbc:rs:dv://host:port;DBTY=DVS; PWD=xxx;UID=xxx; MXBU=4MB; MapReduceClientCount=4; MapReduceClientNumber=1</pre> To disable this feature, set MapReduceClientCount to 0 (MRCC=0) and set MapReduceClient to false (MRC=false).
MapReduceFillValueMaximumInitialSize Alias: MRFVMIS	This value specifies the initial capacity of the result row pre-fetch cache for a given buffer that is used after the MapReduce or Parallel IO read queue exceeds the value set for MapReduceFillValueThreshold. Required: false Default value: 20,000 (rows per buffer)

Table 6: JDBC connection properties (continued)

Property names	Description
MapReduceFillValueThreshold Alias: MRFVT	When using Parallel IO or MapReduce, this value specifies the number of buffers on the parallel IO thread that must be exceeded before pre-filling the column values (converting mainframe row bytes to Java object representations). Required: false Default value: -1 (off).
MapReducePollTimeOutMillis Alias: MRPTO	The MapReduce and Parallel IO inter-thread poll timeout, in milliseconds. Required: false Default value: 50
MapReduceQueueStats Alias: MRQS	Gathers statistics for MapReduce result sets. Required: false Default value: false Valid values: [true, false]
MaximumBufferSize Alias: MXBU	Maximum server-side communication buffer size in bytes. This value can be post-fixed with a unit like KB (K) or MB (M). For example, the following values are all equal: 1048576, 1024 K, 1024 KB, 1 MB. The minimum value is 40960. Required: false Default value: 262144
MaximumFieldSize Aliases: MFS, MaxFieldSize	Maximum field size to return. Required: false Default value: 2147483647
MaximumRows Alias: MR	Maximum number of rows to return. Required: false Default value: 0
MetaDataCharset Alias: MDCS	The database metadata character encoding. Available encodings depends on the runtime. See the Charset property. Required: false Default value: IBM037
NetworkTimeoutMillis Alias: NTOM	Network timeout in milliseconds. Required: false Default value: 0

Table 6: JDBC connection properties (continued)	
Property names	Description
NetworkTimeoutSeconds Alias: NETWORKTIMEOUT	Network timeout in seconds. Required: false Default value: 0
ParrallelIoBufferCount Alias: PIOBC	When the value of the Parallel IO buffer count is > 0 (where the buffer size = MXBU), a background thread reads the rows from the server as the main thread consumes them. This memory is re-usable. For example, if MXBU is set to 4 MB and PIOBC is set to 10, the driver uses 40 MB of memory as the read-ahead buffer (10 x 4 MB buffers). Required: false. Default value: 0.
Password Alias: PWD, PassPhrase	Password or password phrase. Required: false
PasswordToUpperCase Alias: UCLC	Converts passwords to uppercase. This property is ignored when a password phrase has been specified. Required: false Default value: true Valid values: [true, false]
Plan	The DB2 plan name used for a DB2 connection. This is used when the subsystem is set to a valid DB2 subsystem name. Required: false Default value: SDBC1010 (maximum of eight characters in length)
Port	Server port. Required: false Default value: 1200
PrepareMetadataSQL	Determines how the SQL statement metadata calls are handled. Required: false Default value: true Valid values: [true, false] If set to true, the driver prepares the SQL statements for metadata calls. If set to false, the driver builds and executes the SQL statements for metadata calls dynamically, which can be susceptible to SQL injection attacks.

Table 6: JDBC connection properties (continued)	
Property names	Description
QueryTimeoutMillis Alias: QTOM	Query timeout in milliseconds. Required: false Default value: 0
QueryTimeoutSeconds Alias: QUERYTIMEOUT	Query timeout in seconds. Required: false Default value: 0
SelectForReadOnly Alias: RO	Appends FOR FETCH ONLY to JDBC read-only SQL query statements. Required: false Default value: true Valid values: [true, false]
ServerCertificateStrategy	Server certificate strategy. Required: false Default value: Validate Valid values: [Validate, AcceptSelfSigned, Trust]
SocketKeepAlive Alias: SKA	Socket keep alive. Required: false Default value: false Valid values: [true, false]
SocketReceiveBufferSize Alias: SRBS	Socket receive buffer size hint in bytes. This value can be post-fixed with a unit like KB (K) or MB (M). For example, the following values are all equal: 1048576, 1024 K, 1024 KB, 1 MB. Required: false Default value: 0
SocketSendBufferSize Alias: SSBS	Socket send buffer size hint in bytes. This value can be post-fixed with a unit like KB (K) or MB (M). For example, the following values are all equal: 1048576, 1024 K, 1024 KB, 1 MB. Required: false Default value: 0
SocketTcpNoDelay Alias: STCPND	Socket TCP NoDelay. Required: false Default value: false Valid values: [true, false]

Table 6: JDBC connection properties (continued)	
Property names	Description
SqlAuthorizationId Alias: ALUS	SQL authorization ID. Required: false
SslContextProtocol Alias: SSLPROTOCOL	SSL context protocol. Required: false Default value: TLS
StrictJdbcCompliance	Indicates whether the driver complies strictly with the JDBC spec. Required: false Default value: false Valid values: [true, false]
SubSystem Alias: SUBSYS	The database subsystem name. If the data source is DB2 (DatabaseType=DRDAorDB2), enter the DB2 subsystem name as it exists on the mainframe. If the data source is a non-DB2 database (where TYPE must be Member, Group, or LUW), enter the name as it exists in the NAME field of the DEFINE entry in the server configuration member, AZKSIN00. If the data source is DVS, the subsystem name should be set to NONE. Required: false Default value: NONE Valid values: 1 - 4 alphanumeric characters
ThrowUnsupportedAll	[Development tool] Throws an exception when an API is not supported. Required: false Default value: true Valid values: [true, false]

Table 6: JDBC connection properties (continued)

Property names	Description
TraceBrowseAppender	To collect JDBC driver server trace log information, add the name of the collection appender to the log4j configuration file. The following example shows the collection name tag: <pre><Appenders> ... other appenders here ... <Collection name="TB"> <PatternLayout> <Pattern>%d %-5level [%t][%logger] %msg%n%throwable</Pattern> </PatternLayout> </Collection> </Appenders> <Loggers> <Root level="..."> ... other appenders here ... <AppenderRef ref="TB" /> </Root></pre> Where the name of the collection appender (TB in this example) is the name you choose. Add the TraceBrowseAppender setting to the JDBC connect string to provide the collection appender name in the log4j2.xml config file to the server. For example, TraceBrowseAppender=TB. The server parameter TRACE EXTERNAL TRACE DATA (TRACEEXTERNTRACEDATA) must also be enabled. Required: false
TruncateCallLiteral Alias: TRLT	Truncate CALL string. Required: false Default value: true Valid values: [true, false]
TrustStore	The SSL trust store. Required: false
TrustStorePassword	The SSL trust store password. Required: false
UpperCaseAllCharacters	This field controls if all character data sent to the host should be converted to upper case or not. If this field is set to true, then all character data will be converted to upper case. If this field is set to false, then character data will not be converted to upper case. Required: false Default value: false Valid values: [true, false]

Table 6: JDBC connection properties (continued)	
Property names	Description
UpperCaseNonLiterals	This flag controls if all non-literal values in SQL statements passed to the driver should be converted to upper case or not. If this flag is set to true, then strings not in single or double quotes will be converted to upper case. Required: false Default value: true Valid values: [true, false]
URL	Use to specify connection properties. Required: false
user Alias: UID	The user name or identifier.
UserParm Alias: UserParm	Sent in the logon information to complete logon to a host security system and/or database. Maximum of 100 characters in length.
ValidateServerCertificate	Validate server certificate or not. Required: false Default value: true Valid values: [true, false]
VpdGroupMemberCount	Virtual Parallel Data group member count. Required: false Default value: 0-255
VpdGroupName	Virtual Parallel Data group name. Maximum of eight characters in length. Required: false Default value:
VpdGroupOpenTimeoutSeconds	Virtual Parallel Data group open timeout in seconds. Required: false Default value: 0
VpdIoThreadCount	Virtual Parallel Data I/O thread count. Required: false Default value: 0-255
XaEnabled Alias: XAEN	Set to true to enable XA transactions. Required: false Default value: false Valid values: [true, false]

Table 6: JDBC connection properties (continued)	
Property names	Description
XaTransactionManager Alias: XAOP	The type of transaction manager used for XA operations. Required: false Valid value: JTS

Character sets

The list of available character sets that are returned depends on the specific version and supplier of Java, as well as the availability of the ICU jar files on the classpath.

The following Charsets (CS) are supported:

Charsets
Adobe-Standard-Encoding, Big5, Big5-HKSCS, BOCU-1, CESU-8, cp1363, cp851, EUC-JP, EUC-KR, GB18030, GB2312, GB_2312-80, GBK, hp-roman8, HZ-GB-2312, IBM-Thai, IBM00858, IBM01140, IBM01141, IBM01142, IBM01143, IBM01144, IBM01145, IBM01146, IBM01147, IBM01148, IBM01149, IBM037, IBM1026, IBM1047, IBM273, IBM277, IBM278, IBM280, IBM284, IBM285, IBM290, IBM297, IBM420, IBM424, IBM437, IBM500, IBM775, IBM850, IBM852, IBM855, IBM857, IBM860, IBM861, IBM862, IBM863, IBM864, IBM865, IBM866, IBM868, IBM869, IBM870, IBM871, IBM918, ISO-2022-CN, ISO-2022-CN- EXT, ISO-2022-JP, ISO-2022-JP-1, ISO-2022-JP-2, ISO-2022-KR, ISO-8859-1, ISO-8859-10, ISO-8859-13, ISO-8859-14, ISO-8859-15, ISO-8859-2, ISO-8859-3, ISO-8859-4, ISO-8859-5, ISO-8859-6, ISO-8859-7, ISO-8859-8, ISO-8859-9, JIS_X0201, JIS_X0212-1990, KOI8-R, KOI8-U, KSC_5601, macintosh, SCSU, Shift_JIS, TIS-620, US-ASCII, UTF-16, UTF-16BE, UTF-16LE, UTF-32, UTF-32BE, UTF-32LE, UTF-7, UTF-8, windows-1250, windows-1251, windows-1252, windows-1253, windows-1254, windows-1255, windows-1256, windows-1257, windows-1258, windows-31j, x-Big5-HKSCS-2001, x-Big5-Solaris, x-compound-text, x-ebcdic-xml-us, x-euc-jp- linux, x-EUC-TW, x-euc-tw-2014, x-eucJP-Open, x-ibm-1047-s390, x- ibm-1125_P100-1997, x-ibm-1129_P100-1997, x-ibm-1130_P100-1997, x- ibm-1131_P100-1997, x-ibm-1132_P100-1998, x-ibm-1133_P100-1997, x- ibm-1137_P100-1999, x-ibm-1140-s390, x-ibm-1141-s390, x-ibm-1142-s390, x-ibm-1143- s390, x-ibm-1144-s390, x-ibm-1145-s390, x-ibm-1146-s390, x-ibm-1147-s390, x- ibm-1148-s390, x-ibm-1149-s390, x-ibm-1153-s390, x-ibm-1154_P100-1999, x- ibm-1155_P100-1999, x-ibm-1156_P100-1999, x-ibm-1157_P100-1999, x- ibm-1158_P100-1999, x-ibm-1160_P100-1999, x-ibm-1162_P100-1999, x-

Charsets
ibm-1164_P100-1999, x-ibm-1250_P100-1995, x-ibm-1251_P100-1995, x-ibm-1252_P100-2000, x-ibm-1253_P100-1995, x-ibm-1254_P100-1995, x-ibm-1255_P100-1995, x-ibm-1256_P110-1997, x-ibm-1257_P100-1995, x-ibm-1258_P100-1997, x-ibm-12712-s390, x-ibm-12712_P100-1998, x-ibm-1373_P100-2002, x-ibm-1386_P100-2001, x-ibm-16684_P110-2003, x-ibm-16804-s390, x-ibm-16804_X110-1999, x-ibm-25546, x-ibm-33722_P12A_P12A-2009_U2, x-ibm-37-s390, x-ibm-4517_P100-2005, x-ibm-4899_P100-1998, x-ibm-4909_P100-1999, x-ibm-4971_P100-1999, x-ibm-5123_P100-1999, x-ibm-5351_P100-1998, x-ibm-5352_P100-1998, x-ibm-5353_P100-1998, x-ibm-803_P100-1999, x-ibm-813_P100-1995, x-ibm-8482_P100-1999, x-ibm-901_P100-1999, x-ibm-902_P100-1999, x-ibm-9067_X100-2005, x-ibm-916_P100-1995, x-IBM1006, x-IBM1025, x-IBM1046, x-IBM1097, x-IBM1098, x-IBM1112, x-IBM1122, x-IBM1123, x-IBM1124, x-IBM1153, x-IBM1363, x-IBM1364, x-IBM1371, x-IBM1381, x-IBM1383, x-IBM1388, x-IBM1390, x-IBM1399, x-IBM300, x-IBM33722, x-IBM720, x-IBM737, x-IBM833, x-IBM834, x-IBM856, x-IBM867, x-IBM874, x-IBM875, x-IBM921, x-IBM922, x-IBM930, x-IBM933, x-IBM935, x-IBM937, x-IBM939, x-IBM942, x-IBM942C, x-IBM943, x-IBM943C, x-IBM948, x-IBM949, x-IBM949C, x-IBM950, x-IBM954, x-IBM964, x-IBM970, x-IBM971, x-IMAP-mailbox-name, x-iscii-be, x-iscii-gu, x-iscii-ka, x-iscii-ma, x-iscii-or, x-iscii-pa, x-iscii-ta, x-iscii-te, x-ISCII91, x-ISO-2022-CN-CNS, x-ISO-2022-CN-GB, x-iso-8859-11, x-JIS0208, x-JIS7, x-JIS8, x-JISAUTOdetect, x-Johab, x-LMBCS-1, x-mac-centraleurroman, x-mac-cyrillic, x-mac-greek, x-mac-turkish, x-MacArabic, x-MacCentralEurope, x-MacCroatian, x-MacCyrillic, x-MacDingbat, x-MacGreek, x-MacHebrew, x-MacIceland, x-MacRoman, x-MacRomania, x-MacSymbol, x-MacThai, x-MacTurkish, x-MacUkraine, x-MS932_0213, x-MS950-HKSCS, x-MS950-HKSCS-XP, x-mswin-936, x-PCK, x-SJIS_0213, x-UnicodeBig, x-UTF-16LE-BOM, X-UTF-32BE-BOM, X-UTF-32LE-BOM, x-UTF16_OppositeEndian, x-UTF16_PlatformEndian, x-UTF32_OppositeEndian, x-UTF32_PlatformEndian, x-windows-50220, x-windows-50221, x-windows-874, x-windows-949, x-windows-950, x-windows-iso2022jp]

Error handling

The JDBC driver reports errors to an application by throwing SQL exceptions, each of which contains the following details.

- Description of the likely cause of the error, prefixed by the component that generated the error.
- Native error code, if applicable.

All errors, including those generated by the driver framework or the driver layer, have the following format:

```
[DV][JDBC Driver][version #]message
```

To fully interpret the error message, it may be necessary to refer to the JDBC documentation for possible resolutions, or refer to the database management system documentation to interpret error codes.

Debugging and tracing

Enable JDBC driver logging in the Apache Log4j configuration file.

About this task

The JDBC driver implements logging by using Apache Log4j2 (<https://logging.apache.org/log4j/2.x/>).

Note: To include JDBC driver logging information in the Studio **Server Trace** pane, enable the JDBC TraceBrowseAppender property.

Procedure

Open the Log4j configuration file (<https://logging.apache.org/log4j/2.x/manual/configuration.html>) and choose one of the following methods to implement logging:

- Add the directory and folder location to the CLASSPATH
- Set the LogConfiguration connection property to a file path or URI that points to a Log4j configuration file

The following is a sample log4j2.xml configuration file that is included in the JDBC driver installation ZIP file:

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN" monitorInterval="5">
  <Appenders>
    <Console name="Console" ...>
      <PatternLayout ... />
    </Console>
    <RollingRandomAccessFile name="RollingFile" ...>
      <PatternLayout>
        <Pattern>...</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy ... />
      </Policies>
    </RollingRandomAccessFile>
  </Appenders>
  <Loggers>
    <Root level="WARN">
      <AppenderRef ref="Console" />
      <AppenderRef ref="RollingFile" />
    </Root>
    <Logger name="com.rs.jdbc.dv.DvDataSource"
      level="WARN" />
    <Logger name="com.rs.jdbc.dv.log.DvThreadsStateLogger"
      level="WARN" />
    <Logger name="javax.management"
      level="WARN" />
    <Logger name="sun.rmi"
      level="WARN" />
    <Logger name="com.rs.dv.util.LoggerEnvironmentReporter"
      level="OFF" />
  </Loggers>
</Configuration>
```

The sample configuration file does not log anything if the status of the entire configuration and level of the root logger are set to WARN (<Configuration status=="WARN">;<Root level="WARN">). Setting the configuration's level to TRACE causes Log4j to log about itself being configured; this is useful when debugging the configuration file is necessary. Setting the root logger level to DEBUG causes the driver to log more detailed information, and setting the level to TRACE will allow the tracing of JDBC API entries and exits. For more information, refer to the following URL:

<https://logging.apache.org/log4j/2.x/manual/flowtracing.html>

Each driver implementation class has a logger. The logger name is derived from the fully qualified class name. The logging levels for each of these loggers may be changed independently.

By default, setting the root level to TRACE results in logging hexadecimal dumps of the buffers. To enable API logging while disabling buffer tracing, use a Log4j filter.

The following example shows how to define the filter DV.BUFFER to the JDBC configuration file:

```
<Loggers>
...
  <Filters>
    <MarkerFilter marker="DV.BUFFER"
      onMatch="DENY" onMismatch="NEUTRAL" />
  </Filters>
</Loggers>
```

Filtering may also be applied to the communication and SQL operations by setting appropriate values for the marker attribute. You may choose from the following filters:

- DV.BUFFER
 - DV.BUFFER.HEADER
 - DV.BUFFER.PARAM
 - DV.BUFFER.COMPRESSED
 - DV.BUFFER.DECOMPRESSED
 - DV.BUFFER.UNCOMPRESSED
 - DV.BUFFER.DESCRPTION
 - DV.BUFFER.DESCRPTION.CONNECT
- DV.COMM
 - DV.COMM.SOCKET
- DV.SQL
 - DV.SQL.PREPARE.USER
 - DV.SQL.BATCH.USER
 - DV.SQL.BATCH.SERVER
 - DV.SQL.QUERY.USER
 - DV.SQL.QUERY.SERVER
 - DV.SQL.QUERY.POST
 - DV.SQL.UPDATE.USER
 - DV.SQL.UPDATE.SERVER

Connecting to a DRDA database server

To connect to a DRDA (DB2 and Oracle) database server, include the application information on the system CLASSPATH.

About this task

To connect to the database, you need add the CLASSPATH to your system. In your application, you need to define the connection string and the connection.

Procedure

1. Add the CLASSPATH.

To load the JDBC driver, the Java Virtual Machine requires that the location and name of the driver and Log4j implementation files be included in the system CLASSPATH. For example:

```
install_dir/lib/dv-jdbc-[version #].jar
install_dir/lib/log4j-api-[version #].jar
install_dir/lib/log4j-core-[version #].jar
install_dir/lib/
```

All jar files in the `lib` folder must be in the CLASSPATH, and the folder that contains the `log4j2.xml` file must be in the CLASSPATH.

On the Windows command line, you can choose to include all files in the `install_dir/lib`, or you can choose to list each jar file that you want to include. The following example shows how to include all files:

```
classpath install_dir/lib/*;install_dir/lib;
codeph;
```

On *Nix:

```
classpath install_dir/lib/*:install_dir/lib
```

To list each jar file individually, rather than using `install_dir/lib/*`, pass the database connection information to the driver manager in the form of a connection URL. Using the following URL format as an example, you can substitute values that are specific to your database management system:

```
jdbc:rs:dv://host-name:port-number[;property=value[;...]]
```

Substitute the `host-name` with the IP address or computer name of the server that hosts the database management system. Substitute the `port-number` with the TCP/IP port number on which the server listens for incoming requests. Multiple connection properties must be separated by semicolons.

Note: On the command line on Windows, if a property value contains a semi-colon, you must enclose the whole connection string in double quotation marks. On *Nix, the same is true for the colon character.

2. Define the connection string and the connection in your application.

The JDBC driver uses the JDBC Driver Manager to connect to a database server. To connect to the database management system, use the `DriverManager.getConnection()` method to pass the connection URL as the argument, as shown in the following example:

```
Connection conn = DriverManager.getConnection(
    "jdbc:rs:dv://host-name:port-number;
    DatabaseType=DRDAorDB2;
    user=user-id;password=user-password;
    subsystem=my-subsystem;
    Charset=CharsetName");
```

Note: Different database management systems have different required and optional connection properties. You must understand the DBMS to which you want to connect in order to properly configure the JDBC driver.

By default, the `DatabaseType` property is `DRDAorDB2`, which defaults the `subsystem` value to `NONE`. To specify your subsystem, set the value accordingly.

The `Charset` must match the `SQLENGDFLTCCSID` parameter setting of the server, and it must be compatible with the `MCCSID` of the target DB2 subsystem.

To connect to the server, use the following syntax:

```
Connection conn = DriverManager.getConnection(
    "jdbc:rs:dv://host-name:port-number;
    DatabaseType=DVS;
    user=user-id;password=user-password;
    Charset=CharsetName");
```

You can use the `hashpassword.cmd` utility to generate a hashed format of your text password. You can include the hashed password in the connection string or `.ini` file for your application. The script prompts the user for the plain text password. To avoid prompting, the password can be specified as a command-line argument.

For information on interface methods, see [“JDBC driver APIs” on page 95](#)

JDBC performance management

This section describes how you can improve JDBC driver performance when sending and receiving data to and from the client.

To optimize JDBC driver performance, choose from the following options:

Buffering data

When sending large amounts of data to the client or the server, one way to optimize performance is by choosing the appropriate size that the driver uses to divide and send the buffers of data.

The JDBC driver communicates with a server using the Communication Buffer (CMBU) protocol. CMBU specifies that all communications occur with one or more buffers. The maximum size of a buffer is set using the `MaximumBufferSize(MXBU)` JDBC property and the `NETWORKBUFFERSIZE` parameter in the server configuration file. You can use the `MaximumBufferSize` JDBC property as a basic building block for controlling performance.

Network latency that occurs between the client and the server can have a negative impact on performance. When setting the `MaximumBufferSize` value, consider the distance between the client and the server. The buffer size that works best for clients and servers that are closer in proximity (low latency), may not be the buffer size that works best for clients and servers that are not close in proximity (high latency).

When you execute a large SQL `INSERT` statement, or a batch of `INSERT` statements, the statements are divided into multiple data buffers that are no larger than the size you specify for `MaximumBufferSize`. The buffers are then sent to the server. The following illustration shows the buffers of `INSERT` statements being sent to the server:

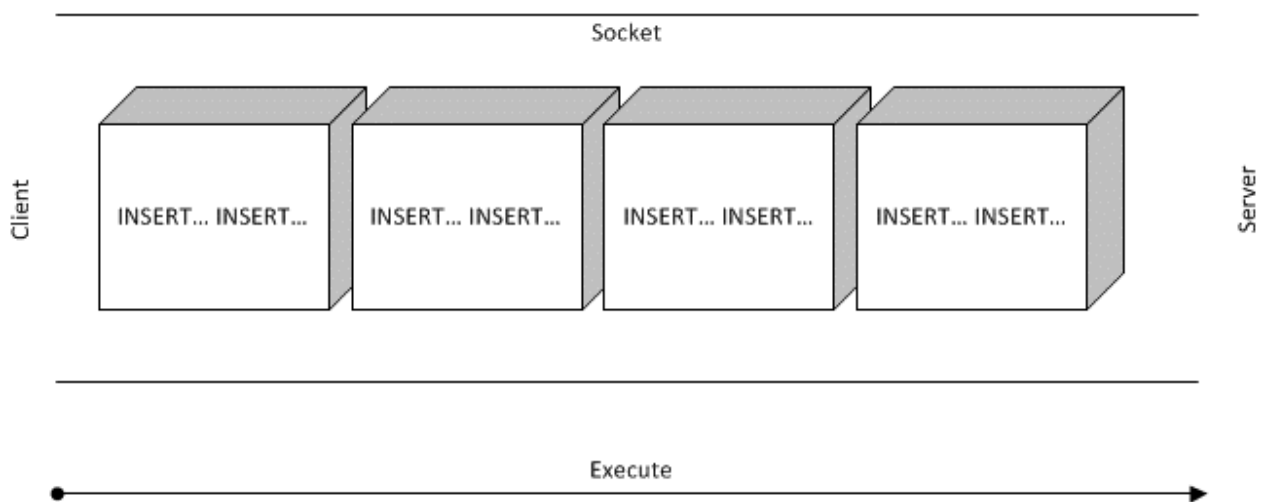


Figure 1: JDBC driver sends data requests using multiple buffers

The actual `MaximumBufferSize` value that the JDBC driver uses is the result of a handshake between the driver and the server. When the driver logs on to the server, it requests that the server support the given `MaximumBufferSize`. In the logon reply, the server tells the driver the actual buffer size value that the driver must use (`NETWORKBUFFERSIZE`). This value may be less than the driver requested value.

After executing a SQL `SELECT` statement, the server divides and returns the requested rows of data in buffers that have been sized appropriately, one buffer at a time. The following illustration shows the server sending the client rows of data in a single buffer:

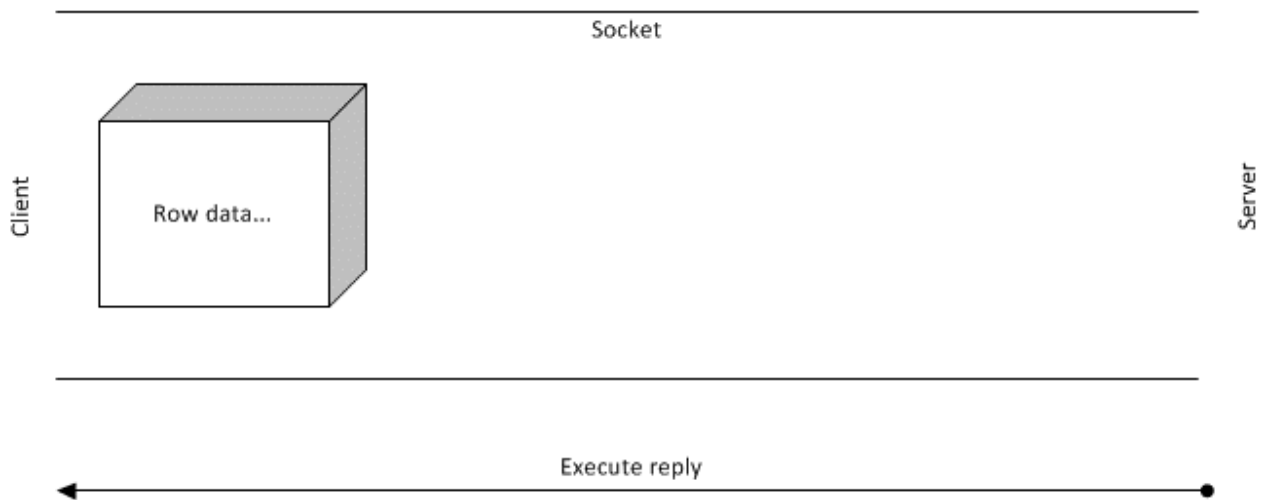


Figure 2: Server returns data one buffer at a time

The client can call `next()` on the result set until all rows are read. When the `next()` call no longer has a row to read from the buffer, the driver issues a request to the server for another row buffer. This cycle continues until the client reads all rows from all buffers. This happens transparently to the JDBC call site. The following code sample shows how this is implemented:

```
Connection con = DriverManager.getConnection(
    "jdbc:rs:dv://host:port;Key=Value;Key=Value;",
    username,
    password);
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");

while (rs.next()) {
    int x = rs.getInt("a");
    String s = rs.getString("b");
    float f = rs.getFloat("c");
}
```

During certain `next()` API calls, the client can experience a pause while the driver fetches another buffer of row data. To remove this pause, enable parallel IO buffering by setting the `ParallelIoBufferCount` (PIOBC) JDBC driver property.

Parallel IO

To improve performance, enable Parallel IO.

When Parallel IO is enabled, the JDBC driver creates a separate parallel IO thread from which a given number of buffers that are sent from the server are pre-fetched, read, and placed in a queue. The driver tries to keep the queue as full as possible so that there is always at least one buffer ready for the `next()` API call to use. This eliminates the pause that can occur as a result of certain `next()` calls. The client continues to call the `next()` buffer in the results set, until all rows in all buffers are read and queued.

The `ParallelIoBufferCount` (PIOBC) property determines the number of buffers to pre-fetch on the parallel IO thread. The illustration that follows, shows row data being sent to the client using both the main and the parallel thread.

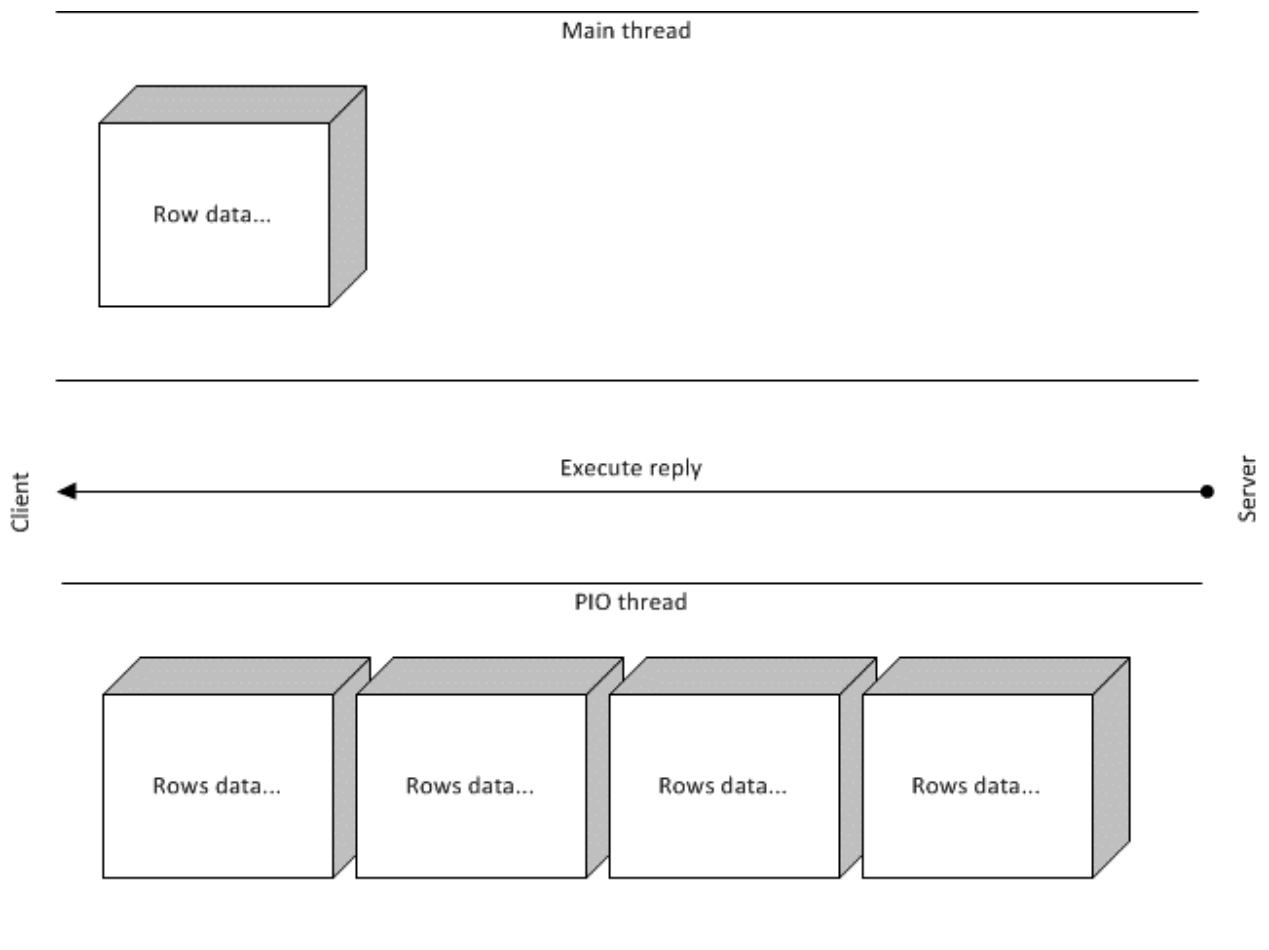


Figure 3: Parallel IO pre-fetches buffers

Materializing row bytes into Java objects on the thread

In addition to prefetching buffers, the Parallel IO thread can be used to materialize row data into their expected Java object representations, as illustrated in the following figure:

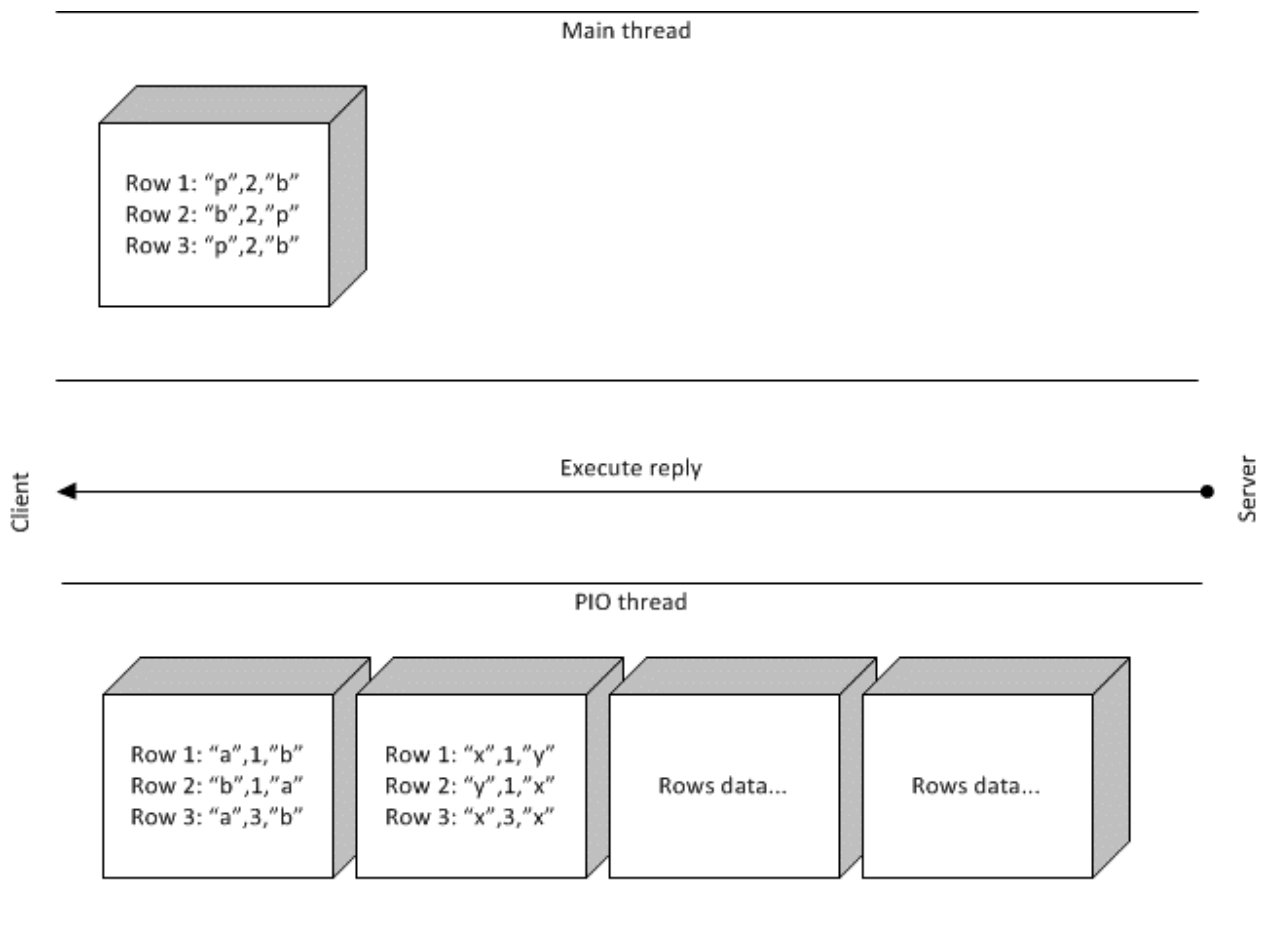


Figure 4: Materializing prefetched row data

The following sample code can be used to convert mainframe bytes to Java objects on the Parallel IO thread. The schema assumes that when the main thread calls a get API on the result set for a given column, that it will use the same get API on subsequent rows; for example `getString()`.

```
Connection con = DriverManager.getConnection(
    "jdbc:myDriver:myDatabase",
    username,
    password);

Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT a, b, c FROM Table1");

while (rs.next()) {
    int x = rs.getInt("a");
    String s = rs.getString("b");
    float f = rs.getFloat("c");
}
```

The `MapReduceFillValueThreshold` (MRFVT) and `MapReduceFillValueMaximumInitialSize` (MRFVMIS) JDBC property settings govern this behavior. When the MapReduce or Parallel IO read queue fills up beyond the `MapReduceFillValueThreshold` size, the driver materializes the column values. The default is off (-1) and a reasonable experimental value is two buffers.

When the MR or PIO read queue fills up beyond the `MapReduceFillValueThreshold` value, the `MapReduceFillValueMaximumInitialSize` row count is the initial capacity of the result row pre-fetch cache for a given buffer. The arbitrary default is 20,000 rows per buffer and it grows dynamically from there.

MapReduce

MapReduce is a the JDBC driver-controlled feature that is used to improve performance by reading results from the server faster.

Choose from the following MapReduce options:

- Server-controlled MapReduce – MapReduce is performed on the server using the JDBC driver.
- Client-controlled MapReduce – MapReduce is performed on the client and the JDBC driver is used for a single connection.

If you are using MapReduce with RDBMS or IMS, you must complete the meta data repository configuration requirements. See "MapReduce" in the *Administrator's Guide*.

Server controlled MapReduce

When configuring MapReduce for the server, the mapping step spawns threads that execute a query using one or more server connections. Each thread hosts one connection to one server. The reduce step reads results on each thread in parallel for each connection, and then transparently presents the data to the JDBC call sites.

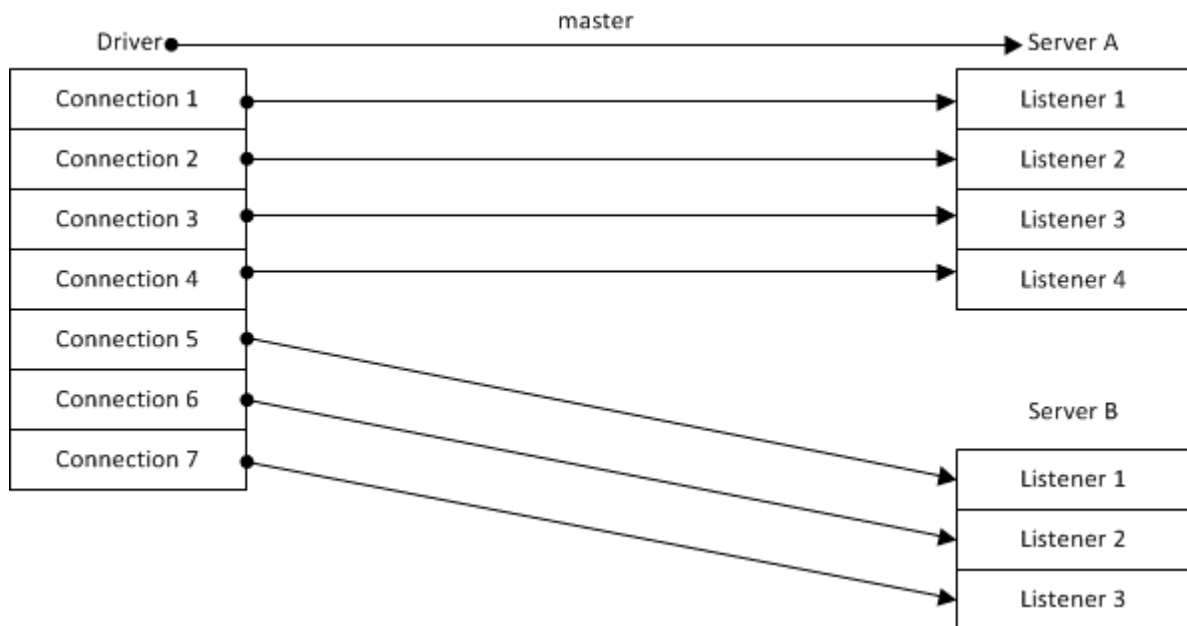


To configure MapReduce to use a single server connection, set the MapReduceClient JDBC property as follows:

```
MapReduceClient = (Hostname, Port, TaskCount)
```

For example: MapReduceClient= (dvs.example.com, 9999, 7)

As the following illustration shows, MapReduce can also be configured to use multiple server connections.



To configure MapReduce to use multiple server connections, set the MapReduceClient property as follows:

```
MapReduceClient = (Hostname1, Port, TaskCount1), (Hostname2, Port, TaskCount2),...
```

For example: MapReduceClient = (dvs1.example.com, 9999, 4), (dvs2.example.com, 1111, 3)

To further control MapReduce to use a specific range of clients, set the MapReduceClient property as follows:

```
MapReduceClient = (Hostname, Port, maxClientNo, startClientNo, endClientNo)
```

For example, the following specifies MapReduce over two servers for client numbers 1 through 4 and 5 through 7: MapReduceClient = (host1, 1200, 7, 1, 4), (host2, 1200, 7, 5, 7)

The maxClientNo must be the same for both servers.

Client controlled MapReduce

In some use cases, such as when using Apache Spark, you may want to let the client perform the MapReduce process and still benefit from the server's MapReduce feature. The JDBC driver supports this use case by specifying a single connection as the JDBC MapReduce connection. That single connection is then available for use by a group of specified connections. The JDBC driver only manages the one connection. The client must aggregate the results from each individual connection that is being managed.



To enable client-controlled MapReduce, set the `MapReduceClientCount` and `MapReduceClientNumber` driver properties.

The `MapReduceClientCount` property is used to specify the total number of connections associated with the group of client connections. The `MapReduceClientNumber` property specifies a specific client connection within the group, and has a value between the number 1 and the number specified for the `MapReduceClientCount` property.

The JDBC driver executes queries using the single MapReduce connection for the client connection specified in the `MapReduceClientNumber` property. Only the rows of data for the specified connection are returned, as opposed to using `MapReduceClientCount` over one or more connections to get all rows of data.

To configure client-side MapReduce, set the JDBC driver `MapReduceClientNumber` and `MapReduceClientCount` parameters as follows:

```
MapReduceClientNumber,MapReduceClientCount
```

For example: `MapReduceClientNumber=2;MapReduceClientCount=7;`

In this example, the JDBC driver only creates a single connection, indicated as connection 2 out of 7 available connections. Using a framework like Apache Spark, you must create and manage all remaining connections and aggregate the results from each of those connections.

JDBC driver APIs

The following tables document the methods that are not supported by each of the java interfaces.

The following methods are not supported with the `java.sql.Connection` interface:

Return type	Method name
Blob	<code>createBlob()</code>
Clob	<code>createClob()</code>
NClob	<code>createNClob()</code>
SQLXML	<code>createSQLXML()</code>
Struct	<code>createStruct(String typeName, Object [] attributes)</code>
String	<code>getSchema()</code>
int	<code>getTransactionIsolation()</code>
boolean	<code>isReadOnly()</code>

Return type	Method name
void	releaseSavepoint (Savepoint savepoint)
void	rollback (Savepoint savepoint)
Savepoint	setSavepoint ()
Savepoint	setSavepoint (String name)

The following methods are not supported with the `java.sql.PreparedStatement` interface:

Return type	Method name
void	setArray(int parameterIndex, Array x)
void	setAsciiStream(int parameterIndex, InputStream x)
void	setAsciiStream(int parameterIndex, InputStream x, int length)
void	setAsciiStream(int parameterIndex, InputStream x, long length)
void	setBinaryStream(int parameterIndex, InputStream x)
void	setBinaryStream(int parameterIndex, InputStream x, int length)
void	setBinaryStream(int parameterIndex, InputStream x, long length)
void	setBlob(int parameterIndex, Blob x)
void	setBlob(int parameterIndex,InputStream inputStream)
void	setBlob(int parameterIndex,InputStream inputStream, long length)
void	setCharacterStream(int parameterIndex,Reader reader)
void	setCharacterStream(int parameterIndex,Reader reader, int length)
void	setCharacterStream(int parameterIndex,Reader reader, long length)
void	setClob(int parameterIndex, Clob x)
void	setClob(int parameterIndex, Reader reader)
void	setClob(int parameterIndex, Reader reader, long length)
void	setNCharacterStream(int parameterIndex,Reader value)
void	setNCharacterStream(int parameterIndex, Reader value, long length)
void	setNClob(int parameterIndex, NClob value)
void	setNClob(int parameterIndex, Reader reader)

Return type	Method name
void	setNClob(int parameterIndex, Reader reader, long length)
void	setRowId(int parameterIndex, RowId x)
void	setSQLXML(int parameterIndex, SQLXML xmlObject)

The following methods are not supported with the `java.sql.Statement` interface:

Return type	Method name
ResultSet	getGeneratedKeys()
void	setEscapeProcessing(boolean enable)

The following methods are not supported with the `java.sql.ResultSet` interface:

Return type	Method name
boolean	absolute(int row)
void	afterLast()
void	beforeFirst()
boolean	first()
Array	getArray(int columnIndex)
Array	getArray(String columnLabel)
NClob	getNClob(int columnIndex)
NClob	getNClob(String columnLabel)
ref	getRef(String columnLabel)
SQLXML	getSQLXML(int columnIndex)
SQLXML	getSQLXML(String columnLabel)
boolean	last()
boolean	previous()
boolean	relative(int rows)
void	updateArray(String columnLabel, Array x)
void	updateAsciiStream(int columnIndex, InputStream x)
void	updateAsciiStream(int columnIndex, InputStream x, int length)
void	updateAsciiStream(int columnIndex, InputStream x, long length)
void	updateAsciiStream(String columnLabel, InputStream x)
void	updateAsciiStream(String columnLabel, InputStream x, int length)
void	updateAsciiStream(String columnLabel, InputStream x, long length)

Return type	Method name
void	updateBigDecimal(int columnIndex, BigDecimal x)
void	updateBigDecimal(String columnLabel, BigDecimal x)
void	updateBinaryStream(int columnIndex, InputStream x)
void	updateBinaryStream(int columnIndex, InputStream x, int length)
void	updateBinaryStream(int columnIndex, InputStream x, long length)
void	updateBinaryStream(String columnLabel, InputStream x)
void	updateBinaryStream(String columnLabel, InputStream x, int length)
void	updateBinaryStream(String columnLabel, InputStream x, long length)
void	updateBlob(int columnIndex, Blob x)
void	updateBlob(int columnIndex, InputStream inputStream)
void	updateBlob(int columnIndex, InputStream inputStream, long length)
void	updateBlob(String columnLabel, Blob x)
void	updateBlob(String columnLabel, InputStream inputStream)
void	updateBlob(String columnLabel, InputStream inputStream, long length)
void	updateBoolean(int columnIndex, boolean x)
void	updateBoolean(String columnLabel, boolean x)
void	updateByte(int columnIndex, byte x)
void	updateByte(String columnLabel, byte x)
void	updateBytes(int columnIndex, byte[] x)
void	updateBytes(String columnLabel, byte[] x)
void	updateCharacterStream(int columnIndex, Reader x)
void	updateCharacterStream(int columnIndex, Reader x, int length)
void	updateCharacterStream(int columnIndex, Reader x, long length)
void	updateCharacterStream(String columnLabel, Reader reader)

Return type	Method name
void	updateCharacterStream(String columnLabel, Reader reader, int length)
void	updateCharacterStream(String columnLabel, Reader reader, long length)
void	updateClob(int columnIndex, Clob x)
void	updateClob(int columnIndex, Reader reader)
void	updateClob(int columnIndex, Reader reader, long length)
void	updateClob(String columnLabel, Clob x)
void	updateClob(String columnLabel, Reader reader)
void	updateClob(String columnLabel, Reader reader, long length)
void	updateDate(int columnIndex, Date x)
void	updateDate(String columnLabel, Date x)
void	updateDouble(int columnIndex, double x)
void	updateDouble(String columnLabel, double x)
void	updateFloat(int columnIndex, float x)
void	updateFloat(String columnLabel, float x)
void	updateLong(int columnIndex, long x)
void	updateLong(String columnLabel, long x)
void	updateNCharacterStream(int columnIndex, Reader x)
void	updateNCharacterStream(int columnIndex, Reader x, long length)
void	updateNCharacterStream(String columnLabel, Reader reader)
void	updateNCharacterStream(String columnLabel, Reader reader, long length)
void	updateNClob(int columnIndex, NClob nClob)
void	updateNClob(int columnIndex, Reader reader)
void	updateNClob(int columnIndex, Reader reader, long length)
void	updateNClob(String columnLabel, NClob nClob)
void	updateNClob(String columnLabel, Reader reader)
void	updateNClob(String columnLabel, Reader reader, long length)
void	updateNString(int columnIndex, String nString)
void	updateNString(String columnLabel, String nString)

Return type	Method name
void	updateNull(int columnIndex)
void	updateNull(String columnLabel)
void	updateObject(int columnIndex, Object x)
void	updateObject(int columnIndex, Object x, int scaleOrLength)
void	updateObject(String columnLabel, Object x)
void	updateObject(String columnLabel, Object x, int scaleOrLength)
void	updateRef(int columnIndex, Ref x)
void	updateRef(String columnLabel, Ref x)
void	updateRowId(int columnIndex, RowId x)
void	updateRowId(String columnLabel, RowId x)
void	updateShort(int columnIndex, short x) updateShort(String columnLabel, short x)
void	updateSQLXML(int columnIndex, SQLXML xmlObject)
void	updateSQLXML(String columnLabel, SQLXML xmlObject)
void	updateString(int columnIndex, String x)
void	updateString(String columnLabel, String x)
void	updateTime(int columnIndex, Time x)
void	updateTime(String columnLabel, Time x)
void	updateTimestamp(int columnIndex, Timestamp x)
void	updateTimestamp(String columnLabel, Timestamp x)

Chapter 3. JDBC Gateway

The *JDBC Gateway* is a Data Service distributed application server that allows direct connectivity to JDBC 4.0 data sources. The use of another federation server is not required.

Data sources

The JDBC Gateway solution is designed to work with any JDBC 4.0 compliant database. The following combinations of JDBC databases and drivers have been tested and verified to be supported by the JDBC Gateway:

- Hadoop 2.9.2 with the Hive 2.0 standalone JDBC driver
- Oracle 12 using the Oracle Thin Driver, version 6
- PostgreSQL version 11.1 using the JDBC driver version 42.2.5

Note: The degree of JDBC compliance can vary across different driver vendor implementations and versions. In some cases, there may be interoperability problems when trying to use a particular JDBC driver to access a particular DBMS.

Components

The JDBC Gateway consists of the following components:

- *JDBC Gateway server*. The server is the backend component that allows communication with the Data Service server.
- *JDBC Gateway administrative console*. The administrative console is the front-end web component that you use to configure your data sources. Only a single user (web client) can access the JDBC Gateway administrative console at a time. When installing the JDBC Gateway, you must specify a specific user ID for this purpose. This user ID is an internal application ID that allows access to the web user interface.

Getting started

Use the following procedure to access your first data source using the JDBC Gateway:

1. Install the JDBC Gateway. See "Installing the JDBC Gateway" in the *Installation and Customization Guide*.
2. Start the JDBC Gateway server. See "Starting the JDBC Gateway server" in the *Installation and Customization Guide*.
3. Launch the JDBC Gateway administrative console in a supported browser using the following URL:

```
http://host:port
```

See "Launching the JDBC Gateway administrative console" in the *Installation and Customization Guide*.

4. In the JDBC Gateway administrative console, perform the following steps:
 - a. Determine the port that the JDBC Gateway will use for listening for incoming DRDA requests. You can review or change the port using the **Server Status** area of the JDBC Gateway administrative console. See ["Using the JDBC Gateway administrative console" on page 102](#).
 - b. Set up access to the data source by performing the following tasks:
 - 1) Locate and add JDBC driver information for the data source. See ["Adding JDBC driver information for a data source" on page 103](#).
 - 2) Create a data source definition entry, specifying the location name, driver, URL and user information. See ["Creating a data source definition entry" on page 105](#).
5. In the Data Service server, set up access to the data source by performing the following tasks:

- a. Register the connection to the JDBC Gateway by entering the location, host and the port for the data source.
- b. Enable the SEF rules and set global variables for the data source.

For information about these tasks, see [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106.

6. Use the Data Service studio to create virtual tables and views from the JDBC data source, just as you do for other supported sources, such as VSAM or IMS.

Using the JDBC Gateway administrative console

Use the JDBC Gateway administrative console to create and manage your data source definitions.

Before you begin

The JDBC Gateway must be installed, the JDBC Gateway server must be active, and the JDBC Gateway administrative console must be launched. See "Installing the JDBC Gateway" in the *Installation and Customization Guide*.

Procedure

Use the JDBC Gateway administrative console to create and manage your data source definitions. The following table describes the areas of the default JDBC Gateway view:

Field/Element	Action
Add New Data Source	Click this button add a new data source. For details, see “Creating a data source definition entry” on page 105.
Location JDBC URL	Displays a list of defined data sources. Select an entry to display properties and location information. • Location: Location name of the data source. Note: This value corresponds to the LOCATION parameter defined for the data source in the Data Service server. • JDBC URL: The URL that points to the data source.
Server Status	Displays and controls the JDBC Gateway server status and the DRDA listener port. • Status: JDBC Gateway server status. Click Start or Stop to control the server. • Port: The port on which the JDBC Gateway is listening for incoming DRDA requests. Click Edit to change the port number. This setting also allows you to control whether the server is started automatically when the JDBC Gateway startServer script is run. Note: This port value will be used when adding a JGATE database definition statement to the Data Service server configuration file (AZKSIN00).
Location Information	Displays the following details for selected data source entry: • Domain: Domain name of the JDBC Gateway. • Location: Name of the target data source. • Port: Port on which the JDBC Gateway is listening for incoming DRDA requests. Note: These values will be used when adding a JGATE database definition statement to the Data Service server configuration file (AZKSIN00). Click Test Connection to test the connection to the data source. If you have specified any information incorrectly you will not be able to connect.

Configuring access to data sources using the JDBC Gateway

Configure access to JDBC data sources that will be accessed using the JDBC Gateway.

To configure access for a data source, you must complete the following steps:

1. Add the compliant JDBC driver for the data source to the JDBC Gateway. See [“Adding JDBC driver information for a data source”](#) on page 103.
2. Create the data source definition entry in the JDBC Gateway, specifying the location name, driver, URL, and user information. See [“Creating a data source definition entry”](#) on page 105.
3. Configure the Data Service server for the data source. See [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106.

Adding JDBC driver information for a data source

Add JDBC driver information to the JDBC Gateway.

Before you begin

The JDBC Gateway must be installed, the JDBC Gateway server must be active, and the JDBC Gateway administrative console must be launched. See "Installing the JDBC Gateway" in the *Installation and Customization Guide*.

About this task

The JDBC Gateway requires a compliant JDBC driver for each data source to be accessed. You must locate and add JDBC driver information for each data source. The driver files must be accessible to the JDBC Gateway. The JDBC Gateway retains the defined JDBC driver information, and you would only repeat this specification process to add new drivers or make changes to the properties of an existing driver.

In preparation for this task, obtain the following driver information for the data source from the data source vendor or from the driver documentation:

- Driver class name. For example: `org.postgresql.Driver`
- Driver JAR files
- URL format. Each data source type has a unique URL format that is used to access the data and is specific by vendor. For example, for Postgres: `jdbc:postgresql://{host}:{port}/{database}`

To add JDBC driver information to the JDBC Gateway, using the JDBC Gateway administrative console, you will define the driver library for the data source, and then add the driver files to the library. Use the following procedure to add JDBC driver information for a data source.

Procedure

1. In the JDBC Gateway administrative console, select **Preferences > JDBC Libraries**.

The following table describes the areas of the page:

Area	Description
JDBC driver libraries	JDBC driver libraries that are already set up. Use the search bar to quickly locate information in the table.
Driver files	JAR files associated with selected driver library.
Details	Additional information about the selected driver library

2. Add a driver library by performing the following steps:
 - a) Click the **Add Driver** button.

b) In the **Add New Driver Library** window, provide the following information:

Field	Action
Enter new library name	Enter a name for the library. The JDBC driver information for each type of database is organized by libraries. It is recommended that the name that you specify describes the JDBC information that will be included in the library. For example, if you are adding JDBC driver information for accessing Postgres databases, you might call the library Postgres. However, this is a descriptive field and can include any text.
Driver class name	Specify the actual name of the driver class that will be used. This information can be found in your JDBC driver documentation. For example: <code>org.postgresql.Driver</code>
URL templates	Optional: Specify a generic example of a correctly formatted URL that could be used to connect to the database. For example, if you are adding JDBC driver information for accessing Postgres databases, you might specify the following JDBC URL template: <code>jdbc:postgresql://{host}:{port}/{database}</code> . The generic information as specified in the template is presented when you are adding data sources, where you will replace the generic information with the specific database information.

Note: The **Validate** and **JDBC Driver Properties** options are not applicable until the driver files have been added.

- c) Click **OK**.
3. Add JDBC driver files to the library by performing the following steps:
- Click the **Add Driver Files** button.
 - In the **Add Files** dialog, click **Add** and specify the path to the JDBC driver files to add.
 - Click **OK**.
4. Optional: Update JDBC driver information as follows:
- To edit the JDBC driver library information, validate the drivers, or add connection keywords, select an existing JDBC driver library from the list and click **Edit Driver**. The **Edit Driver Library** window opens where you can make changes to the library name, class name, and URL templates. You can also use the **Validate** option to validate the driver files, and the **JDBC Driver Properties** option to enter driver-specific connection keywords.
 - To remove a JDBC driver library, select an existing JDBC driver library from the list and click **Remove Driver**. The library, including all the JAR files that it contains, is removed.
 - To remove a JAR file from a JDBC driver library, select an existing file from the list and click **Remove Driver File**. The file is removed.
5. Click **OK**.

Results

The JDBC driver information is saved.

Note: You must repeat this process for each JDBC driver that will be used to access a data source type.

What to do next

Create the data source definition entry, specifying the location name, driver, URL, and user information. [“Creating a data source definition entry” on page 105.](#)

Creating a data source definition entry

Configure the JDBC Gateway for access to data sources.

Before you begin

The JDBC Gateway must be installed, the JDBC Gateway server must be active, and the JDBC Gateway administrative console must be launched. See [“Installing the JDBC Gateway” in the *Installation and Customization Guide*.](#)

Also, the compliant JDBC driver should be added to the JDBC Gateway. See [“Adding JDBC driver information for a data source” on page 103.](#)

About this task

Use the following procedure to create a data source definition entry. This data source definition entry is made in the JDBC Gateway administrative console and is used for access to the data source by the JDBC Gateway.

Procedure

1. In the JDBC Gateway administrative console, click the **Add New Data Source** button.
2. In the **JDBC Gateway** dialog, complete the following fields.

Field	Action
Location	Enter the location name. A valid value is a string 1 - 16 characters. For example: ORCL. Note: This value must match the LOCATION value that will be specified for the corresponding data source definition in the Data Service server configuration file.
Connection Parameters	Enter the JDBC connection information, as follows: • JDBC Driver: Specify the library for the JDBC driver that will be used to access the data source. Select a library from the drop-down list, or click the ellipsis (...) option to the right of the field to open the Select JDBC Driver dialog where you can create additional JDBC driver libraries. (For more information, see “Adding JDBC driver information for a data source” on page 103.) • JDBC URL: Specify the URL that points to the data source to which you want to connect. The format for the URL can be displayed in the drop-down list if a JDBC URL template was supplied when the driver was configured. Note: You can also use the Build URL by URL-Template dialog box to form the correct string. Click Build URL to open the Build URL by URL-Template dialog box. From the JDBC URL drop-down list, select the template. In the table, specify the server, port, and database information and click OK. The result URL string is added to the JDBC URL list. This feature is available if a JDBC URL template was provided when the driver was configured. • Advanced: Click Advanced to specify any driver-specific connection string keywords and their values that will be used for the data source. The list of available advanced properties will change depending on both the type of driver being used, and the version of the driver. For information on any keywords that are required by a selected database driver, see the documentation for the driver.

Field	Action
Set User Information	Click Set User Information to provide authorization information used when accessing the data source. Provide the following information on the User Information dialog: • User ID and password are required: Select this option to require the use of a user ID and password when accessing the data source. If the data source allows access without a user ID and password, selecting this option will override that allowance. • Allow users to save password: Select this option to allow users to save passwords. • Allow users to change password: Select this option to allow users to change passwords. (Note: This option is for Db2 only.) • User name and Password: Specify the user ID and password that will be used to access the data source. The user ID and password that you specify when connecting to the data source are used to authorize the user.
Test Connection	Click Test Connection to test the connection to the data source. If you have specified any information incorrectly, you will not be able to connect.

3. Click **Finish**.

Results

The connection to the data source is validated. If successful, the data source location is added to the list of available data sources.

What to do next

[Configure the Data Service server for the JDBC Gateway source.](#)

Configuring the Data Service server for JDBC Gateway sources

Configure the Data Service server for use with the JDBC Gateway.

Before you begin

Configure access to the data source using the JDBC Gateway. See [“Creating a data source definition entry”](#) on page 105.

About this task

To use the JDBC Gateway to connect to your data source, the following changes must be made to the Data Service server:

- The DEFINE DATABASE TYPE value must be set, as follows:

```
"DEFINE DATABASE TYPE(JGATE)"
```

JGATE

DDF endpoint is the JDBC Gateway.

- Optionally, the following utility and SEF procedure can be configured in support of TYPE(JGATE):

AZKDRATH

A utility that sets encrypted passwords in GLOBALU variables. You can also use this utility to list existing credential information.

AZKEJGAG

An ATH rule that switches credentials when connecting to a JGATE data source using DRDA. This rule uses AES encrypted passwords stored as GLOBALU system variables.

Procedure

1. In the Data Service server configuration file, register the connection to the JDBC Gateway using a definition statement, such as the following example:

```
"DEFINE DATABASE TYPE(JGATE) "
      "NAME(name) "
      "LOCATION(location) "
      "DDFSTATUS(ENABLE) "
      "DOMAIN(your.domain.name) "
      "PORT(port) "
      "IPADDR(1.1.1.1) "
      "CCSID(37) "
```

The following table lists the parameters:

Parameter	Description	Valid values
AUTHTYPE	Authentication type. This can be either DES for Diffie Hellman Encryption Standard or AES for Advanced Encryption Standard. When AUTHTYPE is not supplied, the default is DES. To force AES, the option must be added to the DEFINE DATABASE statement. Each server can be different in what is supported as to AES/DES. For this setting to have effect, you must specify a security mechanism (SECMEC) that requests encryption.	DES Diffie Hellman Encryption Standard (default value) AES Advanced Encryption Standard.
CCSID	Specify the EBCDIC single-byte application CCSID (Coded Character Set Identifier) configured for this RDBMS subsystem on the RDBMS installation panel DSNTIPF, option 7. (<i>Optional</i>)	Refer to the RDBMS vendor documentation for a list of valid CCSIDs.
DDFSTATUS	The DDF activation status can be altered online by using the ISPF 4-DB2 dialog panels. (<i>Required</i>)	ENABLE Make this DDF definition active. DISABLE DDF endpoint is not used.
DOMAIN	The domain name or hostname on which the JDBC Gateway server is running. Either DOMAIN or IPADDR is required, but not both.	No default value.

Parameter	Description	Valid values
IPADDR	The dot-notation IPV4 address of the host on which the JDBC Gateway server is running. Either DOMAIN or IPADDR is required, but not both.	If this parameter is not specified, the value 127.0.0.1 (local host) is the default. For group director definitions, use the DVIPA IP address of the group director.
LOCATION	For JGATE: The location name specified in the JDBC Gateway data source definition entry. See “Creating a data source definition entry” on page 105. (Required)	A valid value is a string 1 - 16 characters.
NAME	The database name as known to the server. (Required)	A valid value consists of 1 - 4 characters. Clients use this ID when they request access to a specific downstream database server.
PORT	The TCP/IP port on which the JDBC Gateway server is listening. (Required)	A valid 1-5 numeric string. If this keyword is not entered, the default DRDA port number 443 is used.
SECMEC	The DRDA security mechanism in force.	EUSRIDPWD Encrypt the user ID and password. USERIDPWD User ID and password are sent as is. No encryption is used. USRIDONL User ID is sent as is. No encryption is used for the user ID only (client security). USRENCPWD Encrypt password only.
TYPE	Defines the DDF endpoint type. JGATE DDF endpoint is the JDBC Gateway.	When using the JDBC Gateway, JGATE is the valid value.

2. Optional: To define alternate authentication information, use the sample job AZKDRATH to add a global default user definition or authentication information for specific mainframe users as follows:
 - a) Locate the AZKDRATH member in the *hlq.SAZKCNTL* data set.
 - b) Modify the JCL according to the instructions provided in the AZKDRATH member.

When adding the SYSIN statements that define the alternate credentials for logging in to your JDBC Gateway source, as instructed in the JCL, make sure to specify the correct DBTYPE. For JDBC Gateway sources, specify DBTYPE=JGATE.

- c) Submit the job.
- d) Optional: To verify the information stored in the GLOBALU variables and list existing authentication, use the REPORT=SUMMARY statement in the AZKDRATH member and submit the job.
3. Optional: If using alternate authentication information, auto-enable the SEF ATH rule SAZKXATH(AZKEJGAG) to provide the logon credentials to each JDBC Gateway data source instance. Global variables are used to define alternate authentication credential mapping for the SEF ATH rule.
 - a) On the IBM Open Data Analytics for z/OS - Primary Option Menu, select option **E** for Rules Mgmt.
 - b) Select option **2** for SEF Rule Management.
 - c) Enter * to display all rules, or ATH to display only authentication rules.
 - d) Enable the rule by specifying E and pressing Enter.
 - e) Set the rule to Auto-Enable by specifying A and pressing Enter.Setting the rule to Auto-enable activates the rule automatically when the server is restarted.
4. Restart the Data Service server.

Results

The connection between the JDBC Gateway and the Data Service server for the JDBC data source has been defined.

What to do next

Use the Data Service Studio to create virtual tables and views from the JDBC data source.

Example: Configuring access to Oracle data

Configure the JDBC Gateway for access to Oracle data.

Before you begin

The JDBC Gateway must be installed, the JDBC Gateway server must be active, and the JDBC Gateway administrative console must be launched. See "Installing the JDBC Gateway" in the *Installation and Customization Guide*.

About this task

Use the following procedure to configure access to Oracle data.

Procedure

1. Download the Oracle Thin Driver from the Oracle website. For example, ojdbc8.jar.
2. In the JDBC Gateway administrative console, select **Preferences > JDBC Libraries**, and then complete the following steps:
 - a) Select the row for the **Driver Library Name** Oracle Thin Driver in the table, and click **Add Driver Files**.
 - b) Use the **Add Files** dialog to add the Oracle Thin Driver file.
 - c) Click **OK** to close the **JDBC Libraries** preference page.
3. Create a JDBC Gateway data source for Oracle as follows:
 - a) Select **File > New > Other**, and then in the **New** wizard dialog, select **Data Source** and click **Next**.
 - b) Complete the following fields:

Field	Action
Location	Enter the location name. For example, Oracle.

Field	Action
Connection Parameters	Enter the connection parameters: • JDBC Driver: From the drop-down list, select Oracle Thin Driver. • JDBC URL: Enter the JDBC URL as follows: jdbc:oracle:thin:@//oracle-host:1521/ORCL
Set User Information	Click Set User Information , and enter the credentials for accessing the Oracle database, as follows: • User name: <i>OracleUser</i> • Password: <i>OraclePwd</i>

- c) Click **Test Connection**.
- d) Click **Finish**.
4. In the Data Service server configuration file, register the connection to the JDBC Gateway data source using a definition statement, such as the following example:

```
"DEFINE DATABASE TYPE(JGATE)"
    "NAME(ORCL)"
    "LOCATION(Oracle)"
    "DDFSTATUS(ENABLE)"
    "SECMEC(USRIDPWD)"
    "PORT(1527)"
    "IPADDR(10.26.4.125)"
    "CCSID(37)"
    "IDLETIME(110)"
```

For details about this statement, see [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106.

5. In the Data Service server, enable rule AZKEJGAG. For more information, see [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106..

Results

The following connections have been established:

- The connection from the JDBC Gateway to the Oracle data source
- The connection between the JDBC Gateway and the Data Service server for the Oracle data source

What to do next

Use the Data Service Studio to create virtual tables and views to access the Oracle data.

Example: Configuring access to Hadoop data

Configure the JDBC Gateway for access to Hadoop data.

Before you begin

The JDBC Gateway must be installed, the JDBC Gateway server must be active, and the JDBC Gateway administrative console must be launched. See "Installing the JDBC Gateway" in the *Installation and Customization Guide*.

About this task

Configuring access to Hadoop data requires both the standalone Hive 2.0 JDBC jar and the Hadoop Common jar driver files.

Use the following procedure to configure access to Hadoop data.

Procedure

1. Download the Apache Hive and Apache Hadoop driver files.
2. In the JDBC Gateway administrative console, select **Preferences > JDBC Libraries**, and then complete the following steps:
 - a) Click **Add Driver**, complete the following fields, and click **OK**:

Field	Action
Enter new library name	Enter HADOOP
Driver class name	Enter org.apache.hive.jdbc.HiveDriver

- b) Select the row for the **Driver Library Name** HADOOP in the table, and click **Add Driver Files**.
 - c) Use the **Add Files** dialog to add the driver files. You need to include both the standalone Hive 2.0 JDBC jar and the Hadoop Common jar.
 - d) Click **OK** to close the **JDBC Libraries** preference page.
3. Create a JDBC Gateway data source for Hadoop as follows:
 - a) Select **File > New > Other**, and then in the **New** wizard dialog, select **Data Source** and click **Next**.
 - b) Complete the following fields:

Field	Action
Location	Enter the location name. For example, Hadoop.
Connection Parameters	Enter the connection parameters: • JDBC Driver: From the drop-down list, select HADOOP. • JDBC URL: Enter the JDBC URL as follows: jdbc:hive2://hadoop-host:10000/default
Set User Information	Click Set User Information , and enter the credentials for accessing the Hadoop database, as follows: • User name: <i>HadoopUser</i> • Password: <i>HadoopPwd</i>

- c) Click **Test Connection**.
 - d) Click **Finish**.
4. In the Data Service server configuration file, register the connection to the JDBC Gateway data source using a definition statement, such as the following example:

```
/*-----*/
/*      HADOOP      */
/*-----*/
"DEFINE DATABASE TYPE(JGATE)"
    "NAME(HIVE)"
    "LOCATION(Hadoop)"
    "DDFSTATUS(ENABLE)"
    "SECMEC(USRIDPWD)"
    "PORT(1527)"
    "IPADDR(10.26.4.125)"
    "CCSID(37)"
    "IDLETIME(110)"
```

For details about this statement, see [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106.

5. In the Data Service server, enable rule AZKEJGAG. For more information, see [“Configuring the Data Service server for JDBC Gateway sources”](#) on page 106..

Results

The following connections have been established:

- The connection from the JDBC Gateway to the Hadoop data source
- The connection between the JDBC Gateway and the Data Service server for the Hadoop data source

What to do next

Use the Data Service Studio to create virtual tables and views to access the Hadoop data.

Setting preferences

The **Preferences** dialog is used to set user preferences and add necessary drivers.

The **Preferences** window consists of two panes. The left pane displays the list of preferences groups and the right pane displays the page for the selected group. The following groups of preferences are displayed in the **Preferences** window:

- [JDBC Libraries](#)
- [Log](#)
- [Output](#)

Setting JDBC driver preferences

Use the **JDBC Libraries** preferences to set up and manage JDBC driver information for your data sources.

About this task

You can use the **JDBC Libraries** preferences page to review, define or update JDBC driver information for each type of database (such as Db2®, Informix®, Oracle) that will be accessed.

Use the following procedure to access the **JDBC Libraries** preferences page. For details about adding new driver definitions, see [“Adding JDBC driver information for a data source” on page 103](#).

Procedure

1. To access the **JDBC Libraries** page, select **Preferences > JDBC Libraries**.
All of the JDBC driver libraries that you have already set up are listed in the **JDBC driver libraries** area. The JAR files associated with selected driver library are listed in the **Driver files** area. Additional information about the selected driver library is displayed on the **Details** panel.
2. For information about adding or editing driver definitions, see [“Adding JDBC driver information for a data source” on page 103](#).

Setting log preferences

Use the **Log** page of the **Preferences** window to activate a log file that will track JDBC Gateway processing information.

About this task

The log file information can be useful in debugging.

It is recommended to leave the log level at the default setting of `error`. Only increase the level at the direction of IBM Software Support.

Use the following procedure to specify the log file preferences.

Procedure

1. Click **Preferences > Log**.

2. Check **Enable log** to activate the log file for debugging purposes. If this check box is selected, the log file option fields are enabled.
3. Check one or more of the log file options to indicate what information should be gathered. It is recommended that all options remain checked. The available log file options are as follows:
 - Print stack trace for log exceptions
 - Print log class and method
 - Print log user token
4. Click **Edit Log Categories** to modify the category level.

The following levels are available: none, emergency, alert, critical, error, warning, notice, info, debug, all.
5. Click **Apply** to save your preferences choices.
6. Click **Restore Defaults** to restore the default preference values.
7. Click **OK** to close the **Preferences** window.

Setting output preferences

You can use the **Output** page of the **Preferences** window to activate the **Output** view that tracks the information about errors and connections in the JDBC Gateway.

About this task

The information from the **Output** view can be useful for debugging. It can be delivered as a report in the **Output** view and automatically added to the log file.

Use the following procedure to specify the output file preferences:

Procedure

1. Click **Preferences > Output**.
2. On the **Output** page, you can specify the following options:
 - Show errors**

This option displays all error texts in the **Output** view.
 - Show connection status**

This option displays the statuses of connections to data sources in the **Output** view.
 - Automatically activate Output view**

When an error occurs or a message appears, this option automatically opens the **Output** view.
3. Click **Apply** to save your preferences choices.
4. Click **Restore Defaults** to restore the default preference values.
5. Click **OK** to close the **Preferences** window.

Troubleshooting

Collect troubleshooting data to provide to technical support.

About this task

Use the following procedure to collect troubleshooting data.

Procedure

1. Set the log level to debug. See [“Setting log preferences” on page 112](#).
2. Reproduce the issue.
3. Set the log level to the previous value.
4. Select **Help > Collect Support Data**.

5. Complete the fields and click **Save Report**.

Chapter 4. Creating your first Apache Spark program to access your z/OS data

Create an Apache Spark program to access mainframe data using Mainframe Data Service.

Before you begin

Verify that all required Data Service components are installed and configured. This also includes copying the JDBC driver files to the appropriate location on the z/OS mainframe. The location to use should be available in `spark.executor.extraClassPath`. If you choose a different location, you will need to provide that location on the `spark-submit` command line.

The following JDBC driver files are required:

- `dv-jdbc-[version #].jar` (the driver core implementation file)
- `log4j-api-[version #].jar` (logging framework API file)
- `log4j-core-[version #].jar` (logging framework implementation file)
- `log4j2.xml` (logging configuration)

For more details about installing and configuring Data Service components, see the *Installation and Customization Guide*.

About this task

You can use the example that follows to create your first Spark program. Using this example, you will access your mainframe data using the Spark SQL module with a JDBC data source. This example uses Scala as the programming language and uses the open source build tool (SBT). SBT is used to compile Scala programs to JAVA Archive (JAR) files that are deployed on Spark.

Note: For SQL and Scala details, see the DataFrames, Datasets and SQL ([sql-programming-guide](#)) page for your installed version of Spark available at [Apache Spark \(spark.apache.org\)](#) and [sbt for Scala \(www.scala-sbt.org\)](#).

Procedure

1. Create virtual tables to map your data to a relational data format using the Data Service Studio. For details, see the *User's Guide*.
2. On your workstation, create a new program. The following shows a sample Scala program:

```
import scala.math.random
import org.apache.spark.{SparkConf, SparkContext}
import org.apache.spark.sql.SQLContext
import org.apache.spark.sql.functions._

object YourClassName {
  def main(args: Array[String]) {
    val conf = new SparkConf().setAppName("Your App Name")
    val spark = new SparkContext(conf)
    val sqlContext = new SQLContext(spark)
    import sqlContext.implicits._

    //configure JDBC data source
    val dataServiceDataFrame = sqlContext.load("jdbc",
      Map( "driver" -> "com.rs.jdbc.dv.DvDriver", "url" ->
        "jdbc:rs:dv://hostname:1200; DBTY=DVS;PWD=XXXX;
        UID=XXX;MRC=true;MRCC=10;LogConfiguration=
        file:///opt/log4j2.xml", "dbtable" ->
        "SCHEMANAME.VIRTUALTABLERNAME"));
  }
}
```

3. Create the SBT folder structure and add your source files, as follows:

- a) On your workstation, create the required SBT folder structure that will contain your source files as follows: **ProjectName** > **src** > **main** > **scala**. For example: `YourClassName.scala`
 - b) Create a `dependencies lib` folder and add program dependency files, such as JDBC drivers, to the folder. For example: **ProjectName** > **lib**
 - c) Create your `ProjectName.sbt` file in the base directory folder, and then rename accordingly. For example:
 - name: "Project Name" for example, `SbtSamp1`
 - version: 1.0
 - scalaVersion: 2.10.4
 - libraryDependencies : "org.apache.spark" %% "spark-core" % "1.5.2"
 - libraryDependencies: "org.apache.spark" %% "spark-sql" % "1.5.2"
4. Create the program JAR file and submit the program to Spark, as follows:
- a) Create the JAR file in your target folder by running the following command in your base directory:
`sbt package.`
 - b) Copy the JAR files to a location on the mainframe.
 - c) In your Spark folder, use a Spark submit command to submit the program. For example:
`bin/spark-submit --class "YourClassName" --jars DSDriverLocation/dv-jdbc-3.1.jar target/scala-2.10/Project-Name_2.10-1.0.jar`
- If your program run successfully, the first 100 rows of data displays.
- Note:** If you choose to copy the URL, note that the URL text in this example might include wrapping to fit the displayed format.

Chapter 5. Accessing data from Python applications

Access z/OS data from Python applications using Mainframe Data Service.

Before you begin

1. Verify that all required Data Service components are installed and configured. For details about installing and configuring Data Service, see the *Installation and Customization Guide*.
2. Verify that the IzODA Anaconda component has been installed on the host and your Unix Systems Services environment has been configured.
3. Specify the location of the Data Service load library:

```
# this can be added to the appropriate profile script
export STEPLIB=hlq.SAZKLOAD
```

About this task

Accessing z/OS data from Python applications using Data Service provides data scientists with a comprehensive solution for bringing the compute to the data.

Accessing z/OS data from Python applications requires the IzODA Anaconda component and use of the Python dsdbc module. The Python dsdbc module is an implementation of the Python Database API Specification v2.0 ([Python Database API Specification v2.0 \(www.python.org/dev/peps/pep-0249/\)](http://www.python.org/dev/peps/pep-0249/)) which enables Python developers on z/OS to use a simple, portable framework to access the Data Service. The dsdbc module is available with the IzODA Anaconda component.

Note: When using the IzODA Anaconda component, Python applications do not use the JDBC driver for data access; the non-Java based Python dsdbc module is used for data access.

To access z/OS data from Python applications, you must use the Data Service Studio to create virtual tables to expose the data over SQL.

Use the following procedure to create a Python DB application to access your z/OS data. Using the procedure, you will access your mainframe data by issuing SQL calls using the Python DB API.

Procedure

1. Create virtual tables to map your data to a relational data format using the Data Service Studio. For details, see the *User's Guide*.
2. On your workstation, create a text file on the host in USS containing the Python code. In this example, the text file is named `sample.py`. In order for the Python interpreter to be able to read the file, ensure that the file is tagged with the appropriate character set. For example, if the file contains EBCDIC-encoded data, then you can use the `chtag` command as follows:

```
$ chtag -tc IBM-1047 sample.py
```

Sample code:

```
conn = dsdbc.connect(SSID="AZKS")
cursor = conn.cursor()
cursor.execute("SELECT * FROM virtual_table")
rows = cursor.fetchall()
print("#rows: {}".format(len(rows)))
cols = ''
for col in cursor.description: cols += "{} ".format(col[0])
print("columns: {}".format(cols))
print("first row: {}".format(rows[0]))
```

3. Execute the code as follows:

```
$ python sample.py
```

Upon successful completion, the application displays the number of rows, the column names, and the first row of data values.

Appendix A. DFStor support

Mainframe Data Service supports DFStor capabilities by enabling users to insert data from a data frame into the target of your choice.

For example, if you have an IBM DB2 Analytics Accelerator that contains critical business data and you are using Spark to perform real-time analytics, Data Service enables you to create a union from data on the accelerator with analytical data on Spark to create invaluable new insights.

By creating a virtual table for both sources in the Data Service and creating a union of both sets of data, you can easily combine data from both Spark and other data sources to achieve greater insights. For procedures about defining your target table for DFStor to DB2 and to IBM DB2 Analytics Accelerator, refer to the IBM Knowledge Center.

Appendix B. DVS_MAP API

Use the Data Service DVS_MAP API to call the Data Mapping Facility (DMF) to map proprietary data as well as sample maps.

Note: The Data Service Studio uses the DVS_MAP API to access the DMF. It is recommended that you use the Data Service Studio to map any data supported by the Data Service server, and that you use the DVS_MAP API only when mapping proprietary data.

Topics:

- [“Syntax” on page 121](#)
- [“XML schema for DMF ” on page 123](#)
- [“Example: SAMPLE-REC” on page 142](#)
- [“Example: STAFF-RECORD” on page 144](#)

Syntax

Use the following syntax when calling the DVS_MAP API.

Call statement

```
DVS_MAP('DMF','function','parm3','parm4','parm5')
```

Parameter	Description	Values	Required
DMF	Indicates call to Data Mapping Facility (DMF)	DMF	Yes
<i>function</i>	Indicates DMF function to perform	EXPORT IMPORT DELETE REFRESH	Yes
<i>parm3</i>	Identifies map to process	For EXPORT, IMPORT, and DELETE: <i>Map name/Map XML</i> For REFRESH: <u>AZKMAPP</u> AZKHTXLB	For EXPORT, IMPORT, and DELETE, this value is required. For REFRESH, this value is optional and defaults to AZKMAPP.

Parameter	Description	Values	Required
<i>parm4</i>	Specifies save options for IMPORT and refresh options for DELETE	For IMPORT: <u>NOSAVE</u> Use to validate XML <u>SAVE</u> Use to create a new map <u>REPLACE</u> Use to replace an existing map For DELETE: <u>NOREFRESH</u> Do not do a DMF refresh <u>REFRESH</u> Do a DMF refresh	For Import, this value is optional and defaults to NOSAVE. For Delete, this value is optional and defaults to NOREFRESH.
<i>parm5</i>	Specifies refresh options for IMPORT	For IMPORT: <u>NOREFRESH</u> Do not do a DMF refresh <u>REFRESH</u> Do a DMF refresh	No

Call examples

• EXPORT

```
DVS_MAP('DMF','EXPORT','MAPNAME')
```

This statement will export a map to a result set with one column and one row for each line of XML.

• IMPORT

```
DVS_MAP('DMF','IMPORT','MAP XML','SAVE','REFRESH')
```

This statement will import a map from an XML document, save the map, and perform a DMF refresh.

```
DVS_MAP('DMF','IMPORT','MAP XML','REPLACE','NOREFRESH')
```

This statement will import a map from an XML document, replace an existing map, and not perform a DMF refresh

• DELETE

```
DVS_MAP('DMF','DELETE','MAPNAME','NOREFRESH')
```

This statement will delete a map and will not perform a DMF refresh.

• REFRESH

```
DVS_MAP('DMF','REFRESH','AZKMAPP')
```

This statement will perform a DMF refresh to the AZKMAPP data set.

Best practices

The following are recommended best practices:

- Use *parm4* as NOSAVE and *parm5* as NOREFESH to have the server validate the XML without saving a copy of the Virtual Table.

- When importing a new map, always set *parm4* as SAVE and *parm5* as REFRESH.

XML schema for DMF

The following XML defines the XML schema for the Data Mapping Facility (DMF) component. All DMF XML metadata (maps) must adhere to this schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- *****
* Copyright Rocket Software, Inc. 1991 - 2017 All Rights Reserved. *
* This defines the XML schema for the Data Mapping Facility (DMF) *
* component. *
* All DMF XML metadata (aka maps) must adhere to this schema. *
***** -->
<xsd:schema xmlns:tns="urn:Server-Metadata-Schema" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  attributeFormDefault="unqualified"
  elementFormDefault="qualified" targetNamespace="urn:Server-Metadata-Schema"
  xml:lang="en">
  <xsd:annotation>
    <xsd:documentation source="http://www.rocketsoftware.com">
      XML schema for the Data Mapping Facility (DMF) component.
      All DMF XML metadata (aka maps) must adhere to this schema.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="http://www.w3.org/2001/xml.xsd"/>
  <xsd:element name="NEONDMF" type="tns:ElemNEONDMF"/>
  <xsd:complexType name="ElemNEONDMF">
    <xsd:sequence>
      <xsd:element minOccurs="0" name="PROLOG" type="tns:ElemPROLOG"/>
      <xsd:group maxOccurs="unbounded" ref="tns:MapDefGroup"/>
    </xsd:sequence>
    <xsd:attribute fixed="2.0" name="version" type="xsd:string" use="required"/>
  </xsd:complexType>
  <xsd:group name="MapDefGroup">
    <xsd:choice>
      <xsd:element name="LPTABLE" type="tns:ElemLPTABLE"/>
      <xsd:element name="PROCEDURE" type="tns:ElemPROCEDURE"/>
      <xsd:element name="MAP" type="tns:ElemMAP"/>
      <xsd:element name="WEBSERVICE" type="tns:ElemWEBSERVICE"/>
      <xsd:element name="VIRTUALDIRECTORY" type="tns:ElemVirtualDirectory"/>
      <xsd:element name="LIBRARY" type="tns:ElemLibrary"/>
      <xsd:element name="WSSYSTEM" type="tns:ElemWSSystem"/>
      <xsd:element name="WSSQL">
        <xsd:complexType>
          <xsd:complexContent>
            <xsd:extension base="tns:ElemWSSQL"/>
          </xsd:complexContent>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="PUBLISH" type="tns:ElemPUBLISH"/>
      <xsd:element name="BROKER" type="tns:ElemBROKER"/>
      <xsd:element name="IMSPSB" type="tns:ElemIMSPSB"/>
      <xsd:element name="IMSDBD" type="tns:ElemIMSDBD"/>
    </xsd:choice>
  </xsd:group>
  <xsd:complexType name="MapDefBaseType">
    <xsd:sequence>
      <xsd:element minOccurs="0" name="NOTATIONS" type="tns:ElemNOTATIONS"/>
    </xsd:sequence>
    <xsd:attribute name="name" type="tns:nbLongExternalMAPName" use="required"/>
    <xsd:attribute default="COBOL" name="syntax" type="tns:enumMapdefSyntax" use="optional"/>
    <xsd:attribute fixed="false" name="reload" type="xsd:string" use="optional"/>
    <xsd:attribute default="enabled" name="status" type="tns:enumGeneralMapStatus" use="optional"/>
  </xsd:complexType>
  <xsd:complexType name="ElemLPTABLE">
    <xsd:complexContent>
      <xsd:extension base="tns:MapDefBaseType">
        <xsd:sequence>
          <xsd:element maxOccurs="unbounded" name="LPAGE" type="tns:ElemLPAGE"/>
        </xsd:sequence>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="ElemLPAGE">
    <xsd:complexContent>
      <xsd:extension base="xsd:anyType">
        <xsd:attribute name="name" type="tns:nbMFSName" use="required"/>
        <xsd:attribute name="field" type="tns:optMFLDName" use="optional"/>
        <xsd:attribute name="offset" type="xsd:nonNegativeInteger" use="required"/>
        <xsd:attribute name="cond" type="tns:enumLPAGECond" use="required"/>
        <xsd:attribute name="length" type="xsd:nonNegativeInteger" use="required"/>
        <xsd:attribute default="string" name="type" type="tns:enumLiteralType" use="optional"/>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
  <xsd:complexType name="ElemPROLOG">
    <xsd:sequence>
```

```

<xsd:any maxOccurs="unbounded" minOccurs="0" namespace="##other" processContents="lax"/>
</xsd:sequence>
<xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="ElemPROCEDURE">
<xsd:complexContent>
<xsd:extension base="tns:MapDefBaseType">
<xsd:sequence>
<xsd:choice>
<xsd:element name="IMSTM" type="tns:ElemIMSTM"/>
<xsd:element name="CICSTM" type="tns:ElemCICSTM"/>
<xsd:element name="CICSEX" type="tns:ElemCICSEX"/>
</xsd:choice>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ProcedureIOBaseType">
<xsd:sequence>
<xsd:element minOccurs="0" name="IN" type="tns:ElemProcIN"/>
<xsd:element minOccurs="0" name="OUT" type="tns:ElemProcOUT"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ElemCICSTM">
<xsd:complexContent>
<xsd:extension base="tns:ProcedureIOBaseType">
<xsd:attribute name="transact" type="tns:optCICSxxTransact" use="required"/>
<xsd:attribute name="program" type="tns:optCICSxxProgram" use="required"/>
<xsd:attribute name="connect" type="tns:optCICSxxConnect" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemCICSEX">
<xsd:complexContent>
<xsd:extension base="tns:ProcedureIOBaseType">
<xsd:attribute name="transact" type="tns:optCICSxxTransact" use="required"/>
<xsd:attribute name="program" type="tns:optCICSxxProgram" use="required"/>
<xsd:attribute name="connect" type="tns:optCICSxxConnect" use="required"/>
<xsd:attribute name="dyntransid" type="tns:enumCICSTransId" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemIMSTM">
<xsd:complexContent>
<xsd:extension base="tns:ProcedureIOBaseType">
<xsd:attribute default="false" name="imspage" type="xsd:boolean" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemProcIN">
<xsd:attribute name="mapname" type="tns:nbExternalMAPName" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemProcOUT">
<xsd:attribute name="mapname" type="tns:nbExternalMAPName" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemWEBSERVICE">
<xsd:complexContent>
<xsd:extension base="tns:MapDefBaseType">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="RESTRICT" type="tns:ElemRESTRICT"/>
<xsd:element maxOccurs="unbounded" name="OPERATION" type="tns:ElemOPERATION"/>
</xsd:sequence>
<xsd:attribute name="servicename" type="tns:nbSERVICENAME" use="required"/>
<xsd:attribute name="targetnamespace" type="tns:nbSERVICENamespace" use="required"/>
<xsd:attribute name="directory" type="tns:nbLongExternalMAPName" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemRESTRICT">
<xsd:attribute name="ip" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="15"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemOPERATION">
<xsd:sequence>
<xsd:element maxOccurs="2" minOccurs="0" name="MSGMap" type="tns:ElemMSGMap"/>
</xsd:sequence>
<xsd:attribute name="name" type="tns:nbOPERATIONName" use="required"/>
<xsd:attribute name="provider" type="tns:nbLongExternalMAPName" use="required"/>
<xsd:attribute name="target" type="tns:BackendName" use="required"/>
<xsd:attribute default="None" name="security" type="tns:enumPROVIDERSecurity" use="optional"/>
<xsd:attribute default="No" name="format" type="tns:enumPROVIDERFormat" use="optional"/>
<xsd:attribute default="Integer" name="numeric" type="tns:enumPROVIDERNumeric" use="optional"/>
<xsd:attribute name="description" use="optional">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="128"/>
</xsd:restriction>
</xsd:simpleType>

```

```

</xsd:attribute>
<xsd:attribute name="type" type="tns:enumPROVIDERType" use="required"/>
<xsd:attribute default="request-response" name="transmissiontype" type="tns:enumTransmissionType" use="optional"/>
<xsd:attribute default="false" name="ia" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemMAP">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:sequence>
        <xsd:choice minOccurs="0">
          <xsd:element name="MSG" type="tns:ElemMSG"/>
          <xsd:element name="VSAM" type="tns:ElemVSAM"/>
          <xsd:element name="ADABAS" type="tns:ElemADABAS"/>
          <xsd:element name="SEQUENTIAL" type="tns:ElemSEQUENTIAL"/>
        </xsd:choice>
        <xsd:element maxOccurs="unbounded" name="FLD" type="tns:ElemFLD"/>
      </xsd:sequence>
      <xsd:attribute name="defines" type="tns:enumMAPDefines" use="required"/>
      <xsd:attribute name="directory" type="tns:nblongExternalMAPName" use="optional"/>
      <xsd:attribute name="member" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="8"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="cssid" type="xsd:int" use="optional"/>
      <xsd:attribute name="conversionTechnique" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="8"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="mrthreadcount" type="tns:mapReduceThreadCount" use="optional"/>
      <xsd:attribute default="false" name="suppressmr" type="xsd:boolean" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemVirtualDirectory">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:attribute name="directory" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="256"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="mflowdsname" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="44"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="mapdsname" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="44"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="rulesetname" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="8"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="requestqueue" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="48"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="replytoqueue" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="48"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="errorqueue" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="48"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="requestqueuemanager" use="optional">

```

```

<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:maxLength value="48"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="soapactionlocation" use="optional">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="8"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<xsd:attribute default="false" name="read" type="xsd:boolean" use="optional"/>
<xsd:attribute default="false" name="write" type="xsd:boolean" use="optional"/>
<xsd:attribute default="false" name="execute" type="xsd:boolean" use="optional"/>
<xsd:attribute default="false" name="internalsecurity" type="xsd:boolean" use="optional"/>
<xsd:attribute default="UseridPassword" name="externalsecurity" type="tns:enumUserPass" use="optional"/>
<xsd:attribute default="false" name="sourceonly" type="xsd:boolean" use="optional"/>
<xsd:attribute default="false" name="transactionalqueue" type="xsd:boolean" use="optional"/>
<xsd:attribute default="wmq" name="payloadtype" type="tns:enumPayloadType" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemLibrary">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:attribute name="dsname" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="44"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="dbid" type="xsd:integer" use="optional"/>
      <xsd:attribute name="filenum" type="xsd:integer" use="optional"/>
      <xsd:attribute name="natlib" type="xsd:string" use="optional"/>
      <xsd:attribute name="servicetype" type="tns:enumBrokerServerType" use="optional"/>
      <xsd:attribute name="description" type="xsd:string" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemBROKER">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:choice maxOccurs="1" minOccurs="1">
        <xsd:element name="CICS" type="tns:ElemBrokerCICS"/>
        <xsd:element name="STC" type="tns:ElemBrokerSTC"/>
        <xsd:element name="BATCH" type="tns:ElemBrokerBATCH"/>
      </xsd:choice>
      <xsd:attribute name="servicename" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="serviceclass" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="serverservicename" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="servertype" type="tns:enumBrokerServerType" use="required">
      </xsd:attribute>
      <xsd:attribute default="NO" name="persistent" type="tns:enumYesNo" use="optional">
      </xsd:attribute>
      <xsd:attribute default="NO" name="secure" type="tns:enumYesNo" use="optional"/>
      <xsd:attribute default="NO" name="securetouserid" type="tns:enumYesNo" use="optional"/>
      <xsd:attribute name="maxservers" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="1"/>
            <xsd:maxInclusive value="9999"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute default="0" name="maxreceives" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="0"/>

```

```

        <xsd:maxInclusive value="99999"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="clienttimer" type="xsd:integer" use="optional">
</xsd:attribute>
<xsd:attribute name="servertime" type="xsd:integer" use="optional">
</xsd:attribute>
<xsd:attribute name="uowmaxbuffer" use="optional">
    <xsd:simpleType>
        <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="0"/>
            <xsd:maxInclusive value="32"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="waitforserver" type="xsd:integer" use="optional">
</xsd:attribute>
<xsd:attribute default="NO" name="uowparticipant" type="tns:enumYesNo" use="optional">
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemBrokerCICS">
    <xsd:attribute name="mirrortran" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="4"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="conn" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="4"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="tran" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="4"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute default="0" name="maxexecute" type="xsd:short" use="optional"/>
    <xsd:attribute name="xctlpgm" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="8"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
    <xsd:attribute name="commarea" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="250"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemBrokerSTC">
    <xsd:attribute name="cmdstring" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="44"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemBrokerBATCH">
    <xsd:attribute name="jcllibrary" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="44"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemIMSPSB">
    <xsd:complexContent>
        <xsd:extension base="tns:MapDefBaseType">
            <xsd:sequence>
                <xsd:element maxOccurs="1" name="PCB" type="tns:ElemIMSPSBPCB"/>
            </xsd:sequence>
            <xsd:attribute name="psbname" use="required">

```

```

    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemIMSPSBPCB">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="SENSEG" type="tns:ElemIMSPSBPCBSENSEG"/>
  </xsd:sequence>
  <xsd:attribute name="name" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="dbdname" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="procopt" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="keylen" type="xsd:integer" use="required">
  </xsd:attribute>
  <xsd:attribute default="SINGLE" name="pos" type="tns:enumPCBPos" use="optional"/>
  <xsd:attribute name="procseq" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemIMSPSBPCBSENSEG">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="SENFLD" type="tns:ElemIMSSENFLD"/>
  </xsd:sequence>
  <xsd:attribute name="name" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="parent" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="8"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="procopt" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="ssptr" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="indices" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>

```

```

</xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemIMSDBD">
<xsd:complexContent>
<xsd:extension base="tns:MapDefBaseType">
<xsd:sequence>
<xsd:element maxOccurs="1" name="SEGM" type="tns:ElemIMSDBDSEGM"/>
</xsd:sequence>
<xsd:attribute name="dbdname" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:minLength value="1"/>
<xsd:maxLength value="8"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="access" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:minLength value="1"/>
<xsd:maxLength value="8"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemIMSDBDSEGM">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" name="FLD" type="tns:ElemIMSFLD"/>
</xsd:sequence>
<xsd:attribute name="name" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:minLength value="1"/>
<xsd:maxLength value="50"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="parent" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:minLength value="1"/>
<xsd:maxLength value="50"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="length" type="xsd:integer" use="required">
</xsd:attribute>
<xsd:attribute name="level" type="xsd:integer" use="required">
</xsd:attribute>
<xsd:attribute name="sequence" type="xsd:integer" use="required">
</xsd:attribute>
</xsd:complexType>
<xsd:complexType name="ElemWSCLIENT">
<xsd:complexContent>
<xsd:extension base="tns:MapDefBaseType">
<xsd:attribute name="directory" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:minLength value="1"/>
<xsd:maxLength value="50"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="template" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="8"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="copybook" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="8"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="service" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="128"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="url" use="required">
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:maxLength value="128"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:attribute>

```

```

<xsd:attribute name="referral" use="optional">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="44"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="format" use="optional">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="44"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
<xsd:attribute name="exit" use="optional">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="44"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:attribute>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemWSSystem">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:attribute name="system" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="8"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="type" type="tns:enumTARGETType" use="required"/>
      <xsd:attribute name="linktype" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="SHADOW"/>
            <xsd:enumeration value="ZCI"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="connection" use="optional">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="8"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemWSSQL">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:sequence>
        <xsd:element name="SQL" type="xsd:string"/>
      </xsd:sequence>
      <xsd:attribute name="sqlname" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="50"/>
            <xsd:whiteSpace value="collapse"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ElemMSG">
  <xsd:attribute name="type" type="tns:enumMSGType" use="required"/>
  <xsd:attribute name="difdof" type="tns:optMFSName" use="required"/>
  <xsd:attribute name="next" type="tns:optMFSName" use="required"/>
  <xsd:attribute default="false" name="sca" type="xsd:boolean" use="optional"/>
  <xsd:attribute name="opt" type="tns:integerMSGOpt" use="optional"/>
  <xsd:attribute default="false" name="page" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemVSAM">
  <xsd:sequence>
    <xsd:element name="BASEPATH" type="tns:ElemBASEPATH"/>
    <xsd:element maxOccurs="8" minOccurs="0" name="AIXPATH" type="tns:ElemAIXPATH"/>
  </xsd:sequence>
  <xsd:attribute name="listcat" type="xsd:boolean" use="required"/>
  <xsd:attribute name="aix" type="xsd:boolean" use="required"/>
  <xsd:attribute name="cicstran" type="tns:optVSAMCicstran" use="required"/>
  <xsd:attribute name="cicsconn" type="tns:optVSAMCicsconn" use="required"/>
  <xsd:attribute name="postread" type="tns:optExitName" use="optional"/>
  <xsd:attribute name="prewrite" type="tns:optExitName" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemBASEPATH">
  <xsd:attribute name="ddname" type="tns:optDDname" use="required"/>
  <xsd:attribute name="dsname" type="tns:optDSname" use="required"/>

```

```

<xsd:attribute name="keyoff" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="keylen" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="reclen" type="xsd:positiveInteger" use="required"/>
<xsd:attribute name="recfmt" type="tns:enumVSAMRecfmt" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemAIXPATH">
<xsd:attribute name="ddname" type="tns:optDDName" use="required"/>
<xsd:attribute name="dsname" type="tns:optDSName" use="required"/>
<xsd:attribute name="keyoff" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="keylen" type="xsd:nonNegativeInteger" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemADABAS">
<xsd:attribute name="dbid" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="filenum" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="database" type="tns:AdabasDatabase" use="required"/>
<xsd:attribute name="filename" type="tns:AdabasFilename" use="required"/>
<xsd:attribute name="subsys" type="tns:AdabasSubsys" use="required"/>
<xsd:attribute name="password" type="tns:AdabasPassword" use="optional"/>
<xsd:attribute default="YES" name="passencrypt" type="tns:enumYesNo" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemSEQUENTIAL">
<xsd:attribute name="ddname" type="tns:optDDName" use="optional"/>
<xsd:attribute name="dsname" type="tns:optDSName" use="required"/>
<xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
<xsd:attribute name="type" type="tns:enumSequentialType" use="optional"/>
<xsd:attribute name="postread" type="tns:optExitName" use="optional"/>
<xsd:attribute name="prewrite" type="tns:optExitName" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemFLD">
<xsd:sequence>
<xsd:element minOccurs="0" name="MFLD" type="tns:ElemMFLD"/>
<xsd:element minOccurs="0" name="LITR" type="tns:ElemLITR"/>
<xsd:element minOccurs="0" name="SEQINFO" type="tns:ElemSEQINFO"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="SOURCE" type="tns:ElemSOURCE"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="FLDOPTION" type="tns:ElemFLDOPTION"/>
<xsd:any maxOccurs="unbounded" minOccurs="0" namespace="##other" processContents="lax"/>
</xsd:sequence>
<xsd:attribute name="name" type="tns:nbFLDName" use="required"/>
<xsd:attribute name="colname" type="tns:nbFLDColname" use="required"/>
<xsd:attribute name="enabled" type="xsd:boolean" use="optional"/>
<xsd:attribute default="1" name="level" type="tns:integerFLDLevel" use="optional"/>
<xsd:attribute name="sqltype" type="tns:enumFLDSqltype" use="required"/>
<xsd:attribute name="format" type="tns:enumFLDFormat" use="required"/>
<xsd:attribute name="fill" type="tns:nbFLDFill" use="required"/>
<xsd:attribute name="offset" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="length" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="precision" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="scale" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute default="0" name="occurs" type="xsd:nonNegativeInteger" use="optional"/>
<xsd:attribute default="false" name="odo" type="xsd:boolean" use="optional"/>
<xsd:attribute default="0" name="odoseq" type="xsd:integer" use="optional"/>
<xsd:attribute default="0" name="sequence" type="xsd:integer" use="optional"/>
<xsd:attribute name="picture" type="xsd:string" use="optional"/>
<xsd:attribute name="value" type="xsd:string" use="optional"/>
<xsd:attribute name="signed" type="xsd:boolean" use="optional"/>
<xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="ElemIMSFLD">
<xsd:sequence>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="FLDOPTION" type="tns:ElemSEGMSFLDOPTION"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="SOURCE" type="tns:ElemSOURCE"/>
<xsd:any maxOccurs="unbounded" minOccurs="0" namespace="##other" processContents="lax"/>
</xsd:sequence>
<xsd:attribute name="name" type="tns:nbFLDName" use="required"/>
<xsd:attribute name="colname" type="tns:nbFLDColname" use="required"/>
<xsd:attribute name="enabled" type="xsd:boolean" use="optional"/>
<xsd:attribute default="1" name="level" type="tns:integerFLDLevel" use="optional"/>
<xsd:attribute name="sqltype" type="tns:enumFLDSqltype" use="required"/>
<xsd:attribute name="format" type="tns:enumFLDFormat" use="required"/>
<xsd:attribute name="fill" type="tns:nbFLDFill" use="required"/>
<xsd:attribute name="offset" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="length" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="precision" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="scale" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute default="0" name="occurs" type="xsd:nonNegativeInteger" use="optional"/>
<xsd:attribute default="false" name="odo" type="xsd:boolean" use="optional"/>
<xsd:attribute default="0" name="odoseq" type="xsd:integer" use="optional"/>
<xsd:attribute default="0" name="sequence" type="xsd:integer" use="optional"/>
<xsd:attribute name="picture" type="xsd:string" use="optional"/>
<xsd:attribute name="value" type="xsd:string" use="optional"/>
<xsd:attribute name="signed" type="xsd:boolean" use="optional"/>
<xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="ElemMFLD">
<xsd:attribute name="segment" type="xsd:positiveInteger" use="required"/>
<xsd:attribute name="dynatts" type="xsd:nonNegativeInteger" use="required"/>
<xsd:attribute name="attrs" type="xsd:NMTOKENS" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemIMSSENFLD">
<xsd:attribute name="name" type="xsd:string" use="required"/>
<xsd:attribute name="start" type="xsd:integer" use="required"/>
<xsd:attribute default="YES" name="replace" type="tns:enumYesNo" use="optional"/>
</xsd:complexType>

```

```

<xsd:complexType name="ElemLITR">
  <xsd:simpleContent>
    <xsd:extension base="tns:preservedString">
      <xsd:attribute default="string" name="type" type="tns:enumLiteralType" use="optional"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="ElemSEQINFO">
  <xsd:attribute name="pseudocolumn" type="tns:enumPseudoColumn" use="required"/>
  <xsd:attribute default="false" name="search" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemFLDOPTION">
  <xsd:attribute default="NO" name="descriptor" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="fixedformat" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="mufield" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="nullsupp" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="pefield" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="phonparent" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="superparent" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="uniquede" type="tns:enumYesNo" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemSEGFLDOPTION">
  <xsd:attribute default="NO" name="imssearch" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="lchild" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute default="NO" name="vkey" type="tns:enumYesNo" use="optional"/>
  <xsd:attribute name="frommap" type="xsd:string" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemPUBLISH">
  <xsd:complexContent>
    <xsd:extension base="tns:MapDefBaseType">
      <xsd:sequence>
        <xsd:choice>
          <xsd:element name="SOURCE" type="tns:PubSOURCE"/>
          <xsd:element name="DESTINATION" type="tns:PubDESTINATION"/>
          <xsd:element name="EVENT" type="tns:PubEVENT"/>
          <xsd:element name="RULEGROUP" type="tns:PubRULEGROUP"/>
        </xsd:choice>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name="ElemSOURCE">
  <xsd:restriction base="tns:preservedString">
    <xsd:maxLength value="80"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="ElemNOTATIONS">
  <xsd:sequence>
    <xsd:element minOccurs="0" name="RESIDENCE" type="tns:ElemRESIDENCE"/>
    <xsd:group maxOccurs="5" minOccurs="0" ref="tns:AuditTrailGroup"/>
    <xsd:element minOccurs="0" name="DOCUMENTATION" type="tns:DocumentationBaseType"/>
    <xsd:any maxOccurs="unbounded" minOccurs="0" namespace="##other" processContents="lax"/>
  </xsd:sequence>
  <xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="ElemRESIDENCE">
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute name="lastmod" type="xsd:dateTime" use="optional"/>
  <xsd:attribute name="dsname" type="tns:optDSname" use="optional"/>
  <xsd:attribute name="user" type="tns:optRACFUserName" use="optional"/>
</xsd:complexType>
<xsd:group name="AuditTrailGroup">
  <xsd:choice>
    <xsd:element name="EXTRACT" type="tns:AuditTrailBaseType"/>
    <xsd:element name="EXPORT" type="tns:AuditTrailBaseType"/>
    <xsd:element name="IMPORT" type="tns:AuditTrailBaseType"/>
  </xsd:choice>
</xsd:group>
<xsd:complexType name="AuditTrailBaseType">
  <xsd:sequence>
    <xsd:element minOccurs="0" name="AGENT" type="tns:ElemAGENT"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="INPUT" type="tns:ElemINPUT"/>
    <xsd:any namespace="##other" processContents="lax"/>
  </xsd:sequence>
  <xsd:attribute name="date" type="xsd:dateTime" use="optional"/>
  <xsd:attribute name="user" type="tns:optRACFUserName" use="optional"/>
</xsd:complexType>
<xsd:complexType name="ElemAGENT">
  <xsd:attribute name="name" type="xsd:NMTOKEN" use="required"/>
  <xsd:attribute name="version" type="tns:optAGENTVersion" use="optional"/>
  <xsd:attribute name="level" type="xsd:dateTime" use="optional"/>
  <xsd:attribute name="location" type="tns:optAGENTLocation" use="optional"/>
  <xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="ElemINPUT">
  <xsd:attribute name="from" type="tns:optINPUTFrom" use="optional"/>
  <xsd:attribute name="lastmod" type="xsd:dateTime" use="optional"/>
  <xsd:attribute name="user" type="tns:optRACFUserName" use="optional"/>
  <xsd:anyAttribute namespace="##other" processContents="lax"/>
</xsd:complexType>
<xsd:complexType name="PubSOURCE">
  <xsd:sequence>
    <xsd:choice>

```

```

<xsd:element name="ADABAS" type="tns:PubSourceAdabas"/>
<xsd:element name="DB2" type="tns:PubSourceDB2"/>
<xsd:element name="CICS" type="tns:PubSourceCICS"/>
<xsd:element name="IDMS" type="tns:PubSourceIDMS"/>
<xsd:element name="IMS" type="tns:PubSourceIMS"/>
<xsd:element name="VSAM" type="tns:PubSourceVSAM"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="name" type="tns:PubNAME" use="required"/>
<xsd:attribute name="description" type="tns:PubDesc" use="optional"/>
<xsd:attribute default="300" name="maxwait" type="tns:PubTime" use="optional"/>
<xsd:attribute default="50" name="queue" type="tns:PubQueue" use="optional"/>
<xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
<xsd:attribute default="false" name="fastpath" type="xsd:boolean" use="optional"/>
<xsd:attribute default="true" name="forcebackout" type="xsd:boolean" use="optional"/>
<xsd:attribute default="false" name="director" type="xsd:boolean" use="optional"/>
<xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
<xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
<xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
<xsd:attribute default="1047" name="CCSID" type="xsd:integer" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubSourceAdabas">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
<xsd:attribute name="connection" type="tns:PubSourceConnection" use="required"/>
<xsd:attribute name="capture" type="tns:PubAdabasType" use="optional" default="Server"/>
</xsd:complexType>
<xsd:complexType name="PubSourceDB2">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
<xsd:attribute name="qualifier" type="tns:nbPlan" use="required"/>
<xsd:attribute name="connection" type="tns:PubSourceConnection" use="required"/>
<xsd:attribute name="capture" type="tns:PubDB2Type" use="optional" default="SQL"/>
</xsd:complexType>
<xsd:complexType name="PubSourceCICS">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
<xsd:attribute name="applid" type="tns:USERID" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubSourceIDMS">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
<xsd:attribute name="cnumber" type="tns:IDMSCVNumber" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubSourceIMS">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
<xsd:attribute name="connection" type="tns:PubSourceConnection" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubSourceVSAM">
<xsd:attribute name="userid" type="tns:USERID" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubAppResp">
<xsd:attribute name="mqreplsystem" type="xsd:string"/>
<xsd:attribute name="mqreplyqname" type="xsd:string"/>
<xsd:attribute name="mqreplytimeout" type="tns:PubTime"/>
<xsd:attribute default="120" name="mqreplscan" type="xsd:positiveInteger" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubDESTINATION">
<xsd:choice>
<xsd:element name="TCPIP" type="tns:ElemDestTCPIP"/>
<xsd:element name="MQSERIES" type="tns:PubDestMQ"/>
<xsd:element name="MQBROKER" type="tns:ElemDestMQBroker"/>
<xsd:element name="TIBCO" type="tns:PubDestRV"/>
<xsd:element name="GROUP" type="tns:PubGroup"/>
</xsd:choice>
<xsd:attribute name="name" type="tns:PubNAME" use="required"/>
<xsd:attribute default="60" name="retryinterval" type="tns:PubTime" use="optional"/>
<xsd:attribute default="50" name="queue" type="tns:PubQueue" use="optional"/>
<xsd:attribute name="machine" type="xsd:string" use="optional"/>
<xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
<xsd:attribute name="type" type="tns:enumPubDestType" use="optional"/>
<xsd:attribute default="false" name="post" type="xsd:boolean" use="optional"/>
<xsd:attribute name="description" type="tns:PubDesc" use="optional"/>
<xsd:attribute name="dbuserid" type="tns:USERID" use="optional"/>
<xsd:attribute name="opt" type="tns:enumOpt" use="optional"/>
<xsd:attribute default="437" name="CCSID" type="xsd:integer" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubGroup">
<xsd:sequence>
<xsd:element maxOccurs="16" name="LIST" type="tns:PubList"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="PubRULEGROUP">
<xsd:sequence>
<xsd:element maxOccurs="13" name="LIST" type="tns:PubRULE"/>
</xsd:sequence>
<xsd:attribute name="name" type="tns:PubNAME" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubMONITOR">
<xsd:sequence>
<xsd:element maxOccurs="13" name="LIST" type="tns:PubRULE"/>
</xsd:sequence>
<xsd:attribute fixed="JCA1.5" name="type" type="xsd:string" use="required"/>
<xsd:attribute name="source" type="tns:PubNAME" use="required"/>
<xsd:attribute name="table" type="tns:TableName" use="required"/>
<xsd:attribute name="machineid" type="tns:PubMachineID" use="required"/>
<xsd:attribute name="service" type="tns:PubJCAService" use="required"/>

```

```

<xsd:attribute name="options" type="tns:PubJCAOptions" use="optional"/>
<xsd:attribute name="event" type="tns:enumUpdateType" use="required"/>
<xsd:attribute name="dest" type="tns:PubNAME" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubList">
  <xsd:attribute name="name" type="tns:PubNAME" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemDestTCPIP">
  <xsd:sequence>
    <xsd:element maxOccurs="10" minOccurs="0" name="EXTRAHEADERS" type="xsd:string"/>
  </xsd:sequence>
  <xsd:attribute name="url" type="tns:optSOURCEUri" use="required"/>
  <xsd:attribute name="ssl" type="xsd:boolean" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubDestMQ">
  <xsd:sequence>
    <xsd:element minOccurs="0" name="APPLICATIONRESPONSEEXPECTED" type="tns:PubAppResp"/>
  </xsd:sequence>
  <xsd:attribute name="qname" use="required"/>
  <xsd:attribute name="mqsystem" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubDestRV">
  <xsd:attribute name="daemon" type="tns:PubRVname" use="required"/>
  <xsd:attribute name="service" type="tns:PubRVname" use="required"/>
  <xsd:attribute name="network" type="tns:PubRVname" use="optional"/>
  <xsd:attribute name="subject" type="tns:PubRVname" use="required"/>
</xsd:complexType>
<xsd:complexType name="ElemDestMQBroker">
  <xsd:sequence>
    <xsd:element maxOccurs="10" minOccurs="0" name="TOPICS" type="xsd:string"/>
  </xsd:sequence>
  <xsd:attribute name="mqsystem" use="optional"/>
  <xsd:attribute name="qname" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubEVENT">
  <xsd:sequence>
    <xsd:choice>
      <xsd:element name="ADABASEVENT" type="tns:PubAdabasEvent"/>
      <xsd:element name="DB2EVENT" type="tns:PubDB2Table"/>
      <xsd:element name="CICSEVENT" type="tns:PubCICSEvent"/>
      <xsd:element name="IDMSEVENT" type="tns:PubIDMSEvent"/>
      <xsd:element name="IMSEVENT" type="tns:PubIMSEvent"/>
      <xsd:element name="VSAMEVENT" type="tns:PubVSAEvent"/>
    </xsd:choice>
  </xsd:sequence>
  <xsd:attribute default="false" name="J2CA" type="xsd:boolean" use="optional"/>
  <xsd:attribute default="1047" name="CCSID" type="xsd:integer" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubAdabasEvent">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="FILEEVENT" type="tns:PubFileEvent"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubFileEvent">
  <xsd:sequence>
    <xsd:element name="EVENTID" type="tns:PubEventID"/>
  </xsd:sequence>
  <xsd:attribute name="mapname" type="tns:nbExternalMAPName"/>
  <xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
  <xsd:attribute default="ALL" name="type" type="tns:enumUpdateType"/>
  <xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute default="CDC" name="propagationtype" type="tns:enumPropType" use="optional"/>
  <xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubDB2Table">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="TABLE" type="tns:PubDB2Event"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubDB2Event">
  <xsd:sequence>
    <xsd:element name="EVENTID" type="tns:PubEventID"/>
  </xsd:sequence>
  <xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
  <xsd:attribute name="type" type="tns:enumUpdateType" use="required"/>
  <xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute default="CDC" name="propagationtype" type="tns:enumPropType" use="optional"/>
  <xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
  <xsd:attribute name="qualifier" type="tns:nbPlan" use="optional"/>
  <xsd:attribute name="name" type="tns:TableName" use="required"/>
</xsd:complexType>
<xsd:complexType name="PubCICSEvent">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="DLIEVENT" type="tns:PubDLIEvent"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="TEMPSTOREVENT" type="tns:PubTempStorEvent"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="TRANSIENTDATAEVENT" type="tns:PubTransEvent"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="FILEEVENT" type="tns:PubFileEvent"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>

```

```

<xsd:attribute name="transaction" type="tns:CICSTran" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubEventID">
  <xsd:attribute name="id" type="tns:nbID"/>
  <xsd:attribute name="description" type="tns:PubDesc"/>
</xsd:complexType>
<xsd:complexType name="PubIDMSEvent">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="FILEEVENT" type="tns:PubFileEvent"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>
  <xsd:attribute name="transaction" type="tns:CICSTran" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubIMSEvent">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="DLIEVENT" type="tns:PubDLIEvent"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>
  <xsd:attribute name="transaction" type="tns:CICSTran" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubVSAMEvent">
  <xsd:sequence>
    <xsd:element maxOccurs="unbounded" name="FILEEVENT" type="tns:PubFileEvent"/>
  </xsd:sequence>
  <xsd:attribute name="sourcename" type="tns:PubNAME" use="required"/>
  <xsd:attribute name="checkpoint" type="xsd:positiveInteger" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubDLIEvent">
  <xsd:sequence>
    <xsd:element name="EVENTID" type="tns:PubEventID"/>
  </xsd:sequence>
  <xsd:attribute name="mapname" type="tns:nbExternalMAPName" use="required"/>
  <xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
  <xsd:attribute default="ALL" name="type" type="tns:enumUpdateType"/>
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute default="CDC" name="propagationtype" type="tns:enumPropType" use="optional"/>
  <xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
  <xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubTransEvent">
  <xsd:sequence>
    <xsd:element name="EVENTID" type="tns:PubEventID"/>
  </xsd:sequence>
  <xsd:attribute name="tdname" type="tns:TDQName" use="required"/>
  <xsd:attribute name="mapname" type="tns:nbExternalMAPName" use="required"/>
  <xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
  <xsd:attribute default="ALL" name="type" type="tns:enumUpdateType"/>
  <xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute default="CDC" name="propagationtype" type="tns:enumPropType" use="optional"/>
  <xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubTempStorEvent">
  <xsd:sequence>
    <xsd:element name="EVENTID" type="tns:PubEventID"/>
  </xsd:sequence>
  <xsd:attribute name="tsname" type="tns:CICSName" use="required"/>
  <xsd:attribute name="mapname" type="tns:nbExternalMAPName" use="required"/>
  <xsd:attribute default="UP" name="desiredstate" type="tns:enumUpDown" use="optional"/>
  <xsd:attribute default="ALL" name="type" type="tns:enumUpdateType"/>
  <xsd:attribute name="destination" type="tns:PubNAME" use="optional"/>
  <xsd:attribute name="member" type="tns:optMemberName" use="optional"/>
  <xsd:attribute default="CDC" name="propagationtype" type="tns:enumPropType" use="optional"/>
  <xsd:attribute name="rulename" type="tns:PubRULE" use="optional"/>
</xsd:complexType>
<xsd:complexType name="PubEventField">
  <xsd:attribute name="name" type="tns:nbFLDName"/>
  <xsd:attribute name="length" type="xsd:nonNegativeInteger"/>
  <xsd:attribute name="offset" type="xsd:nonNegativeInteger"/>
  <xsd:attribute name="sqltype" type="tns:enumFLDSqltype"/>
</xsd:complexType>
<xsd:complexType name="PubEventColumn">
  <xsd:attribute name="name" type="tns:nbFLDName"/>
</xsd:complexType>
<xsd:complexType name="ElemMSGMap">
  <xsd:attribute name="type" type="tns:enumMSGType" use="required"/>
  <xsd:attribute name="index" type="xsd:integer" use="required"/>
  <xsd:attribute name="name" type="tns:nbLongExternalMAPName" use="required"/>
</xsd:complexType>
<xsd:complexType name="DocumentationBaseType">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="source" type="tns:optSOURCEUri" use="optional"/>
      <xsd:attributeGroup ref="xml:specialAttrs"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<xsd:simpleType name="preservedString">
  <xsd:restriction base="xsd:string">
    <xsd:whiteSpace value="preserve"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="prefixedHexOctet">

```

```

<xsd:restriction base="xsd:string">
  <xsd:pattern value="0(x|X)[abcdefABCDEF0-9][abcdefABCDEF0-9]"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumFLDFillTokens">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="null"/>
    <xsd:enumeration value="NULL"/>
    <xsd:enumeration value="zero"/>
    <xsd:enumeration value="ZERO"/>
    <xsd:enumeration value="blank"/>
    <xsd:enumeration value="BLANK"/>
    <xsd:enumeration value="none"/>
    <xsd:enumeration value="NONE"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPROVIDERNumeric">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="Integer"/>
    <xsd:enumeration value="Decimal"/>
    <xsd:enumeration value="None"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumUpdateType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ALL"/>
    <xsd:enumeration value="DELETE"/>
    <xsd:enumeration value="INSERT"/>
    <xsd:enumeration value="UPDATE"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumUpdateTypeDB2">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="DELETE"/>
    <xsd:enumeration value="INSERT"/>
    <xsd:enumeration value="UPDATE"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbFLDFill">
  <xsd:union memberTypes="tns:prefixedHexOctet tns:enumFLDFillTokens"/>
</xsd:simpleType>
<xsd:simpleType name="enumLiteralType">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="string"/>
    <xsd:enumeration value="hex"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumFLDFormat">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ADA_CHARACTER"/>
    <xsd:enumeration value="BIGINT"/>
    <xsd:enumeration value="BINARY"/>
    <xsd:enumeration value="CHARACTER"/>
    <xsd:enumeration value="DATE"/>
    <xsd:enumeration value="FLOAT"/>
    <xsd:enumeration value="GROUP"/>
    <xsd:enumeration value="INTEGER"/>
    <xsd:enumeration value="SMALLINT"/>
    <xsd:enumeration value="UNICODE"/>
    <xsd:enumeration value="DISPLAY_NUMERIC"/>
    <xsd:enumeration value="PACKED"/>
    <xsd:enumeration value="TIME"/>
    <xsd:enumeration value="ADA_UNPACKED"/>
    <xsd:enumeration value="ADA_LOGICAL"/>
    <xsd:enumeration value="LOGICAL"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumFLDSqltype">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="SQL_BIGINT"/>
    <xsd:enumeration value="SQL_CHAR"/>
    <xsd:enumeration value="SQL_NUMERIC"/>
    <xsd:enumeration value="SQL_DECIMAL"/>
    <xsd:enumeration value="SQL_INTEGER"/>
    <xsd:enumeration value="SQL_SMALLINT"/>
    <xsd:enumeration value="SQL_FLOAT"/>
    <xsd:enumeration value="SQL_REAL"/>
    <xsd:enumeration value="SQL_DOUBLE"/>
    <xsd:enumeration value="SQL_DATE"/>
    <xsd:enumeration value="SQL_TIME"/>
    <xsd:enumeration value="SQL_TIMESTAMP"/>
    <xsd:enumeration value="SQL_VARCHAR"/>
    <xsd:enumeration value="SQL_LONGVARCHAR"/>
    <xsd:enumeration value="SQL_BINARY"/>
    <xsd:enumeration value="SQL_LONGVARBINARY"/>
    <xsd:enumeration value="SQL_UNICODE"/>
    <xsd:enumeration value="SQL_UNICODE_VARCHAR"/>
    <xsd:enumeration value="SQL_UNICODE_LONGVARCHAR"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPubFormat">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="XML"/>
  </xsd:restriction>

```

```

    <xsd:enumeration value="HTX"/>
    <xsd:enumeration value="SQLDA"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumSequentialType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="none"/>
    <xsd:enumeration value="zfs"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="integerFLDLevel">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="49"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="mapReduceThreadCount">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="2"/>
    <xsd:maxInclusive value="9"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbFLDColname">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbPlan">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubDesc">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubRULE">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubRVname">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="48"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubJCAService">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="32"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubJCAOptions">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="254"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubMachineID">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="32"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbID">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="16"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DLIName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbFLDName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbExternalMAPName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
    <xsd:whiteSpace value="preserve"/>
  </xsd:restriction>
</xsd:simpleType>

```

```

</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbLongExternalMAPName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
    <xsd:whiteSpace value="preserve"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optCICSxxTransact">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optCICSxxProgram">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optCICSxxConnect">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optVSAMCicstran">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optVSAMCicsconn">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optDDname">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optDSname">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="44"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optExitName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optMemberName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbMemberName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="AdabasDatabase">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="AdabasFilename">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="50"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="AdabasSubsys">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="AdabasPassword">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>

```

```

<xsd:simpleType name="enumVSAMRecfmt">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="fixed"/>
    <xsd:enumeration value="variable"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optMFSName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbMFSName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbSERVICENamespace">
  <xsd:restriction base="xsd:anyURI">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="128"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optSERVICENamespace">
  <xsd:restriction base="xsd:anyURI">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="128"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbSERVICENAME">
  <xsd:restriction base="xsd:NCName">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="32"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbPROVIDERName">
  <xsd:restriction base="xsd:NCName">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="32"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPROVIDERTYPE">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="MFLOW"/>
    <xsd:enumeration value="BLI"/>
    <xsd:enumeration value="ACIBATCH"/>
    <xsd:enumeration value="ACICICS"/>
    <xsd:enumeration value="DATA"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPROVIDERSecurity">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="None"/>
    <xsd:enumeration value="Userid"/>
    <xsd:enumeration value="UseridPassword"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPROVIDERFormat">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="No"/>
    <xsd:enumeration value="Format"/>
    <xsd:enumeration value="Short"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="nbOPERATIONName">
  <xsd:restriction base="xsd:NCName">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="32"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumMSGType">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="inout"/>
    <xsd:enumeration value="input"/>
    <xsd:enumeration value="output"/>
    <xsd:enumeration value="outvar"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="integerMSGOpt">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:totalDigits value="1"/>
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="3"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optMFLDName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPseudoColumn">

```

```

<xsd:restriction base="xsd:string">
  <xsd:enumeration value="dsname"/>
  <xsd:enumeration value="member"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumLPAGECond">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="EQ"/>
    <xsd:enumeration value="NE"/>
    <xsd:enumeration value="GT"/>
    <xsd:enumeration value="GE"/>
    <xsd:enumeration value="LT"/>
    <xsd:enumeration value="LE"/>
    <xsd:enumeration value="NA"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumMapdefSyntax">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="COBOL"/>
    <xsd:enumeration value="WSCOBOL"/>
    <xsd:enumeration value="ASM"/>
    <xsd:enumeration value="C"/>
    <xsd:enumeration value="EDITOR"/>
    <xsd:enumeration value="PLI"/>
    <xsd:enumeration value="XML"/>
    <xsd:enumeration value="DB2"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optAGENTVersion">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="12"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optAGENTLocation">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="48"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumMAPDefines">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="area"/>
    <xsd:enumeration value="msg"/>
    <xsd:enumeration value="vsam"/>
    <xsd:enumeration value="sequential"/>
    <xsd:enumeration value="natural"/>
    <xsd:enumeration value="adabas"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumTARGETType">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="CICS"/>
    <xsd:enumeration value="CA-IDMS"/>
    <xsd:enumeration value="IMSTM"/>
    <xsd:enumeration value="ACI"/>
    <xsd:enumeration value="ADABAS"/>
    <xsd:enumeration value="REXX"/>
    <xsd:enumeration value="Internal"/>
    <xsd:enumeration value="DB2"/>
    <xsd:enumeration value="IMSDB"/>
    <xsd:enumeration value="VSAM"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumGeneralMapStatus">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="enabled"/>
    <xsd:enumeration value="disabled"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumTransmissionType">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="request-response"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optRACFUserName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="IDMSCVNumber">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="256"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DBID">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="1"/>
    <xsd:maxInclusive value="65535"/>
  </xsd:restriction>
</xsd:simpleType>

```

```

<xsd:simpleType name="USERID">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optINPUTFrom">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="0"/>
    <xsd:maxLength value="64"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="optSOURCEUri">
  <xsd:restriction base="xsd:anyURI">
    <xsd:maxLength value="128"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubNAME">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubQueue">
  <xsd:restriction base="xsd:positiveInteger">
    <xsd:minInclusive value="10"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPropType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="CDC"/>
    <xsd:enumeration value="DBR"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumUpDown">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="UP"/>
    <xsd:enumeration value="DOWN"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumYesNo">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="YES"/>
    <xsd:enumeration value="NO"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumBrokerServerType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="CICS"/>
    <xsd:enumeration value="BATCH"/>
    <xsd:enumeration value="STC"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumUserPass">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Userid"/>
    <xsd:enumeration value="UseridPassword"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumOpt">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="DURABLE"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPubDestType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="J2CA"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubTime">
  <xsd:restriction base="xsd:nonNegativeInteger">
    <xsd:minInclusive value="0"/>
    <xsd:maxInclusive value="1440"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="CICSName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="BackendName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="8"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="CICSTran">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>

```

```

<xsd:simpleType name="PubSubsys">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="DB2Column">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="128"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="TableName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="27"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubSourceConnection">
  <xsd:union memberTypes="tns:PubSubsys tns:DBID"/>
</xsd:simpleType>
<xsd:simpleType name="TDQName">
  <xsd:restriction base="xsd:string">
    <xsd:minLength value="1"/>
    <xsd:maxLength value="4"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumDirectoryAccess">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="read"/>
    <xsd:enumeration value="write"/>
    <xsd:enumeration value="execute"/>
    <xsd:enumeration value="all"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPayloadType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="WMQ"/>
    <xsd:enumeration value="USR"/>
    <xsd:enumeration value="JMS"/>
    <xsd:enumeration value="wmq"/>
    <xsd:enumeration value="usr"/>
    <xsd:enumeration value="jms"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumPCBPos">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="MULTIPLE"/>
    <xsd:enumeration value="SINGLE"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubDB2Type">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="SQL"/>
    <xsd:enumeration value="IFI"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="enumCICSTransId">
  <xsd:restriction base="xsd:NMTOKEN">
    <xsd:enumeration value="None"/>
    <xsd:enumeration value="Prefix"/>
    <xsd:enumeration value="Suffix"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PubAdabasType">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Server"/>
    <xsd:enumeration value="Logstream"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:schema>

```

Example: SAMPLE-REC

Before using the DVS_MAP API, review this example.

COBOL copy book

```

IDENTIFICATION DIVISION.                                00020001
PROGRAM-ID.  DMFSAMPLE.                                  00030001
*                                                         00040001
*****                                                    00050001
* © COPYRIGHT ROCKET SOFTWARE, INC. 1991, 2017          * 00051001
* SAMPLE COPYBOOK LISTING WITH VARIOUS TYPES          * 00052002
*****                                                    00050001

```

```

*
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
DATA DIVISION.
WORKING-STORAGE SECTION.
*****
*
01  SAMPLE-REC.
05  KEY-5          PIC X(05).
05  CHAR-30       PIC X(30).
05  DISPNUM-3-3   PIC S9(03)V9(3).
05  DISPNUM-0-3   PIC SV9(3).
05  DISPNUM-15-0  PIC S9(15).
05  DISPNUM-31-0  PIC S9(31).
05  INT5-2        PIC S9(04) COMP-5.
05  INT-2         PIC S9(04) COMP.
05  INT-4         PIC S9(09) COMP.
05  INT-8         PIC S9(18) COMP.
05  PACKED-3-3    PIC S9(03)V9(3) COMP-3.
05  PACKED-0-3    PIC SV9(3) COMP-3.
05  FLOAT-4       USAGE IS COMP-1.
05  FLOAT-8       USAGE IS COMP-2.
05  CHAR-300      PIC X(300).
03  FILLER        PIC X(56).

*
PROCEDURE DIVISION.
MOVE +0 TO RETURN-CODE.
GOBACK.

```

Sample XML

```

<?xml version="1.0" encoding="UTF-8"?>
<NEONDMF xmlns="urn:Server-Metaddata-Schema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0"
xsi:schemaLocation="urn:Server-Metaddata-Schema dmf20.xsd">
  <MAP defines="sequential" name="DMFSAMPLE">
    <SEQUENTIAL ddname="VDDN" dsname="userid1.DMFSAMP.DATA.SEQ" member="" type="none"/>
    <FLD colname="SAMPLE_REC" enabled="false" fill="null" format="GROUP" length="480" level="1" name="SAMPLE-REC"
offset="0" precision="0"
scale="0" signed="false" sqltype="SQL_CHAR">
      <SOURCE>
        01  SAMPLE-REC.
      </FLD>
      <FLD colname="KEY_5" enabled="true" fill="blank" format="CHARACTER" length="5" level="2" name="KEY-5"
offset="0" precision="5" scale="0"
signed="false" sqltype="SQL_CHAR">
        <SOURCE>
          05  KEY-5          PIC X(05).
        </FLD>
        <FLD colname="CHAR_30" enabled="true" fill="blank" format="CHARACTER" length="30" level="2" name="CHAR-30"
offset="5" precision="30"
scale="0" signed="false" sqltype="SQL_CHAR">
          <SOURCE>
            05  CHAR-30       PIC X(30).
          </FLD>
          <FLD colname="DISPNUM_3_3" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="6" level="2"
name="DISPNUM-3-3" offset="35"
precision="6" scale="3" signed="true" sqltype="SQL_CHAR">
            <SOURCE>
              05  DISPNUM-3-3   PIC S9(03)V9(3).
            </FLD>
            <FLD colname="DISPNUM_0_3" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="3" level="2"
name="DISPNUM-0-3" offset="41"
precision="3" scale="3" signed="true" sqltype="SQL_CHAR">
              <SOURCE>
                05  DISPNUM-0-3   PIC SV9(3).
              </FLD>
              <FLD colname="DISPNUM_15_0" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="15" level="2"
name="DISPNUM-15-0" offset="44"
precision="15" scale="0" signed="true" sqltype="SQL_CHAR">
                <SOURCE>
                  05  DISPNUM-15-0  PIC S9(15).
                </FLD>
                <FLD colname="DISPNUM_31_0" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="31" level="2"
name="DISPNUM-31-0" offset="59"
precision="31" scale="0" signed="true" sqltype="SQL_CHAR">
                  <SOURCE>
                    05  DISPNUM-31-0  PIC S9(31).
                  </FLD>
                  <FLD colname="INT5_2" enabled="true" fill="null" format="SMALLINT" length="2" level="2" name="INT5-2"
offset="90" precision="2" scale="0"
signed="true" sqltype="SQL_SMALLINT">
                    <SOURCE>
                      05  INT5-2        PIC S9(04) COMP-5.
                    </FLD>
                    <FLD colname="INT_2" enabled="true" fill="null" format="SMALLINT" length="2" level="2" name="INT-2" offset="92"
precision="2" scale="0"
signed="true" sqltype="SQL_SMALLINT">
                      <SOURCE>
                        05  INT-2         PIC S9(04) COMP.
                      </FLD>
                      <FLD colname="INT_4" enabled="true" fill="null" format="INTEGER" length="4" level="2" name="INT-4" offset="94"
precision="4" scale="0"
signed="true" sqltype="SQL_INTEGER">
                        <SOURCE>
                          05  INT-4         PIC S9(09) COMP.
                        </FLD>

```

```

        <FLD colname="INT_8" enabled="true" fill="null" format="BIGINT" length="8" level="2" name="INT-8" offset="98"
precision="8" scale="0"
signed="false" sqltype="SQL_BIGINT">
        <SOURCE>          05 INT-8                                PIC S9(18) COMP.                010911</SOURCE>
    </FLD>
    <FLD colname="PACKED_3_3" enabled="true" fill="null" format="PACKED" length="4" level="2" name="PACKED-3-3"
offset="106" precision="6"
scale="3" signed="true" sqltype="SQL_DECIMAL">
    <SOURCE>          05 PACKED-3-3                                PIC S9(03)V9(3) COMP-3.        010911</SOURCE>
    </FLD>
    <FLD colname="PACKED_0_3" enabled="true" fill="null" format="PACKED" length="2" level="2" name="PACKED-0-3"
offset="110" precision="3"
scale="3" signed="true" sqltype="SQL_DECIMAL">
    <SOURCE>          05 PACKED-0-3                                PIC S9(3) COMP-3.            010911</SOURCE>
    </FLD>
    <FLD colname="FLOAT_4" enabled="true" fill="null" format="FLOAT" length="4" level="2" name="FLOAT-4"
offset="112" precision="4" scale="0"
signed="false" sqltype="SQL_FLOAT">
    <SOURCE>          05 FLOAT-4                                USAGE IS COMP-1.</SOURCE>
    </FLD>
    <FLD colname="FLOAT_8" enabled="true" fill="null" format="FLOAT" length="8" level="2" name="FLOAT-8"
offset="116" precision="8" scale="0"
signed="false" sqltype="SQL_DOUBLE">
    <SOURCE>          05 FLOAT-8                                USAGE IS COMP-2.</SOURCE>
    </FLD>
    <FLD colname="CHAR_300" enabled="true" fill="blank" format="CHARACTER" length="300" level="2" name="CHAR-300"
offset="124" precision="300"
scale="0" signed="false" sqltype="SQL_VARCHAR">
    <SOURCE>          05 CHAR-300                                PIC X(300).</SOURCE>
    </FLD>
    <FLD colname="FILLER" enabled="false" fill="blank" format="CHARACTER" length="56" level="2" name="FILLER"
offset="424" precision="56"
scale="0" signed="false" sqltype="SQL_CHAR">
    <SOURCE>          03 FILLER                                PIC X(56).</SOURCE>
    </FLD>
</MAP>
</NEONDMF>

```

Example: STAFF-RECORD

Before using the DVS_MAP API, review this example.

COBOL copybook

```

009400 01 STAFF-RECORD.
009500 05 ID          PIC S9(3) COMP.
009700 05 NAME        PIC X(9).
009800 05 DEPT        PIC S9(3) COMP.
009900 05 JOB         PIC X(5).
010000 05 YEARS      PIC S9(3) COMP.
010000 05 SALARY     PIC 9(6)V9(2).
010000 05 COMM       PIC 9(6)V9(2).

```

Sample XML

```

<?xml version="1.0" encoding="UTF-8"?>
<NEONDMF xmlns="urn:Server-Metadata-Schema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.0"
xsi:schemaLocation="urn:Server-Metadata-Schema http://192.168.55.28:1501/SWICNTL/xsdm20.xsd">
    <MAP defines="sequential" name="STAFF_SEQ">
        <SEQUENTIAL ddname="VDDN" dsname="CSD.AI38.HLV1.STAFF.SEQ1" member=""/>
        <FLD colname="STAFF_RECORD" enabled="false" fill="null" format="GROUP" length="36" level="1" name="STAFF-
RECORD" offset="0" precision="0"
scale="0" signed="false" sqltype="SQL_CHAR">
            <SOURCE>009400 01 STAFF-RECORD.                                00010000</SOURCE>
        </FLD>
        <FLD colname="ID" enabled="true" fill="null" format="SMALLINT" length="2" level="2" name="ID" offset="0"
precision="2" scale="0"
signed="true" sqltype="SQL_SMALLINT">
            <SOURCE>009500 05 ID          PIC S9(3) COMP.                                00020000</SOURCE>
        </FLD>
        <FLD colname="NAME" enabled="true" fill="blank" format="CHARACTER" length="9" level="2" name="NAME" offset="2"
precision="0" scale="0"
signed="false" sqltype="SQL_CHAR">
            <SOURCE>009700 05 NAME        PIC X(9).                                00030000</SOURCE>
        </FLD>
        <FLD colname="DEPT" enabled="true" fill="null" format="SMALLINT" length="2" level="2" name="DEPT" offset="11"
precision="2" scale="0"
signed="true" sqltype="SQL_SMALLINT">
            <SOURCE>009800 05 DEPT        PIC S9(3) COMP.                                00040000</SOURCE>
        </FLD>
        <FLD colname="JOB" enabled="true" fill="blank" format="CHARACTER" length="5" level="2" name="JOB" offset="13"
precision="5" scale="0"
signed="false" sqltype="SQL_CHAR">

```

```

        <SOURCE>009900    05  JOB      PIC X(5).                                00050000</SOURCE>
    </FLD>
    <FLD colname="YEARS" enabled="true" fill="null" format="SMALLINT" length="2" level="2" name="YEARS" offset="18"
precision="2" scale="0"
signed="true" sqltype="SQL_SMALLINT">
        <SOURCE>010000    05  YEARS    PIC S9(3) COMP.                        00060000</SOURCE>
    </FLD>
    <FLD colname="SALARY" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="8" level="2" name="SALARY"
offset="20" precision="8"
scale="2" signed="false" sqltype="SQL_CHAR">
        <SOURCE>010000    05  SALARY    PIC 9(6)V9(2).                        00070002</SOURCE>
    </FLD>
    <FLD colname="COMM" enabled="true" fill="blank" format="DISPLAY_NUMERIC" length="8" level="2" name="COMM"
offset="28" precision="8"
scale="2" signed="false" sqltype="SQL_CHAR">
        <SOURCE>010000    05  COMM      PIC 9(6)V9(2).                        00080002</SOURCE>
    </FLD>
</MAP>
</NEONDMF>

```

Java code

The following example shows the JDBC connection string and the call to the DVS_MAP API. For information about the JDBC connection string, see "Connecting to a data source using JDBC" in the *Solutions Guide*.

```

import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.sql.*;

class ImportMap {
    public static void main(String[] args) throws IOException, SQLException {
        if (args.length < 1) {
            System.out.println("Usage: filename argument required");
            return;
        }
        String xmlText = new String(Files.readAllBytes(Paths.get(args[0])));

        final String url = "jdbc:rs:dv://host:port;DatabaseType=DVS;password=password;user=user1d1;";

        try (final Connection conn = DriverManager.getConnection(url)) {
            final String sql = "call DVS_MAP('DMF', 'IMPORT', ?, 'SAVE', 'REFRESH')";

            try (final PreparedStatement statement = conn.prepareStatement(sql)) {
                statement.setString(1, xmlText);
                statement.execute();
            }
        }
    }
}

```


Accessibility

The publications for IBM Open Data Analytics for z/OS are available in IBM Knowledge Center and Adobe Portable Document Format (PDF) and comply with accessibility standards. If you experience difficulties when you use any of the information, notify IBM through one of the comment methods described in [“How to send your comments to IBM” on page xi](#).

Notices

This information was developed for products and services offered in the U.S.A. or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Site Counsel
IBM Corporation
2455 South Road
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

COPYRIGHT LICENSE:

This information might contain sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)® are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" (www.ibm.com/legal/copytrade.shtml).

Rocket is a registered trademark of Rocket Software, Inc.

Other company, product, or service names may be trademarks or service marks of others.

Index

A

- accessibility
 - contact IBM [147](#)
- ACIMAPREDUCETRACKS [45](#)
- Adabas
 - configuring [2](#)
 - server configuration [3](#)
- Anaconda [117](#)
- analytics
 - configure [48](#)
- ANSI SQL data access [1](#)
- applications
 - connecting an application to a data source [70](#)

B

- Big SQL
 - configure [26](#)

C

- CA IDMS
 - configuring [34](#)
 - modifying server configuration member [34](#), [43](#)
 - verify access [35](#)
- configuring
 - access to data sources [105](#)
 - Adabas [2](#)
 - ADDI [53](#)
 - CA IDMS [34](#)
 - Hadoop access [110](#)
 - IBM MQ [42](#)
 - JDBC Gateway [106](#)
 - Oracle access [109](#)
 - RAA [60](#)
 - relational database management systems (RDBMS) [5](#)
- Configuring
 - DB2 for Distributed Relational Database Architecture (DRDA) [12](#)
 - DB2 for Resource Recovery Services attachment facility (RRSAF) [14](#)
 - Resource Recovery Services attachment facility (RRSAF) [12](#)
 - server started task JCL [7](#)
- configuring CA IDMS
 - modifying server configuration member [34](#), [43](#)
- configuring IMS
 - modifying server configuration member [38](#)
 - server started task JCL [37](#)
- configuring System Management Facility (SMF) files [49](#)

D

- dashDB
 - configure [27](#)

- data sources
 - CA IDMS [34](#)
 - configuring Adabas [2](#)
 - configuring relational database management systems (RDBMS) [5](#)
 - IBM MQ [42](#)
 - sequential (using ADDI) [53](#)
 - sequential (using RAA) [60](#)
 - VSAM (using ADDI) [53](#)
 - VSAM (using RAA) [60](#)
- Db2
 - configure the started task [7](#)
- Db2 data access [16](#)
- Db2 Direct [16](#)
- DB2 unload data set, configuring access [15](#)
- Db2 Virtualization (DB2V) [65](#)
- DBCTL
 - configure server configuration [37](#)
- DFstore support [119](#)
- DISABLEATTACH parameter [33](#)
- distributed databases
 - configure [20](#)
- Distributed Relational Database Architecture (DRDA)
 - configuring access [7](#), [21](#)
- driver class name [112](#)
- DVS_MAP API [121](#), [123](#), [142](#), [144](#)

F

- feedback [xi](#)

H

- Hadoop
 - configuring access [110](#)

I

- IBM Application Discovery and Delivery Intelligence
 - authentication [59](#)
 - configuration [54](#), [55](#), [57](#), [58](#)
 - configuring [53](#)
 - rules [58](#)
- IBM DB2 Analytics Accelerator [119](#)
- IBM DB2 for z/OS
 - configure access to data [6](#)
- IBM MQ
 - configuration [43](#)
 - configuring [42](#)
 - rules [43](#)
- IBM Rational Asset Analyzer
 - authentication [64](#)
 - configuration [60](#), [61](#), [63](#), [64](#)
 - configuring [60](#)
 - rules [64](#)
- IBM z/OS Platform for Apache Spark [119](#)

- IMS
 - configuring server started task JCL [37](#)
- IMS database
 - configure access to data [36](#)
- IMS Direct
 - modifying server configuration member [38](#)

J

- JARS [112](#)
- JCL
 - configuring [2](#), [34](#), [42](#)
 - started task [2](#), [34](#), [42](#)
- JDBC
 - APIs [95](#)
 - buffering data [89](#)
 - Connection properties [70](#)
 - debugging and tracing [85](#)
 - MapReduce [93](#)
 - Parallel IO [90](#)
 - performance [88](#)
- JDBC Gateway
 - configuring [106](#)
 - preferences [112](#)
- JDBC libraries [112](#)
- JDBC preferences [112](#)
- JDBCWeb Services
 - Error handling [85](#)

L

- Linux, UNIX, and Windows databases
 - configure [28](#)
- Log preferences [112](#)

M

- Microsoft SQL Server
 - configure [29](#)
- modifying the data buffer for VSAM files [46](#)
- modifying the index buffer for VSAM files [46](#)
- MULTACC rule [45](#)

N

- native Db2 subsystem, access and display [33](#)
- Notices [149](#)

O

- Oracle
 - configuring access [109](#)
- Oracle Database Provider for DRDA
 - configure [30](#)
- Output preferences [113](#)

P

- preferences
 - JDBC [112](#)
 - Log [112](#)
 - Output [113](#)

- Python [117](#)

Q

- QMF DRDA Server
 - configure [31](#)
- query acceleration rule [14](#)

R

- relational database management systems (RDBMS)
 - configuring [5](#)

S

- Security
 - configure [6](#)
- sending to IBM
 - reader comments [xi](#)
- sequential files
 - configuring access [44](#)
- server configuration member
 - configuring for CA IDMS [34](#), [43](#)
 - configuring for IMS Direct [38](#)
- server event facility (SEF)
 - configure ADDI [58](#)
 - configure Big SQL [26](#)
 - configure dashDB [27](#)
 - configure DB2 unload data set access [15](#)
 - configure IBM MQ [43](#)
 - configure Linux, UNIX, and Windows databases [28](#)
 - configure Microsoft SQL Server [29](#)
 - configure Oracle Database Provider for DRDA [30](#)
 - configure QMF DRDA Server [31](#)
 - configure query acceleration [14](#)
 - configure RAA [64](#)
 - read ahead tracks [45](#)
- server parameters
 - DISABLEATTACH [33](#)
 - SQLENGVSAMDATABUFF [46](#)
 - SQLENGVSAMINDEXBUFF [46](#)
- server started task JCL
 - configuring for IMS [37](#)
- SET CURRENT QUERY ACCELERATION [14](#)
- SQLENGVSAMDATABUFF parameter [46](#)
- SQLENGVSAMINDEXBUFF parameter [46](#)
- summary of changes for IBM Open Data Analytics for z/OS Solutions Guide [xii](#)
- SYSLOG
 - configuring [52](#)
- System Management Facility (SMF) files
 - configuring [49](#)

U

- unload data set, configuring access [15](#)
- URL template [112](#)
- User-defined table functions [65](#)

V

- VSAM
 - access to VSAM data [45](#)

VSAM (*continued*)
configure access [46](#)

Z

zFS files
configuring access [48](#)



SC27-9036-00

