IBM Db2 Analytics Accelerator for z/OS
7.5.12

Stored Procedures Reference

SH12-7103-13

Note

Before you use this information and the product it supports, read the information in "Notices" on page 301.

Fourteenth Edition, May 2024

This edition applies to version 7.5.12 (updated version May 2024) of IBM® Db2® Analytics Accelerator for z/OS® (product number 5697-DA7), and to all subsequent releases and modifications until otherwise indicated in new editions. This edition replaces SH12-7103-12. Changes to this edition are marked with a vertical bar.

Contents

About this book

This book provides information about the following subjects:

- Stored procedures of IBM Db2 Analytics Accelerator for z/OS
- Error handling
- Using client applications or batch processes to run these stored procedures
- Messages about errors and warnings that might be displayed as a result of running these stored procedures

Audience

This book is intended for Db2 for z/OS administrators, z/OS administrators, and application programmers who are entrusted with the following tasks:

- Running IBM Db2 Analytics Accelerator stored procedures to define and update tables as well as administer accelerators
- Operating Db2 for z/OS with an accelerator running on an IBM Integrated Analytics System or with Db2 Analytics Accelerator on Z
- User rights management with regard to the stored procedures for IBM Db2 Analytics Accelerator for z/OS
- Recovery from errors caused by these stored procedures

What's new in version 7.5?

IBM Db2 Analytics Accelerator for z/OS Version 7.5 is a major upgrade that provides powerful new features:

- IBM Integrated Synchronization for incremental update processing
- Multi-node installations of Db2 Analytics Accelerator on Z with increased processing power for massive workloads

Enhancements in this release

For enhancements in this or previous follow-up releases, see the What's new document.

Chapter 1. Stored procedures

IBM Db2 Analytics Accelerator for z/OS stored procedures are the administration interface for your accelerators. When you invoke a function from your administration client, the corresponding stored procedure is called. In addition, IBM Db2 Analytics Accelerator for z/OS stored procedures can be run from the command line or be embedded in custom applications. The stored procedures provide functions that are related to tables and accelerators. All stored procedures commit the transactions that were triggered by the calling applications. The IBM Db2 Analytics Accelerator for z/OS stored procedures are presented here in alphabetical order.

The installation of the stored procedures is described in the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide*.

Related information

IBM DB2 Analytics Accelerator for z/OS: Installation Guide

SYSPROC.ACCEL_ADD_ACCELERATOR2

The stored procedure associates an accelerator with a Db2 subsystem or Db2 data-sharing group. The process is called pairing and is a mandatory configuration step; without it, an accelerator cannot be used.

Details

This stored procedure requires a valid pairing code that you need to provide as an input value. You obtain this code by using the IBM Db2 Analytics Accelerator Console. The process is described in the topic *Obtaining the pairing code for accelerator authentication* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*. The pairing code can be valid for a period of up to 24 hours. Its use is not restricted to a particular Db2 subsystem. It is therefore valid only for the first SYSPROC.ACCEL_ADD_ACCELERATOR2 call after its creation, and is invalidated after this call even if the stored procedure did not run successfully.

Attention:

- A successful run of the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure results in the creation of an authentication token. This token is stored in a system table of the associated Db2 subsystem. For each IP address and port combination, there can be only one accelerator. The stored procedure refuses duplicate definitions. If the accelerator already contains data that is related to a Db2 subsystem or data sharing group and if this data is associated with the same location name, then this data is deleted during the stored-procedure run. The removal of old data ensures that a newly added accelerator is always "clean".
- If you restore a backup of the Db2 catalog that does not contain the most recent authentication token for an accelerator, the data on the accelerator becomes orphaned. The data cannot be recovered because a renewed pairing process will delete it.

Syntax

```
CALL SYSPROC.ACCEL_ADD_ACCELERATOR2
(accelerator_name,
ip_address,
port,
pin,
location,
options,
message);
```

Options description

accelerator_name

The name of the accelerator. The following restrictions are in place for this name:

- It must be unique within the Db2 subsystem or data sharing group.
- It must not consist of more than eight characters.
- Allowed characters are A-Z, 0-9.

The SYSACCEL.SYSACCELERATORS table contains the names of accelerators that are already defined.

ip_address

The IP address (IPv4 or IPv6) or DNS name of the virtual LAN (VLAN) that routes the network traffic to your accelerator.

port

The DRDA communication port of the chosen VLAN, that is, the numeric port or symbolic service that the VLAN uses to listen to requests.

pin

A four-digit number of temporary validity that you generate by using the IBM Db2 Analytics Accelerator Console.

location

The unique name of the accelerator in the SYSIBM.LOCATIONS table. If you do not specify a location name (value NULL), the value of *accelerator_name* is used.

Attention:

- The name must not be listed in the SYSIBM.LOCATIONS table of the Db2 communications database for a remote location.

  If you want to connect more than one Db2 subsystem to the same accelerator, you must use a different location name for each pairing. For each IP address and port combination, the assignment to an accelerator must be logically unique. If you disregard this rule and use the same location name, you will lose the connection to the previously paired Db2 subsystem, and the data on the accelerator will be deleted. This mechanism was implemented on purpose because it ensures that a newly added accelerator is always “clean”.
- The name must not consist of more than eight characters.
- Allowed characters are A-Z, 0-9.

options

An additional XML string that will allow you to specify options for the pairing process in future releases. Currently, you need not set this parameter.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the

SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF®).
- The name that you use as the value of the *accelerator_name* parameter must not pre-exist in the Db2 for z/OS catalog table SYSACCEL.SYSACCELERATEDTABLES.
- The location names of Db2 subsystems or data sharing groups that use the same accelerator must be unique and stable. To display the location name of a Db2 subsystem or data sharing group, run the following query:

```
SELECT CURRENT SERVER FROM SYSIBM.SYSDUMMYU
```

Important: If you must change a location name, first remove the accelerator by using the SYSPROC.ACCEL_REMOVE_ACCELERATOR stored procedure. After changing the name, redefine the accelerator.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

Each time you add an accelerator, the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure inserts a row in the SYSACCEL.SYSACCELERATORS catalog table in Db2 for z/OS. In addition, rows are inserted in the following tables of the Db2 communication database:

- SYSIBM.LOCATIONS
- SYSIBM.IPNAMES
- SYSIBM.USERNAMES

On the accelerator, the stored procedure permanently saves the authentication token, which is generated as a result of the pairing operation. It also creates the data structures that the Db2 subsystem requires for using the accelerator.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member

Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_ADD_TABLES

Defines accelerator-shadow tables. The input consists of a list of existing Db2 for z/OS tables, which serve as the basis. Instructions can be added for the use of unique constraints by the query optimizer. The stored procedure is not suitable for creating accelerator-only tables.

Details

The accelerator-shadow tables are populated in a later step by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. This later step provides the data that allows you to run accelerated queries.

In addition, the SYSPROC.ACCEL_ADD_TABLES stored procedure acknowledges the existence of the accelerator-shadow tables by writing corresponding entries to the SYSIBM.SYSACCELERATEDTABLES catalog table in Db2 for z/OS. Using the entries in this catalog table, Db2 for z/OS “knows” which queries it can route to an accelerator.

Each accelerator-shadow table will first be in *InitialLoadPending* state. This state indicates that you must load such a table with data and enable it before you can run accelerated queries against it.

Note: Tables or table columns that contain an edit procedure (EDITPROC) call are supported.

Restrictions

A table column is ignored, that is, not included in the definition of the accelerator-shadow table if one of the following conditions applies:

- The column uses a field procedure, that is, its value in the FLDPROC column of the SYSCOLUMNS table is `'Y'`.
- The column uses the AS SECURITY LABEL attribute, that is, its value in the DEFAULT column of the SYSCOLUMNS table is `'L'`.
- The data type of the column is not in the following list:
  - BIGINT
  - CHAR
  - DATE
  - DECIMAL
  - DOUBLE
  - FLOAT
  - GRAPHIC
  - INTEGER
  - REAL
  - SMALLINT
  - TIME
  - TIMESTAMP
  - VARCHAR
  - VARGRAPHIC

- Column-level access control is defined for the column, that is, SYSCOLUMNS(CONTROL_ID) is set to a value other than 0.

An accelerator-shadow table cannot be defined if one of the following conditions applies:

- The table is an accelerator-only table. These tables are created by the SQL statement CREATE TABLE ... IN ACCELERATOR. Because they already exist on an accelerator, they cannot be defined by this stored procedure. The stored procedure therefore fails if it encounters an accelerator-only table.
- The table uses a row-level security label, that is, its value in the SECURITY_LABEL column of the SYSTABLES table is `'R'`.
- None of the table columns qualifies for query routing to an accelerator.
- Row-level access control is defined for the table, that is, its value in the CONTROL column of the SYSTABLES table is `'R'` or `'B'`.
- The table space of the table is encoded in single-byte EBCDIC or in ASCII and the CCSID is not supported by the accelerator.
- The table contains columns that use CCSID 1200 or CCSID 1208, but the corresponding table space is not encoded in UNICODE.

  Queries against such tables cannot be routed to an accelerator because they would be queries in a mixed encoding, which is not supported.
- The length of one or more rows in the table exceeds the allowed maximum on the accelerator. This can happen if the table has two or more long columns of the data type VARCHAR FOR MIXED DATA.

Furthermore, if a table in the Db2 subsystem contains one or more columns of the type VARGRAPHIC or VARCHAR, and some of the column values exceed 16296 (VARGRAPHIC) or 16762 (VARCHAR) characters in length, the overlong values are truncated. Truncation also occurs under these conditions with VARCHAR FOR SBCS DATA columns encoded in UNICODE. The truncation increases the likelihood for deviating results. Warning AQT10086W is issued in cases like this.

Syntax

```
CALL SYSPROC.ACCEL_ADD_TABLES
(accelerator_name,
table_specifications,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_specifications

An XML input string that lists the accelerator-shadow tables to be defined. This string must conform to the structure of the `tableSpecifications` element in the SAQTSAMP(AQTSXSD1) data set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:tableSpecifications xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
  <table name="SALES" schema="BCKE">
    <distributionKey>
        <column name="STOREID"  />
        <column name="PRODUCTID" />
    </distributionKey>

  </table>
  <table name="PRODUCT" schema="BCKE" /> <!-- sub-elements are optional -->
</aqt:tableSpecifications>
```

This sample XML string defines the tables SALES and PRODUCT on the specified accelerator. Both tables belong to the BCKE schema. For the SALES table, a distribution key is selected, which consists of the STOREID and PRODUCTID columns. In addition, the CUSTOMERID and ORDERDATE columns are listed as organizing keys for the table. You can define a maximum of four columns in a distribution key.. For more information about distribution keys, see *Distribution keys* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

`<informationalUniqueKey>` is another element that you can specify within the `<table>...</table>` block. It is needed if you want to run the incremental update function on a table that does not have a unique key (primary key, primary index).

You can also use this element to let the query optimizer take advantage of informational unique constraints in case a table does not have a unique key. See <useConstraintsForQueryOptimization> for more information.

The column that you specify must contain unique values. If you specify more than one column, the combined values of these columns must be unique. If your table does not contain qualifying columns, you can add a hidden column to the source table and later specify this column with the `<informationalUniqueKey>` option. In this case, proceed as follows:

1. Add a hidden column to the source table, for example:

   ```
   ALTER TABLE table1
   ADD COLUMN unid BIGINT
   IMPLICITLY HIDDEN GENERATED ALWAYS AS IDENTITY
   (NO CYCLE NO ORDER CACHE 2000)
   ```

2. Run the REORG utility on this table. This causes Db2 for z/OS to assign unique and stable values to all existing rows, and generate new values for new rows.
3. Specify the newly added hidden column as the unique key to be used by SYSPROC.ACCEL_ADD_TABLES, for example:

   ```
   <?xml version="1.0" encoding="UTF-8" ?>
   <aqt:tableSpecifications xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
       version="1.0">
     <table name="table1" schema="BCKE">
       <informationalUniqueKey>
              <column name="unid" />
          </informationalUniqueKey>
     </table>
   </aqt:tableSpecifications>
   ```

`<useConstraintsForQueryOptimization>` adds primary keys and unique keys in the Db2 for z/OS catalog as informational unique constraints to the listed accelerator-shadow tables. It also enables all informational unique keys, including those explicitly defined by the user, for use by the query optimizer. Informational unique keys can improve the performance of accelerated queries when they are used by the optimizer.

If you omit the `<useConstraintsForQueryOptimization>` element, informational unique constraints are not used.

Important:

- If you set this option on one or more of the listed accelerator-shadow tables, you can no longer reload just a subset of the partitions of these tables (partial reload).
- If a table is replicated, and unique keys of the Db2 for z/OS table were removed from the catalog, you have to run SYSPROC.ACCEL_ALTER_TABLES to re-synchronize the Db2 for z/OS and the accelerator-shadow table. Otherwise, you might receive incorrect query results.

The use of constraints for query optimization can be switched on or off for each table listed in the `<tableSpecifications>` block. The value can be `true` or `false`. Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:tableSpecifications xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
  <table name="SALES" schema="TBSCH1">
    <informationalUniqueKey>
           <column name="STOREID" />
           <column name="PRODUCTID" />
```

```
       </informationalUniqueKey>
       <useConstraintsForQueryOptimization>
          true
       </useConstraintsForQueryOptimization>
  </table>
</aqt:tableSpecifications>
```

In this example, there is only one table called SALES. Setting `<useConstraintsForQueryOptimization>` to `true` allows the query optimizer to use the primary key of that table, any other unique keys defined for the table in the Db2 for z/OS catalog, and the informational unique key for performance optimization. The informational unique key defined by the user is a combined key consisting of two columns.

For more information on informational unique keys, see the description of the `<informationalUniqueKey>` element.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)
Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)
This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.
- The table names that you specify must exist in Db2 for z/OS.
- Accelerator-shadow tables must not yet exist for these tables.
- A table name must not appear more than once in the table specification.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 (needed for calling ADMIN_INFO_SYSPARM from the procedure)
- Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

This stored procedure inserts a row in the IBM Db2 Analytics Accelerator catalog table SYSACCEL.SYSACCELERATEDTABLES in Db2 for z/OS. On the accelerator, it updates the IBM Db2 Analytics Accelerator catalog. If the accelerator does not yet contain tables, SYSPROC.ACCEL_ADD_TABLES creates a database on the accelerator for the corresponding Db2 subsystem.

Results

After a successful run of the SYSPROC.ACCEL_ADD_TABLES stored procedure, you find the specified shadow tables on the accelerator, in *InitialLoadPending* state. For the various table states, refer to *Table states* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Environment variables
The job control language (JCL) for the configuration of the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures contains a data definition `(DD) "AQTENV"`. This data definition includes a data set in which environment variables are defined. These variables control the behavior of several stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_ALTER_TABLES

Changes the distribution key or organizing keys for an accelerator-shadow table or a set of such tables.

Details

To change the distribution key or organizing keys, a table on the accelerator must be in one of the following states:

- *InitialLoadPending*
- *Error*

- *Loaded*

In other words, do not call the procedure while a load operation or another ALTER TABLE operation is running on any of the tables in the specified set. To avoid interference with other ALTER TABLE and load operations, the *SchemaModificationInProgress* state is assigned to the tables in the set while SYSPROC.ACCEL_ALTER_TABLES is running.

After a successful completion, the tables are set to their previous states. If an error occurs during processing, the affected table is set to the *Error* state and the stored procedure returns an error message. Changes are not rolled back for already altered tables, and processing does not continue for the remaining tables. However, you can rerun the stored procedure with the same input after fixing the error.

Old keys remain valid until SYSPROC.ACCEL_ALTER_TABLES has run to completion. During an operation, the tables in the set can be queried as usual. Even concurrent queries are allowed.

The data types that can be used in distribution key columns or organizing key columns are limited. Not all data types are supported. For more information, see *Specifying or changing a distribution key or organizing keys* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

Important:

- Contrary to earlier product documentation, this product version does not allow you to change the distribution key or organizing keys of accelerator-only tables by running SYSPROC.ACCEL_ALTER_TABLES .
- Do not change the keys of more than one table at a time if tables with sizes of more than 1 GB are involved.
- An organizing key can only be changed after a table has been loaded because the key change effects a resorting of the table. And after the next reload, the organizing key and the sorting are lost, so that the key must be set again.

Syntax

```
CALL SYSPROC.ACCEL_ALTER_TABLES
(accelerator_name,
table_alter_specification,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_alter_specification

Specifies an XML input document that lists the tables or partitions to be updated. Any table on the accelerator that is not listed in the input XML document will not be updated.

Not only do you have to specify the keys that you want to add or change, but also those that you want to keep. Otherwise, the existing keys are removed.

The XML content must conform to the structure of the `tableSpecifications` element in the SAQTSAMP(AQTSXSD1) data set. For information on how to specify the input XML string, follow the link to the element content in the appendix at the end of this topic.

Sub-elements:

<informationalUniqueKey>

For more information, see the description of the `<informationalUniqueKey>` element for the SYSPROC.ACCEL_ADD_TABLES stored procedure.

Important:

- It is not possible at the moment to change an informational unique constraint once it has been selected.

- If you omit an informational unique constraint for a table, although you specified it for earlier runs of the stored procedure, you actually give instructions to remove the constraint.
- To make the query optimizer use an informational unique constraint you have defined, you must also set the `<useConstraintsForQueryOptimization>` element.

<useConstraintsForQueryOptimization>

This sub-element enables the use of informational unique constraints by the query optimizer. This includes informational unique constraints that are not explicitly set by a user (see sub-option `<informationalUniqueKey>`), but are based on the primary key and other unique keys of a table in the Db2 for z/OS catalog.

If you omit the `<useConstraintsForQueryOptimization>` element, informational unique constraints are not used.

Important:

- If you set this option on one or more of the listed accelerator-shadow tables, you can no longer reload just a subset of the partitions of these tables (partial reload).
- If a table is replicated, and unique keys of the Db2 for z/OS table were removed from the catalog, you have to run SYSPROC.ACCEL_ALTER_TABLES to re-synchronize the Db2 for z/OS and the accelerator-shadow table. Otherwise, you might receive incorrect query results.

The use of constraints for query optimization can be switched on or off for each table listed in the `<tableSpecifications>` block. The value can be `true` or `false`. Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:tableSpecifications xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
  <table name="SALES" schema="TBSCH1">
    <informationalUniqueKey>
           <column name="STOREID" />
           <column name="PRODUCTID" />
       </informationalUniqueKey>
       <useConstraintsForQueryOptimization>
          true
       </useConstraintsForQueryOptimization>
  </table>
</aqt:tableSpecifications>
```

In this example, there is only one table called SALES. Setting `<useConstraintsForQueryOptimization>` to `true` allows the query optimizer to use the primary key of that table, any other unique keys defined for the table in the Db2 for z/OS catalog, and the informational unique key for performance optimization. The informational unique key defined by the user is a combined key consisting of two columns.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *`accelerator_name`* parameter must specify an accelerator name that is listed in the catalog tables of the product.
- The tables must exist on the accelerator that the *`accelerator_name`* parameter specifies.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)).
- Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_ALTER_TABLES stored procedure does not modify data in Db2 for z/OS. On the accelerator, it updates the IBM Db2 Analytics Accelerator catalog and existing tables in the *Loaded* state.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_ARCHIVE_TABLES

Archives Db2 for z/OS table partitions in a storage area on an accelerator. The stored procedure is not suitable for accelerator-only tables.

Details

The SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure shifts the data of one or more table partitions in Db2 to an accelerator. The purpose of this is to reduce the disk space consumption of tables that are not updated anymore (historical data) on the Db2 side.

After archiving a partition, accelerated queries can only be run against the data in that partition if the GET_ACCEL_ARCHIVE=YES special register or ZPARM is used.

After archiving, the table data cannot be updated anymore, neither in the original Db2 partition, nor on the accelerator. This is because the original Db2 partitions are set to the Partition Read-Only (PRO) state, which prevents further updates.

Using this stored procedure, you can archive one or more partitions initially and add more partitions later on. It is possible to archive just a few partitions of a table and use the entire table data, including the archived data, on the accelerator for accelerated queries. For such an operation to work, however, the entire table must be defined and loaded as usual on the accelerator.

Tip: To avoid transferring partition data twice, archive partition data before the initial load.

Partitions that belong to the same table can be archived in parallel. By default, the data of up to four partitions can be transferred at the same time. To change this behavior, adjust the AQT_MAX_UNLOAD_IN_PARALLEL environment variable accordingly. Note that the Workload Manager settings apply in the same way as they do for the SYSPROC.ACCEL_LOAD_TABLES stored procedure.

It is possible to run accelerated queries while partition data is being archived. However, it cannot be determined whether this data is included in the query. Even if the the CURRENT_GET_ACCEL_ARCHIVE special register is set to include archived data, there is no guarantee until the archiving process has been completed.

During the process, Db2 image copies are created automatically. You can use these image copies to restore the original data manually, by using the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure, or by using the corresponding function in your administration client. You can create up to four image copies for each partition to be archived. A minimum of one image copy is required. To guarantee consistency between the partition data and the image copy, each partition is locked individually.

The stored procedure always creates up to four new image copies for each archived partition. The SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure uses the Db2 COPY utility to create up to two local images and two recovery images. (See the description of the `data-set-spec` clause of the Db2 for z/OS COPY Utility for more details).

You control the creation of image-copy data-sets by using template definitions as values for the following environment variables in the AQTENV data set:

- AQT_ARCHIVE_COPY1
- AQT_ARCHIVE_COPY2
- AQT_ARCHIVE_RECOVERYCOPY1
- AQT_ARCHIVE_RECOVERYCOPY2

See “Prerequisites” on page 14 for more information.

The removal process is called pruning and requires exclusive access to the tables in a partition. Before the actual pruning, the **`START DATABASE ACCESS(UT)`** command is run to give exclusive access to the Db2 utilities. The UT lock remains active when the stored procedure fails during the pruning phase, so that the respective partition is protected against accidental data changes. The lock also allows to resume the process (retry) where it has been left off.

The number of retries and the time interval between retries can be controlled by the AQT_MAX_RETRIES_DSNUTILU and the AQT_SECONDS_BEFORE_RETRY_DSNUTILU environment variables.

It is still possible to run into a situation in which the archiving operation cannot be completed. In this case, run the **Move Partitions to Accelerator** function in your administration client to display the status of each partition. Alternatively, you can run the following stored procedures to find out which partitions you must repair or process again:

SYSPROC.ACCEL_GET_TABLES_INFO
To identify tables with incompletely archived partitions

SYSPROC.ACCEL_GET_TABLES_DETAILS
To identify the partitions that require reprocessing

Despite these safety precautions, manual restore steps cannot always be avoided. For example, if an incomplete archiving job requires the restoration of a partition from an image copy, and an automatic restore operation by the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure or the corresponding function in your administration client fails, the original Db2 partition might be left in an unprotected state (no PRO state). In such a case, you might want to try a different image copy of the partition. This you can only do if you run a manual restore process with the help of Db2 utilities.

Resuming or starting a new archiving operation might be impossible as long as a failing partition has not been restored successfully.

Syntax

```
CALL SYSPROC.ACCEL_ARCHIVE_TABLES
(accelerator_name,
table_archive_specification,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_archive_specification
An XML input string that lists the tables and specifies the partitions to be archived. The string must conform to the structure of the `tableSetForArchiving` element in the SAQTSAMP(AQTSXSD1) data set.

A table name must not occur more than once in the list. If you specify partitions that have already been archived, the corresponding partitions are ignored during the operation.

You specify partitions by their logical or physical partition numbers, separated by commas (,). Ranges can be specified by using a colon (:) between the range boundaries. The range boundaries are inclusive. Using negative values, you can specify partitions by counting backwards from the upper end of the entire range of partition numbers. In this context, `-1` is used to identify the last partition of a table. In a sorted sequence of partition numbers, the position of a partition in the sequence can be determined according to the following formula:

```
(number of last partition) + 1 - n
```

where n is the number of partitions to be deducted from the upper end.

Table 1. Examples of specifications for partitions and partition ranges

XML specification for partitions	Description
`<partitions>1</partitions>`	Includes only the first partition
`<partitions>1:10</partitions>`	Includes the first 10 partitions of a table

Table 1. Examples of specifications for partitions and partition ranges (continued)

XML specification for partitions	Description
`<partitions>-10:-1</partitions>`	Includes the last 10 partitions of a table
`<partitions>1:-11</partitions>`	Includes all partitions except for the last 10
`<partitions>1,10:13,-1</partitions>`	Includes the partitions 1, partitions 10 through 13, and the last partition of a table

Examples:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:tableSetForArchiving xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
 usePhysicalPartitionNumbers="true"  version="1.0">
    <table name="SALES" schema="BCKE">
        <!-- explicitly specified partition numbers -->
        <partitions>1,5:10,20</partitions>
    </table>
    <table name="CUSTOMER" schema="BCKE" allArchivedPartitionsRetained="true">
        <!-- negative partition numbers count from the end, -1 is the last
             element example for "all partitions except last 10" -->
        <partitions>1:-11</partitions>
    </table>
    <table name="ORDER2009" schema="BCKE">
        <!-- archive all partitions of the table -->
    </table>
</dwa:tableSetForArchiving>
```

The sample XML string includes the attribute setting `usePhysicalPartitionNumbers="true">`, which means that the specified partition numbers are regarded as *physical* partition numbers.

Furthermore, the sample string lists three Db2 tables from which to archive partition data: SALES, CUSTOMER, and ORDER2009. From the SALES table, partition data is to be archived from the physical partitions 1, 5 through 10, and 20.

From the CUSTOMER table, the data of all partitions is to be archived, except for the last 10 partitions. Assume that this table has 20 partitions. According to the formula, this gives you 20 + 1 -11 = 10.

The entire data of the ORDER2009 table will be archived (all partitions). No `<partitions>` element is necessary here. The table name is sufficient.

`message`

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Prerequisites

The following conditions must be met if you want to archive partitions with the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure:

1. For the creation of the image copies, the AQTENV data set must be referenced by the Workload Manager (WLM) environment that has been set up for the IBM Db2 Analytics Accelerator for z/OS stored procedures, and a user with sufficient access rights must have set at least one of the following environment variables in this data set (a z/OS systems programmer or database administrator usually has the required access rights):

   - AQT_ARCHIVE_COPY1
   - AQT_ARCHIVE_COPY2
   - AQT_ARCHIVE_RECOVERYCOPY1
   - AQT_ARCHIVE_RECOVERYCOPY2

   These variables determine how many local image copies and recovery image copies are created for each archived partition and how the data sets are named. If you set a variable, a corresponding data set is created when partition data is archived. At least one of the variables must be set. Default values do not exist. Each value is a template specification as used in the Db2 TEMPLATE utility, for example:

```
AQT_ARCHIVE_COPY1 = &USERID..&DB..&TS..P&PART..&UNIQ.
```

where

&USERID.
ID of the user who runs SYSPROC.ACCEL_ARCHIVE_TABLES

&DB.
Name of the database that a partition resides in

&TS.
Name of the table space that the partition resides in

&PART.
Identifier of the (physical) partition. The letter P in the example is a text or string constant used as a prefix. This is required because &PART. resolves to a numeric value like 00001, and this not valid for qualifiers in a data-set name.

&UNIQ.
Causes the creation of a unique identifier

The template specification in the example could, for instance, result in the following image copy name:

```
BCKE.V4L1.BCKERTSE.CKRANGE3.P00001.D72R4KHN
```

- All template variables that are documented for the Db2 COPY utility can be used, with the exception of &SEQ (&SQ), &LIST (&LI), and &DSNUM.
- The chosen variables must ensure the uniqueness of image-copy data-set names. It is therefore recommended that you use at least the &PART. and &UNIQ. template variables.
- Templates must resolve to valid z/OS data set names.
- The template data-set names that you use must have been mapped to suitable data classes in the DFSMS.

Important: The AQT_ARCHIVECOPY_HLQ environment variable that was used with earlier versions of the High Performance Storage Saver is deprecated. Remove it from the AQTENV data set.

2. The stored procedure SYSPROC.ACCEL_ARCHIVE_TABLES creates image copies whose size might exceed 65535 tracks on direct access storage devices (DASDs). Therefore, a z/OS systems programmer must define and use a data class in the storage management subsystem (SMS) that supports larger sequential data sets and must associate this data class with the range of template data-set names for image copies. Data set types that can be used to this end are shown in Table 2 on page 15.

Table 2. Suitable data-set types for storage saver image copies and corresponding SMS data-class settings

Data-set type	Settings in SMS data class	Properties of data-set type
Extended-format data set	• `Data Set Name Type = EXT` • One of the following extension sub-parameter settings: – `Ext subparameter = R` (required) – `Ext subparameter = P` (preferred) • `Volume Count > 1`	• Requires multiple volumes • Maximum volume size still 65535 tracks

Table 2. Suitable data-set types for storage saver image copies and corresponding SMS data-class settings (continued)

Data-set type	Settings in SMS data class	Properties of data-set type
Large-format data set	`Data Set Name Type = LARGE`	• Multiple volumes possible, but not required • Volume sizes of more than 65535 tracks possible
Extended-attribute data set	`EATTR = OPT`	• Data sets must be stored on an extended-address volume (EAV) with at least 65521 cylinders. • Multiple volumes possible, but not required. A single volume might be sufficient.

To associate a suitable data class with the range of template data-set names, the z/OS systems programmer must create or modify a corresponding automatic class selection routine (ACS routine). The following example shows an excerpt of such an ACS routine, in which it has been assumed that the value of AQT_ARCHIVE_COPY1 starts with `IDAAHPSS`, as in `IDAAHPSS.&USERID..&DB..&TS..P&PART..&UNIQ.`:

```
FILTLIST LARGE INCLUDE(IDAAHPSS.**)
...
WHEN (&DSN = &LARGE) DO SET &DATACLAS = 'LARGE' ...
```

Tip: You can also define the ACS routine in such a way that the image copies created by the High Performance Storage Saver are stored on tape devices so that space can be saved on more expensive storage devices.

3. The table must be partitioned by range.
4. Anaccelerator-shadow table must have been defined by the SYSPROC.ACCEL_ADD_TABLES stored procedure or the corresponding function in your administration client.
5. The accelerator-shadow table must be in *InitialLoadPending* or *Loaded* state. This implies that incremental updates must be disabled for this table or that incremental updates must be stopped for the entire Db2 subsystem.

   Important: It is possible to have archived partitions while the corresponding table on the accelerator is still in *InitialLoadPending* state. To run accelerated queries against such a table, the remaining partitions of the table (that is, those that have not been archived) must be loaded as usual. Otherwise, the table cannot be enabled for query acceleration.
6. Db2 databases and table spaces that contain the table partitions to be archived must not be in *read-only* state, *stopped* state, or any other state that prevents an update of the data in the partition.

Restrictions

The following restrictions exist:

- This stored procedure cannot be used for accelerator-only tables.
- You must not invoke the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure more than once for the same table at the same time. While the procedure is running, the state of the accelerator-shadow table changes to *ArchivingInProgress*, which prevents an additional invocation for the same table on the same accelerator. However, this security mechanism does not work if you run the procedure simultaneously for the same table on different accelerators.
- Partitions of a particular table can be archived on more than one accelerator. To archive the same partition on another accelerator, the data in the image copy is used because the table space of the

partition was already emptied when the partition was archived for the first time. Archived partitions are protected against updating. If you have to update an archived partition none the less, first restore the data. For more information, see *Restoring archived partitions* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

- A subsequent stored-procedure run fails if columns were added or altered in the original table after the previous run.
- You can archive partitions that contain *Db2 archive tables*. However, you cannot archive *Db2 archive-enabled tables*. Do not confuse these tables with partitions and tables that were archived on an accelerator by the High Performance Storage Saver (accelerator-archive tables). If necessary, check the Db2 documentation for definitions of the terms *Db2 archive table* and *Db2 archive-enabled table*. Bear in mind also that when a Db2 archive table has been processed by the High Performance Storage Saver, you can no longer shift rows from the corresponding Db2 archive-enabled table to the Db2 archive table or vice versa. A message with `SQLCODE=-904` is returned when you try to do that.
- Partitions of tables in which one or more columns serve as parents in a foreign-key-relationship cannot be archived.
- Tables whose partitions are to be archived must not contain XML or LOB columns.
- Partitions to be archived must be allocated and in the *available* state. Other states, especially *check-pending*, will prevent a successful archiving of the partition data.
- If a range-partitioned table is index-controlled, the partitioning index must exist.
- Normally, you cannot rotate a Db2 partition or change partition boundaries if the data of this partition data has been archived on an accelerator. Commands to rotate a partition or change partition boundaries will be rejected by Db2 for z/OS. This restriction can be circumvented by first removing the Partition Read-Only (PRO) state with the help of the Db2 REPAIR utility. If you rotate a partition after removing the PRO state, you are still able to recover the archived partitions by using the image copies, except for those partitions that have been rotated out. If you change the partition boundaries, on the other hand, the image copies become unusable and the data cannot be restored. You will also end up with orphaned data on the accelerator.

  Attention: A rotation leads to the deletion of the data in those Db2 partitions that have been "rotated out". In addition, the corresponding image copies, which were created when the data was archived, will become unusable. The first archiving operation on the table that used to contain the "rotated -out" partitions will also delete the data of the "rotated-out" partitions from the accelerator, irrespective of the partitions that you specified for archiving.
- A rollback does not take place if errors occur during processing. Data that could be archived successfully before the error occurred remains on the accelerator. If the error occurred while the data was in the process of being archived, you can recover the data in most cases by rerunning the stored procedure. Only if this does not work, you must recover the data manually. To do so, follow this procedure:

  1. If data was already removed from the original partition in Db2, restore the data from the image copy with the help of the RECOVER utility.
  2. You can delete the image copy to free up space.
  3. You can also delete the records that pertain to the deleted image copy from the SYSIBM.SYSCOPY table with the help of the MODIFY RECOVERY utility.
- **Attention:** The PRO state is not maintained after a cold start of the Db2 for z/OS data server. If you do not set the PRO state manually after a cold start of Db2, you might lose the related image copy information in the SYSIBM.SYSCOPY catalog table because these records are protected only for the time that the PRO state is set. If you lose the image copy information, you might be unable to restore archived partitions. Therefore, if you must perform a cold start, complete the following steps before you resume regular INSERT, UPDATE, and DELETE operations:

  1. Run the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure against all tables with archived partitions.
  2. Search the XML output in the result set for occurrences of the partition attribute value *archiveErrorState = ArchivedParttionNotReadonly*.

Alternatively, you can start the **Move Partitions to Accelerator** or the **Restore Partitions to Db2** function in your administration client. In the dialog window that opens, partitions that must be repaired are marked as *not read only*.

3. For each archived partition not in the PRO state (not read-only), set the PRO state by running the REPAIR utility.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the `message` input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- Authorization to read the original DB2® base tables of the accelerator-shadow tables that are listed in the *`table_archive_specification`* parameter
- Authorization to run the UNLOAD utility on all affected DB2 tables.
- Authorization to run the **`-STOP DATABASE`** command on the Db2 for z/OS table spaces that contain the table partitions to be archived
- Authorization to run the **`-DISPLAY DATABASE`** command on all Db2 for z/OS databases that contain table partitions to be archived
- Authorization to run the **`-START DATABASE`** command on the Db2 for z/OS table spaces that contain the table partitions to be archived
- MONITOR1 authorization
- Authorization to run the COPY utility on all affected Db2 for z/OS tables
- Authorization to run the LOAD utility on all affected Db2 for z/OS tables
- Authorization to run the RUNSTATS utility on all affected tables
- Authorization to run the REPAIR utility on all affected Db2 for z/OS tables

- Read/write and execute access to the /tmp directory for the user who calls the stored procedure (UNIX System Services pipes are created in this directory)
- Privilege to connect to Db2 for z/OS via RRSAF.

  Important: If you want to grant the required DB2 privileges to secondary authorization IDs, for example groups, make sure that the DB2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests.
- Privilege to create data sets with the qualifier *<USERID>*.AQT, where *<USERID>* is the user ID of the person invoking the stored procedure
- ACCESS(ALTER) in RACF for COPY data sets
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

During archiving, the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure triggers or completes the following tasks:

- Original Db2 tables are unloaded to archive the data.
- A shared lock is held on the partition to be archived while data is being unloaded and while the image copies are being created.
- Image copies are created as described.
- Database tables are created on an accelerator for the partition data to be archived.
- After the data transfer to the accelerator, the data is removed from the original Db2 tables (pruning).
- A partition is set to the UT state during the pruning phase.
- Entries in the ARCHIVED column of the SYSACCEL.SYSACCELERATEDTABLES table are updated. If an accelerator-shadow table has been defined for accelerated queries and partitions of the same table have been archived on the same accelerator, the value in the ARCHIVED column is set to A. If a table exists on an accelerator, but partitions of this table were archived on a different accelerator, the value is set to C. If partitions of a table have not been archived at all, the ARCHIVED column contains a space character for this table.
- The SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure invalidates the Db2 dynamic statement cache for the tables to be archived.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

SYSPROC.ACCEL_GET_TABLES_DETAILS
Collects information about a set of tables with regard to data changes (data consistency) or archiving operations with the High Performance Storage Saver. The set of tables can include accelerator-only tables. For referencing accelerator-only tables, which were created for federated access, it returns the table name, the schema, and the table type.

SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES
Restores the data of archived partitions to their original locations. The stored procedure works on partitions that were archived with the High Performance Storage Saver (SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure).

Environment variables
The job control language (JCL) for the configuration of the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures contains a data definition `(DD) "AQTENV"`. This data definition includes a data set in which environment variables are defined. These variables control the behavior of several stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS

Invokes the RUNSTATS utility for an accelerator-shadow table, an accelerator-only table, or a set of such tables on an accelerator.

Details

To run the RUNSTATS utility successfully on an accelerator table, the table must be in one of the following states:

Accelerator-shadow tables	**Accelerator-only tables**
• *Loaded* • *ReplicationInProgress* for tables that are incrementally updated by the IBM Integrated Synchronization function of the product. The stored procedure does not work for tables updated by IBM InfoSphere® Change Data Capture for z/OS (CDC). See the Restriction note further down. • *UpdateInProgress* • *ArchivingInProgress* • *SchemaModificationInProgress*	• *Operational* • *UpdateInProgress*

Restriction: As soon as a Db2 subsystem is found for which CDC replication is enabled, SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS refuses to run and exits with message AQT20139E. This means that in case your accelerator is paired with more than one Db2 subsystem, the stored procedure will also not work for tables from the other Db2 subsystems, no matter if these subsystems are CDC-enabled or not.

If you run the stored procedure on an accelerator-shadow table that is in the process of being fully reloaded, the statistics are collected on the previous version of the table. That is, after the completion of the reload, the statistics of the newest version of the table might not be up-to-date. Therefore, complete a running reload before you run this stored procedure.

If you specify more than a single table as input for this stored procedure, the RUNSTATS utility is run in parallel. That is, a single execution thread is started for each table in the set. This might result in a high demand for CPU resources and system memory. To collect statistics on a large number of tables, it is therefore better to run the stored procedure several times in a sequence.

The SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure works synchronously. This means the procedure does not return results to the caller before all specified tables have been processed. This allows you to include this stored procedure in extract, transform, and load (ETL) processes. For example:

1. Insert rows into an accelerator-only table A.
2. Collect statistics on this table.

3. Run a second INSERT statement that selects rows in accelerator-only table A in order to insert these rows into accelerator-only table B.

Important usage notes

In most cases, it is not necessary to run this stored procedure because the accelerator automatically collects statistics when certain events occur:

early stats
The collection starts immediately after an initial table load.

copy stats
Statistics are copied from the initially loaded table during a full table reload.

auto stats
The collection is initiated by a daemon of the database engine that runs every two hours to monitor table changes. New statistics are collected on tables with many changes.

fabricated statistics
These statistics are generated during query execution if statistics are missing.

Generally, statistics should only be collected manually if tables were modified considerably, that is, if many updates occurred or if a significant amount of data was deleted or inserted. See the following list of occasions on which a manual collection of statistics might be useful:

- After you changed an accelerator-only table by INSERT, UPDATE, or DELETE operations and before you reference the table in a query.
- If tables were updated by the IBM Integrated Synchronization function because many INSERT, UPDATE or DELETE operations had to be synchronized, and you want to collect new statistics before any of the affected tables is referenced in a query.
- If you have opened a service request and IBM support asks you to collect statistics.

Important: A statistics collection requires considerable processing and memory resources. In addition, the catalog of the database engine must be updated, which might cause concurrency issues.

- Do not start too many statistics collections at the same time, as these might take resources away from other processes, such as queries or table load operations.
- Start a statistics collection only if the current statistics are outdated and if the performance is unsatisfactory, that is, if it takes comparably long to obtain results from the affected tables when these are referenced in queries.

For more information, see Table Statistics and IBM Db2 Analytics Accelerator for z/OS.

Syntax

```
CALL SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS
(accelerator_name,
table_statistics_specification,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_statistics_specification
An XML input string that lists the tables you want to collect statistics on. A table name must not occur more than once in the input string.

The XML string must conform to the structure of the `tableSetForStatisticsCollection` element in the SAQTSAMP(AQTSXSD1) data set. For information on how to specify the input XML string, follow the link to the element content in the appendix at the end of this topic.

Example:

```
 <?xml version="1.0" encoding="UTF-8"?>
<dwa:tableSetForStatisticsCollection
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <table name="NATION" schema="TPCH"/>
    <table name="REGION" schema="TPCH"/>
 </dwa:tableSetForStatisticsCollection>
```

The `<passThrough>` element is an optional element you can specify inside a `<table> </table>` block. You should only use it when IBM support advises it. The `<passThrough>` element allows you to add command options to the RUNSTATS statement that is finally executed.

With these command options, you can focus on specific table columns or column groups. You can select the columns or column groups that you want to collect statistics or distribution metrics on. You can also determine the sampling type as well as set, update, or reference a statistics profile.

For details, see the IBM Documentation topic on the RUNSTATS command.

Example:

```
<table schema="TPCH" name="NATION">
    <passThrough>
       WITH DISTRIBUTION AND SAMPLED DETAILED INDEXES ALL SET PROFILE ONLY
    </passThrough>
</table>
```

This results in the execution of the following RUNSTATS command on the accelerator:

```
RUNSTATS ON TABLE "DWA<location>"."NATION..."
WITH DISTRIBUTION AND SAMPLED DETAILED INDEXES
ALL SET PROFILE ONLY
```

Note: The syntax of the command options used within the `<passThrough>` element might change over time or vary with different product versions. A particular syntax you have used in the past might not be supported anymore in the future.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER

- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.
- The tables must exist on the accelerator that the *accelerator_name* parameter specifies.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)).
- Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure does not modify data in Db2 for z/OS. On an accelerator, it updates the runtime statistics of accelerator-shadow tables and accelerator-only tables.

Related information

RUNSTATS command

SYSPROC.ACCEL_CONFIG_OPTIMIZATION_PROFILE

Allows you to create, update, delete, or list optimization profiles for the accelerator database. An optimization profile can contain global optimization guidelines for improving query performance or specific optimization guidelines that apply to individual SQL statements. IBM support evaluates requests for optimization profiles in support cases and recommends a profile it is useful. IBM support also provides the content for such a profile.

Details

To run this stored procedure successfully, optimization profile support must have been set up on the accelerator (target) database. For more information, contact IBM support.

Syntax

```
CALL SYSPROC.ACCEL_CONFIG_OPTIMIZATION_PROFILE
(accelerator_name,
configure_command,
optimization_profile,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

configure_command

An XML input string that instructs the stored procedure to create, update, delete, or list optimization profiles. To determine the type of operation, set the `command` attribute in the XML string to the appropriate value:

- CREATE
- UPDATE
- DELETE
- LIST

Example: See the following example of an XML string for updating an optimization profile:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:configOptimizationProfileCommand
    xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0" >
       <optimizationProfileCommand
        command = "UPDATE"
        optimizationProfileSchema = "XYZSCH"
        optimizationProfileName = "PROF2" />
</dwa:configOptimizationProfileCommand>
```

If you want to run the LIST command, do not specify the `optimizationProfileSchema` and `optimizationProfileName` attributes. The XML string would then look like this:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:configOptimizationProfileCommand
    xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0" >
       <optimizationProfileCommand
        command = "LIST" />
</dwa:configOptimizationProfileCommand>
```

optimization_profile

The definition of the optimization profile to be created or updated. If you want to delete a profile or list existing profiles, leave this parameter out.

Example: See the following example of an XML string for updating an optimization profile. The instruction for the creation of a profile would look the same:

```
<?xml version="1.0" encoding="UTF-8"?>
<OPTPROFILE VERSION="9.5.0">
    <OPTGUIDELINES>
        <REGISTRY>
            <OPTION NAME="DB2_EXTENDED_OPTIMIZATION"
                VALUE="COL_RTABLE_UD_THR 0,
                XGBPART ON FULL NONHDIR,
                NI2NE_WITH_NULLS OUTER,
                COLJOIN 1,
                DISTINCT2GBY_SEL FORCE" />
        </REGISTRY>
    </OPTGUIDELINES>
</OPTPROFILE>
```

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

This stored procedure always returns three result sets:

First result set (OPTPROFINFO_CURSOR)

This result set is only populated if you run the LIST command and returns an XML string containing the optimization profile schema name, the selected optimization profile name and a list of the optimization profiles on the specified accelerator. The result set contains the following columns:

- SEQID of type INTEGER
- OPTIMIZATIONPROFILEINFO of type VARCHAR, with a maximum length of 32698 characters

Second result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of <message> elements (no truncation). The structure of the XML string conforms to that of the messageOutput element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

Optimization profiles must have been set up and enabled by IBM support for the database engine of the accelerator database.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- MONITOR1 authorization
- Read/write and execute access to the /tmp directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

During the execution, the stored procedure inserts, updates, or deletes entries from the SYSTOOLS.OPT_PROFILE table in the accelerator database. To list optimization profiles, the stored procedure reads from this table.

Related reference

The message parameter

The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member

Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E

The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSTOOLS.OPT_PROFILE table

Access plan and query optimization profiles

SYSPROC.ACCEL_CONTROL_ACCELERATOR

For a particular accelerator, this stored procedure retrieves information about the version of the installed components and their status. This stored procedure offers several functions to control an accelerator. It allows you to set the trace level, reset it to its default, retrieve trace settings, the entire trace content or other information, as well as remove trace data from an accelerator. In addition, by running this stored procedure, you can obtain a list of tasks currently running on the accelerator as well as cancel these tasks.

Syntax

```
CALL SYSPROC.ACCEL_CONTROL_ACCELERATOR
(accelerator_name,
 command,
 result,
 message)
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

command

An XML input string that specifies the functions that you want to use. The string must conform to the structure of the `controlCommand` element in the SAQTSAMP(AQTSXSD1) data set. The *command* parameter is an input parameter, which is defined as a CLOB with a maximum size of 256 KB.

Each element is linked to a user-defined function, for which access rights can be set individually. This allows you to grant the right to run specific functions of the stored procedure selectively. If a user does not have the right to use a particular element, that is, is not allowed to run the user-defined function invoked by the element, a message similar to this one is returned in the output XML string:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:messageOutput xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
   <message severity="error" reason-code="AQT10308I">
      <text>The user-defined function "DSNAQT.ACCEL_CONTROL_GETACCELERATORINFO"
            lacks EXECUTE authority for the caller of the stored procedure
            "SYSPROC.ACCEL_CONTROL_ACCELERATOR".
            This procedure was called with an input parameter value for
            "COMMAND" that requests the execution of the sub-function
            "getAcceleratorInfo".
       </text>
       <description>The requested sub-function within the stored procedure
            requires EXECUTE authorization on a separate user-defined function
            in Db2.
       </description>
```

```
        <action>Grant EXECUTE authority for the user-defined function to the
             caller of the stored procedure.
        </action>
    </message>
 </dwa:messageOutput>
```

XML Elements that can be used in the *command* `parameter`:

`<cancelTasks>`

Allows you to cancel a task that is running on the accelerator. The function requires the task ID as input. Provide the task ID in the `<cancelTasks>` element.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
    <cancelTasks>
        <task id="24" />
    </cancelTasks>
</aqt:controlCommand>
```

`<clearTraceData>`

Deprecated. It can still be run to avoid compatibility issues with product version 5.1.x, but has no effect in connection with product version 7.x.x. In version 5.1.x, this sub-function is used to remove trace data from the accelerator, such that the result of the next `<getTrace>` request does not contain previously collected data. The following files are removed:

- Trace files (because new trace files are created)
- Core dumps

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:controlCommand
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.2" >

       <clearTraceData/>

</aqttables:controlCommand>
```

`<getAcceleratorInfo>`

Retrieves status information about the accelerator. This information includes software version numbers and accelerator state information. You can also set an optional attribute, `includeEncryptionInfo = "true"`, which will include information about the encryption of data in motion. The information that is returned by the *`result`* output parameter is included in an XML element called `acceleratorInfo`, which is a child element of the root element `controlResult`. For more information, follow the link to the AQTSXSD1 data set at the end of this topic.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:controlCommand
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.4" >

       <getAcceleratorInfo includeEncryptionInfo = "true" />

</aqttables:controlCommand>
```

The output looks similar to the XML block in the following example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlResult
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
  <acceleratorInfo state="Online"
   authenticationTokenTimestamp="2017-11-24T08:39:40.545249Z"
   currentAcceleratorTimestamp="2017-11-24T08:44:35.196719Z"
   activeTraceProfile="DEFAULT" >

     <versionInformation>
```

```
        <component name="Accelerator" version="7.1.0.201711241547" />
        <component name="Accelerator container"
         version="ibmdashdb/local:v2.0.0-20171124-2243" />
        <component name="Access Server" version="11.4.10204" />
        <component name="Appliance software"
         version="1.0.0.0 [Build 20171124154706]" />
        <component name="BackendDBS" version="11.1.9.0
         [Build 1724041600]" />
        <component name="Linux Operating System"
         version="CentOS Linux release 7.2.1511 (AltArch)" />
        <component name="Replication BackendDBS JDBC Version" version="" />
        <component name="Replication DB2 JDBC Version" version="" />
        <component name="Replication Engine" version="11.4.0.Head_5432" />
     </versionInformation>

     <acceleratorSetting name="ENCRYPTION_AT_REST" value="enabled" />
     <acceleratorSetting name="ENCRYPTION_IN_MOTION" value="enabled" />
     <acceleratorSetting
      name="ENCRYPTION_IN_MOTION_CERTIFICATE_LIFETIME" value="90" />

     <acceleratorSetting name="MAXIMUM_NUMBER_CONCURRENT_RUNNING_SQL_TRANSACTIONS"
      value="110" />
     <acceleratorSetting name="SERIAL_NUMBER" value="4284629" />
     <acceleratorSetting name="SOFTWARE_UPDATE_APPLY" value="enabled" />
     <acceleratorSetting name="SOFTWARE_UPDATE_DEPLOY" value="enabled" />
     <acceleratorSetting name="SOFTWARE_UPDATE_LIST_DEPLOYED"
      value="enabled" />
     <acceleratorSetting name="SOFTWARE_UPDATE_REMOVE_DEPLOYED"
      value="enabled" />
     <acceleratorSetting name="LAST_REPLICATED_CHANGE_TIMESTAMP"
      value="2018-03-07 02:46:57.045000" />
     <acceleratorSetting name="REPLICATION_SUBSCRIPTION_NAME"
      value="DWAS45UY" />

     <replicationInfo state="STARTED"
      lastChangeTimestamp="2016-09-30T23:45:00.123456"
      latencyInSeconds="60">
         <sourceAgent insertCount="1" updateCount="2" deleteCount="3" />
         <targetAgent insertCount="11" updateCount="21" deleteCount="31"
          spilledRows="1234" spilledRowsApplied="234"/>
     </replicationInfo>

      <encryptionInformation>
         <encryptionOfDataInMotion>
           <certificates></certificates>
           <peers></peers>
          </encryptionOfDataInMotion>
       </encryptionInformation>

  </acceleratorInfo>
</aqt:controlResult>
```

Notes:

- Naturally, the setting of the `includeEncryptionInfo` attribute can have an effect only if encryption of data in motion is enabled.
- The `<replicationInfo>` block is shown only if the incremental update function is enabled.
- The `lastChangeTimestamp` attribute in the `<replicationInfo>` block shows the last time that the incremental update status changed from *Started* to *Stopped* or vice versa.
- The *replication latency* is disclosed through different channels, at different points in time. Sometimes, it is calculated by different systems. Values are reported:
  - In the **Accelerator** view as described in the previous table
  - When you run the Db2 command `-DIS ACCEL(<name>) DETAIL`
  - In trace files:

    ```
    <Event id="61098" timestamp="2018-11-28 18:07:51" level="W" addressee="0" >
            The current replication latency of XXXXXs  on DB2 location ADB2K04
    (subscription DWE9PLVY)
            has exceeded the threshold of 3600s.
    </Event>
    ```

  - In the status monitoring counter Q8STCRL

- As a result of running the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure in connection with the `<getAcceleratorInfo>` parameter.

These systems or reporting channels are not synced, so deviating values are inevitable. Deviations can be considerable, depending on the channel that is tapped and on the time when this is done. Stick to one reporting channel if you need to compare latency values.

`<getAcceleratorTasks>`

Retrieves a list of the tasks that are running on the accelerator. This information, especially the task IDs, can be used to cancel specific tasks. The information that is returned by the *`result`* output parameter is included in an XML element called `acceleratorTasks`, which is a child element of the root element `controlResult`. For more information, follow the link to the AQTSXSD1 data set at the end of this topic.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.2">
    <getAcceleratorTasks/>
</aqt:controlCommand>
```

`<getActivationLog>`

Retrieves information about previous software activations by the SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure (`ActivateDeployedPackage` subcommand).

Software updates are usually activated asynchronously, that is, after the completion of the SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure. If the accelerator cannot be reached during or after the activation of software updates, you can use this subcommand to check the activation progress and retrieve the latest log information.

If a software activation is ongoing at the time you run the stored procedure with the `<getActivationLog>` command, message `AQT10518I` is issued to indicate this.

In addition, the stored procedure reports the activation status by issuing an `AQT10517I` message for each affected component. Statuses are flagged as follows:

not_started
The activation has been scheduled, but not yet started.

started
The activation is currently in progress.

upgrade_skip_up_to_date
The activation of the update is unnecessary because the component is already up-to-date.

install_complete
The activation has been completed.

postinstall_complete
The activation has been completed and checks were applied successfully.

After a successful activation of software updates, the accelerator reboots. This takes about 30 minutes. If you run the stored procedure during that time, communication error `AQT10051E` is returned.

You can set an additional attribute, `diagnostics="true"` to include additional information in the third result set (identified by MESSAGES_CURSOR).

The input XML code must conform to the structure of the `controlCommand` element in the SAQTSAMP(AQTSXSD1) data set. The `<getActivationLog>` element must be specified. See the following example:

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
  <aqt:controlCommand
```

```
   xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <getActivationLog diagnostics="true" />
  </aqt:controlCommand>
```

<getAdditionalSupport planID="*<number>*" traceCommand="*<tcommand>*" timeoutInMinutes="*<number>*"

Generates Db2 for z/OS support documents or CDE performance trace information (CDE stands for *columnar data engine*). CDE performance trace information also includes the Db2 support documents for a given plan ID.

The information you collect is related to individual queries. Therefore, you need to obtain the plan ID of the query first. Run SYSPROC.ACCEL_GET_QUERIES2 for this purpose. You can use, for example, the following XML input string as the value of the **`query_selection`** parameter for SYSPROC.ACCEL_GET_QUERIES2:

```
<?xml version="1.0" encoding="UTF-8"?>
<dwa:querySelection xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
  <filter scope="all" />
</dwa:querySelection>
```

In the result set, you will find the plan IDs of the queries run recently on the specified accelerator. For example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:queryList xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.4">
<query user="SYSADM" planID="0">
```

In this example, the plan ID is 0.

Depending on the type of information you want to collect, specify one of the following strings (*`<tcommand>`*) as the value of the **`traceCommand`** attribute:

DB2SUPPORT
To collect Db2 for z/OS support documents.

CDE_PERFORMANCE_TRACE
To collect CDE performance data and Db2 for z/OS support documents.

SYSTEM_PERFORMANCE_DATA
To collect system performance data. This includes information about the utilization of the accelerator and the backend database engine, as well as information about SQL statements, accelerator tasks, and so on.

Optionally, you can specify a timeout value on the `timeoutInMinutes` attribute. If the stored procedure run cannot be completed during the specified period (in minutes), it is aborted, and you receive an `AQT10050E` error that includes the following diagnostic information:

```
AQT10050E - An internal error occurred on the 'XYZ001' accelerator:
ODBC operation executeQuery failed with SQLSTATE 22003 because
the timeout was reached, and the query execution was stopped.
Nevertheless, trace information could be collected successfully.
<ODBCDiagnostics>
    <SQLSTATE>22003</SQLSTATE>
    <SQLCODE>-904</SQLCODE>
    <Tokens num="0">
    </Tokens>
    <Message></Message>
</ODBCDiagnostics>
```

Input string example (value of command parameter):

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
    <getAdditionalSupport planID="0" traceCommand="DB2SUPPORT"
     timeoutInMinutes="0"/>
</aqt:controlCommand >
```

Output example:

The result set shows the file paths of the compressed trace files:

```
<?xml version="1.0" encoding="UTF-8" ?><aqt:controlResult
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
   <additionalSupport planID="0"
    absoluteAdditionalSupportZipFilePath="/head/dwa/var/log/additional_support_info/
    cdePerformanceAndDb2SupportForTask_314_PlanId_0.zip"
    fileSize="5463619"/>
```

Note: You can also use an option in IBM Db2 Analytics Accelerator Studio to collect CDE performance and system performance trace information. In the **Save Trace** window, select **Manually collected diagnostic information**.

<getReplicationEvents fromTimestamp="yyyy-mm-ddThh:mm:ss.µµµµµµZ" toTimestamp="yyyy-mm-ddThh:mm:ss.µµµµµµZ" minSeverity="INFORMATION">

Creates a report consisting of log information that was collected during incremental update activities. You can restrict the information to be returned by defining a timeframe using the `fromTimestamp` and `toTimestamp` attributes or by specifying the minimum severity of the information using the `minSeverity` attribute. This attribute can take on the following values (sorted by restrictiveness; most restrictive value first):

ERROR

All information whose severity is classed as *Error* or worse

WARNING

All information whose severity is classed as *Warning* or worse

INFORMATION

All information whose severity is classed as *Information* or worse

The information that is returned by the *`result`* output parameter is included in an XML element called `replicationEvents`, which is a child element of the root element `controlResult`. For more information, follow the link to the AQTSXSD1 data set at the end of this topic.

Example: The output looks similar to this block of XML code:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <aqt:controlResult xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.1">
      <replicationEvents>
         <event id="123456" message="Something went terribly wrong.
          I'll cancel the subscription." originator="SourceAgent"
          time="2011-01-11T10:33:42.487678Z" severity="ERROR" />
         <event id="123453" message="Start the subscription 0815 now"
          originator="TargetAgent" time="2011-01-09T08:21:42.487678Z"
          severity="INFO" />
      </replicationEvents>
   </aqt:controlResult>
```

`<getTraceConfig>`

Retrieves information about the trace configuration from the accelerator. This information includes:

- The currently active trace profile
- A list of all available trace profiles

The information that is returned by the *`result`* output parameter is included in an XML element called `traceConfig`, which is a child element of the root element `controlResult`. For more information, follow the link to the AQTSXSD1 data set at the end of this topic.

`<getTraceData>`

Retrieves diagnostic information for IBM support from the specified accelerator:

This information does not include trace information about stored procedures. Tracing for stored procedures is controlled by the *`message`* `INOUT` parameter.

The trace file generated by the stored procedure is a compressed archive, which includes other files.

The <content> element in the <getTraceData> element allows you to include or exclude certain types of information. The following values are supported:

Table 3. Possible values of the <content> element

Value	**Description**
ACCELERATOR	Collects accelerator and SQL engine logs, along with system diagnostics from the Docker container in which IBM Db2 Analytics Accelerator and the SQL engine run.
APPLIANCE	Collects system diagnostics from the physical environment of the Docker container, that is, the IBM Integrated Analytics System. **Note:** This option is only for IBM Db2 Analytics Accelerator on anIBM Integrated Analytics System. To generate similar trace information for Db2 Analytics Accelerator on Z, you must log in to the **Admin UI** of the Secure Service Container LPAR and create a *dump*. See *Creating a trace file (dump)* in the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide*.
ADDITIONALSUPPORTINFO	Retrieves support information collected during online support sessions with a client and saves these to file (one file per session). The files are saved to an ADDITIONALSUPPORTINFO folder. For more information, see Example C. In connection with this value, you can set the attribute sinceLastSaveOnly on the <getTraceData> parent element. Possible values are true or false. If set to true, the attribute causes the stored procedure to include only those support information files in the trace archive that were added since the previous generation of the archive. However, none of the older support information files are deleted from the ADDITIONALSUPPORTINFO folder; they are just not included in the trace archive to be generated. See Example D.
BACKENDENVIRONMENTFILE	Collects environment information about the target database on the accelerator if the element is specified in addition to ADDITIONALSUPPORTINFO.
CURRENTTASKS	Retrieves current accelerator task data, which is needed by the **Current Tasks** view of the accelerator monitoring function in your administration client.
DATAUSAGE	Retrieves information about currently processed data slices, which is needed by the **Data Slices** view of the accelerator monitoring function in your administration client.

Table 3. Possible values of the `<content>` element (continued)

Value	Description
`SYSTEMUTILIZATION`	Retrieves current accelerator usage data, which is needed by the accelerator monitoring function in your administration client.
`SYSTEMUTILIZATIONHISTORY`	Retrieves all available usage data for the selected accelerator. This data is used by the accelerator monitoring function in your administration client. This data is needed by the accelerator monitoring function in your administration client.
`TASKMANAGER`	
`TASKMANAGERHISTORY`	Retrieves all available information about completed accelerator tasks. This information is needed by the **Finished Tasks** view of the accelerator monitoring function in your administration client.
`REPLICATION`	Retrieves information about the current replication workload and the current replication latency. This data is needed by the **Replication** view of the accelerator monitoring function in your administration client.
`REPLICATIONHISTORY`	Retrieves all available information about the replication workload and the replication latency. This data is needed by the **Replication** view of the accelerator monitoring function in your administration client.

Example A: In the following example, the `<getTraceData>` element causes IBM Db2 Analytics Accelerator to return trace information for both, `ACCELERATOR` and `APPLIANCE`, in a result set:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
    <getTraceData keepConfiguration="true">
        <content>ACCELERATOR</content>
        <content>APPLIANCE</content>
    </getTraceData>
</aqt:controlCommand>
```

Example B: This example uses the `<getTraceData>` element in the XML string submitted as the value of the *command* parameter and the `<trace>` element in the XML string submitted as the value of the *message* input parameter. The aim is to generate two result sets, one containing trace information about IBM Db2 Analytics Accelerator and the SQL engine, the other one containing trace information about the stored procedures:

XML string for the *command* parameter:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
    <getTraceData keepConfiguration="true">
        <content>ACCELERATOR</content>
    </getTraceData>
</aqt:controlCommand>
```

XML string for the *message* input parameter:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:messageControl
xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">

   <traceConfig>
      <component name="PROCEDURE" level="DEBUG" />
   </traceConfig>

</aqt:messageControl>
```

Example C: In this example, the ADDITIONALSUPPORTINFO content element has been added to the <getTraceData> element in addition to ACCELERATOR and APPLIANCE. ADDITIONALSUPPORTINFO refers to a folder that holds information collected during WebEx support sessions with a client. The extra information is kept in this folder for a period of 30 days. If you use this element, the extra information is retrieved and written to the specified output location.

```
<aqt:controlCommand
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <getTraceData outputLocation="/u/bcke/trace.tgz">
       <content>ACCELERATOR</content>
       <content>APPLIANCE</content>
       <content>ADDITIONALSUPPORTINFO</content>
    </getTraceData>
</aqt:controlCommand>
```

The ADDITIONALSUPPORTINFO element does not work with product versions released before version 7.1.8. In older versions, the extra information that it provides is part of the output retrieved by the other two options (ACCELERATOR and APPLIANCE). Note, however, that even if you run version 7.1.8 with a back-level set of stored procedures, you must use the ADDITIONALSUPPORTINFO element, as it has become the only way to retrieve this kind of trace information.

Example D: In addition to ADDITIONALSUPPORTINFO, you can specify a BACKENDENVIRONMENTFILE element. Using this element will collect trace information about the environment of the target database on the accelerator. Code example:

```
<aqt:controlCommand
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <getTraceData outputLocation="/u/bcke/trace.tgz" sinceLastSaveOnly="true">
       <content>ADDITIONALSUPPORTINFO</content>
       <content>BACKENDENVIRONMENTFILE</content>
    </getTraceData>
</aqt:controlCommand>
```

The BACKENDENVIRONMENTFILE element can only be used if ADDITIONALSUPPORTINFO has also been specified.

Under the hood, the use of BACKENDENVIRONMENTFILE causes a run of the `db2support -cl 0` command, which gathers the environment information about the target database.

The attribute setting `sinceLastSaveOnly="true"` ensures that only support information files added since the last generation of the trace archive are included in the next version of the trace archive.

<setTraceConfig>

Allows you to change the trace configuration of the accelerator. Trace profiles determine the accelerator components to be traced and the trace detail level. They also determine the approximate size of the compressed trace data when this data is gathered by the getTraceData function. Furthermore, a profile determines whether a trace record is immediately written to disk (flush) or if it is cached in memory for better performance. Trace profiles can be selected and activated. Furthermore, you can define new profiles and remove existing ones (see examples).

Several trace profiles are preconfigured on the accelerator. You can retrieve available trace profiles by using the <getTraceConfig/> element.

Example: The following XML code instructs the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure to remove a trace profile named DEBUG_CONNECTIVITY, add a new profile with the same name, and activate this new profile. The trace information that is collected by the DEBUG_CONNECTIVITY profile has a default verbosity level of INFO. The trace-file size is limited to 2 MB. Old trace information will not be overwritten when this limit is reached. When trace information about the DRDA, CONTROLLER, and REPLICATION components is collected, the verbosity level changes to DEBUG, which means that all available information about these components will be collected. In addition, debug information will be collected from the accelerator.

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">

    <setTraceConfig>
        <removeTraceProfile name="DEBUG_CONNECTIVITY" />
        <addTraceProfile name="DEBUG_CONNECTIVITY" defaultLevel="INFO"
            traceFileSizeInMB="2" forceRecordFlush="false">
            <description>This is for debugging connectivity problems</description>
            <component name="DRDA" level="DEBUG" />
            <component name="CONTROLLER" level="DEBUG" />
      <component name="REPLICATION" level="DEBUG" />
        </addTraceProfile>
        <activateTraceProfile name="DEBUG_CONNECTIVITY" />
    </setTraceConfig>
</aqt:controlCommand>
```

Important: Using a trace profile other than the DEFAULT profile might lower the performance of the accelerator. Therefore, only use a different profile when instructed to do so by IBM support. Re-enable the DEFAULT profile when the support activities have been completed.

<startReplication/>

Starts the automatic transfer of data changes in Db2 for z/OS tables to the corresponding accelerator-shadow tables, provided that these tables have been enabled for incremental updates.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">

   <startReplication/>

</aqt:controlCommand>
```

<stopReplication/>

Stops incremental updates.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:controlCommand
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">

   <stopReplication/>

</aqt:controlCommand>
```

Restriction: If you stop replication inadvertently while a regular reload of a replication-enabled accelerator-shadow table is taking place (started by the load function or the SYSPROC.ACCEL_LOAD_TABLES stored procedure), the table state changes to *Suspended* after the load. That is, the table is treated like a faulty table. To re-enable incremental updates for this table, restart incremental updates on the accelerator and reload the table again.

<waitForReplication/>

Suspends the execution of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure until pending incremental updates are applied. These are updates that have been recorded in the DB2 log, but not yet propagated to the accelerator at the time that the stored procedure was called.

When this option is used, the stored procedure does not return results before the relevant tables are in sync with the DB2 log at the time the stored procedure was called. This ensures that the result data returned by SYSPROC.ACCEL_CONTROL_ACCELERATOR is based on the data matching this point in time. The incremental update function must have been started for this option to work properly.

You can specify an additional timeout attribute for this element (`timeoutInSeconds`). This prevents the execution of the stored procedure from being postponed endlessly, but has the adverse affect that you cannot be sure whether all pending updates have been applied. By default, this attribute is set to the value `0`. Setting it to a value greater than `0` makes the stored procedure return results when the relevant data has been replicated, or after the passing of the specified period, even if incremental updates have not finished at that time.

Example: The following XML code delays the execution of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure up to a maximum of ten minutes (600 seconds), which gives the incremental update function ten minutes to finish table synchronization. If the job cannot be completed during this period, the procedure returns results without waiting any longer for the incremental updates to finish.

```
<?xml version="1.0" encoding="UTF-8" ?>
   <aqt:controlCommand
    xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">

      <waitForReplication timeoutInSeconds="600" />

   </aqt:controlCommand>
```

Restriction: This option is not available in connection with IBM Integrated Synchronization. To use this option, the chosen incremental update technology must be IBM InfoSphere Change Data Capture for z/OS (CDC).

You can achieve a similar behavior in connection with IBM Integrated Synchronization if you run a WAITFORDATA query on a replicated table. When the query returns results, all committed data at a given point in time has been replicated, and you can be sure that SYSPROC.ACCEL_CONTROL_ACCELERATOR will take that data into account.

To this end, proceed as follows:

1. Create a dummy table, for example T_DUMMY in Db2 for z/OS and add a row to this table.
2. Add this table to an accelerator.
3. Enable replication for this table and load it.
4. Update the row in T_DUMMY.
5. Commit the update.
6. Submit the following SQL query:

   ```
   SET CURRENT QUERY ACCELERATION = ALL;
   SET CURRENT QUERY ACCELERATION WAITFORDATA n;
   SELECT COUNT(*) FROM T_DUMMY FOR FETCH ONLY;
   ```

   where *n* is the maximum time the query is postponed so that the latest incremental updates can be applied. It corresponds to the `timeoutInSeconds` attribute on the `<waitForReplication/>` element.

When the query results are returned, you have the guarantee that all table data on the accelerator matches the data in Db2 for z/OS at the time of the commit step in “5” on page 36.

Suggestion: Write a Db2 stored procedure that includes the steps “4” on page 36 through“6” on page 36 to preserve the details of the query.

For more information about WAITFORDATA queries, see *Making queries wait for incremental updates* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

result

Status and version information as a result of using the `<getAcceleratorInfo>` element.

Depending on the input parameter, this output parameter returns the following information:

Table 4. Output returned by result parameter

Input parameter (value of *command*)	**Output**
`<getTraceConfig/>`	Current® trace configuration
`<getAcceleratorInfo/>`	Software version of IBM Db2 Analytics Accelerator for z/OS and internal status
`<getAcceleratorTasks/>`	List of the tasks running on the specified accelerator.
`<getReplicationEvents/>`	Report of incremental update activities

This input XML file must conform to the structure of the `controlCommand` element in the SAQTSAMP(AQTSXSD1) data set. If you do not use any of the input parameters in Table 4 on page 37, the value of the *`result`* parameter is `NULL`.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

This stored procedure returns three result sets:

First result set (ACCELERATOR_TRACE_CURSOR)

If the sub-function `<getTraceData>` is used, the first result set contains the trace archive of the accelerator. If the sub-function is not used, the first result set will be empty. The cursor ACCELERATOR_TRACE_CURSOR identifies the first result set.

The information in the TRACEDATA column is used by IBM support for the analysis of error situations. It contains byte-encoded data. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *`message`* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)).
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- If an output file or data-set location is specified:

  Write access in RACF to the output data set for trace data or write access to the specified location in the z/OS UNIX file system.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure reads the catalog tables of IBM Db2 Analytics Accelerator in your database management system.

The trace configuration is changed if you use the following elements that are defined in the `controlCommand` element of the SAQTSAMP(AQTSXSD1) data set:

- `<setTraceConfig>`
- `<clearTraceData>`
- `<getTraceData>`

In contrast, the trace configuration is not changed if you use the following elements:

- `<getTraceConfig>`
- `<getAcceleratorInfo>`
- `<getAcceleratorTasks>`
- `<cancelTasks>`

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E

The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_CREATE_REFERENCE_TABLES

Creates table metadata in the catalog of a referencing Db2 subsystem and pointers (references) to accelerator-shadow tables or accelerator-only tables, so that tables defined in a remote Db2 subsystem can be queried.

Details

To run this stored procedure successfully, access to the original tables must have been granted by the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure or the equivalent function in your administration client.

The referencing Db2 subsystem must be paired with the same accelerator as the remote Db2 subsystem that provides the original tables. The pointers or references are created on this accelerator. The accelerator-shadow tables can then be queried from the referencing Db2 subsystem by means of SELECT statements.

Table metadata and pointers are created only if the table names in the specified set do not match the names of already existing tables. If such tables exist, they need not be created, but for the stored procedure to run error-free, the tables' schema names must match the schema names of the original tables in the remote Db2 subsystem. Otherwise, an error is returned.

Syntax

```
CALL SYSPROC.ACCEL_CREATE_REFERENCE_TABLES
(accelerator_name,
remote_location_name,
table_reference_set
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

remote_location_name
Location name of the remote Db2 subsystem owning the original tables. Location name means: the name of the subsystem as listed in the SYSIBM.LOCATIONS table. You can display the location name by running the `-DISPLAY DDF` Db2 command. For an example, see remote_location_name.

The remote Db2 subsystem must be paired with the accelerator specified by the *accelerator_name* parameter. The referencing Db2 subsystem might use a different name for that accelerator.

table_reference_set
An XML string that lists the original tables (accelerator-shadow tables or accelerator-only tables) that you want to reference. Table metadata and a pointer is created for each original table that you specify. The tables must have been defined on the specified accelerator by the SYSPROC.ACCEL_ADD_TABLES stored procedure, or, for accelerator-only tables, by the `CREATE TABLE ... IN ACCELERATOR` command. Names must not occur more than once in the list. The list or string must not be empty. The XML string has its own set of attributes. Some of these attributes are mandatory.

name (mandatory)
The name of the original table as specified in the SYSIBM.SYSTABLES catalog table of the remote Db2 subsystem.

schema (mandatory)

The schema name of the original table as specified in the SYSIBM.SYSTABLES catalog table of the remote Db2 subsystem.

userSpecifiedAotName (optional)

A different name that you can specify for a table reference to avoid naming conflicts. By default, the name of the reference is the same as the name of the original table.

If you reference tables that only exist in the remote Db2 subsystem, there is no need to specify a different table name because the table names do not yet exist in the catalog of the referencing Db2 subsystem. However, if tables exist in both, the remote Db2 subsystem and the referencing Db2 subsystem, you must specify a different table or schema name.

userSpecifiedAotSchema (optional)

A different schema that you can specify for a table reference to avoid naming conflicts. By default, the schema of the reference is the same as the schema of the original table.

userSpecifiedDatabase (optional)

The name of a Db2 subsystem that you want to store the referencing table metadata in. If not specified, the metadata is created in an implicitly defined Db2 subsystem.

Example:

```
<dwa:federatedTableSetInput
  xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <table name="ORDER" schema="TPCD2PCT"
               userSpecifiedAotName ="ORDER"
               userSpecifiedAotSchema="TPCD2PCT"/>
    <table name="PARTSUPP" schema="TPCD2PCT"
               userSpecifiedAotName ="PARTSUPP"
               userSpecifiedAotSchema="TPCD2PCT"/>
    <table name="SUPPLIER" schema="TPCD2PCT"
               userSpecifiedAotName ="SUPPLIER"
               userSpecifiedAotSchema="TPCD2PCT"/>
    <table name="NATION" schema="TPCD2PCT"
               userSpecifiedAotName ="NATION"
               userSpecifiedAotSchema="TPCD2PCT"/>
</dwa:federatedTableSetInput>
```

This example creates references to the remote tables ORDER, PARTSUPP, SUPPLIER, and NATION. The tables names and the schema stay the same.

`message`

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *`message`* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the

SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- Access to the original tables must have been granted by the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure or the equivalent function in your administration client.
- The accelerator that is specified by the `accelerator_name` parameter must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR stored procedure.
- The Db2 subsystem specified by the `remote_location_name` must be the location name of a remote subsystem connected to the same accelerator as the referencing Db2 subsystem (that is, the accelerator specified by `accelerator_name`)
- The table reference set (`table_reference_set` parameter) must point to existing tables in the remote Db2 subsystem.
- The SYSPROC.ACCEL_GRANT_TABLES_REFERENCE must have been run for all the tables in the table reference set because the privilege to access these tables is required. If the privilege is missing, table references cannot be created.
- The remote and the referencing Db2 subsystems must use the same coded character set identifier (CCSID).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- DBCTRL authority on the database explicitly specified by the stored procedure option `userSpecifiedDatabase`, or, if `userSpecifiedDatabase` is not specified, DBCTRL authority on database DSNDB04. You can also grant a higher authority that is not database-specific, such as SYSADM, system DBADM, and so on.
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- SELECT, INSERT, and UPDATE on SYSACCEL.SYSACCELERATEDTABLES
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

During execution, the stored procedure carries out the following tasks:

- It connects to the accelerator specified by the *`accelerator_name`* parameter and, to this end, uses the information in the Db2 for z/OS communications database.
- It verifies that each table in the table reference set (as specified by the `table_reference_set` parameter) exists in the remote Db2 subsystem that is connected to the specified accelerator, and that access to these tables by the referencing Db2 subsystem has been granted.

- For each table in the set, it creates a row in the SYSACCEL.SYSACCELERATEDTABLES table which includes the name of the associated accelerator.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES
Removes accelerator-only tables that were created by the SYSPROC.ACCEL_CREATE_REFERENCE_TABLES stored procedure for access by a referencing Db2 subsystem.

SYSPROC.ACCEL_GRANT_TABLES_REFERENCE
Grants a referencing Db2 subsystem the right to access tables that belong to a different, remote Db2 subsystem.

SYSPROC.ACCEL_GET_QUERIES

Deprecated (needed just for compatibility with product version 5). Wherever possible, use SYSPROC.ACCEL_GET_QUERIES2. The SYSPROC.ACCEL_GET_QUERIES stored procedure returns information about past queries and queries that are currently running on an accelerator. This includes information about queries against accelerator-only tables.

Details

The input XML string allows you to specify filtering conditions, determine the result order, and limit the number of results. Multiple filter conditions can be specified; a result is returned only if all conditions match.

All SQL statements in the resulting XML string are cut off after the first 128 bytes.

The SYSPROC.ACCEL_GET_QUERIES stored procedure returns only the queries that were run on the specified accelerator.

Syntax

```
CALL SYSPROC.ACCEL_GET_QUERIES
(accelerator_name,
query_selection,
query_list,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

`query_selection`
An XML input string containing the filter conditions to be used. The string must conform to the structure of the `querySelection` element in the SAQTSAMP(AQTSXSD1) data set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:querySelection
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <!-- filter conditions are combined with AND -->
    <filter scope="completed"
        fromTimestamp="2010-01-01T10:45:00Z"
    toTimestamp="2010-01-01T12:45:00Z"
    minElapsedTimeSec="120"
    minExecutionTimeSec="60"
    user="BJONES" />
    <result order="elapsedTime" maxRows="100"/>
</dwa:querySelection>
```

`query_list`

An output XML string containing the query information extracted from the accelerator. The structure of this string conforms to that of the `queryList` element in the SAQTSAMP(AQTSXSD1) data set.

`message`

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *`message`* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *`accelerator_name`* parameter must specify an accelerator name that is listed in the catalog tables of the product.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_GET_QUERIES2

Replacement for SYSPROC.ACCEL_GET_QUERIES. The stored procedure returns information about past queries and queries that are currently running on an accelerator. This includes information about queries against accelerator-only tables.

Details

The input XML string allows you to specify filtering conditions, determine the result order, and limit the number of results. Multiple filter conditions can be specified; a result is returned only if all conditions match.

In contrast to its predecessor, SYSPROC.ACCEL_GET_QUERIES2 returns all SQL statements in full length.

The SYSPROC.ACCEL_GET_QUERIES2 stored procedure returns only the queries that were run on the specified accelerator.

Syntax

```
CALL SYSPROC.ACCEL_GET_QUERIES2
(accelerator_name,
query_selection,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

query_selection
An XML input string containing the filter conditions to be used. The string must conform to the structure of the `querySelection` element in the SAQTSAMP(AQTSXSD1) data set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:querySelection
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <!-- filter conditions are combined with AND -->
    <filter scope="completed"
        fromTimestamp="2010-01-01T10:45:00Z"
    toTimestamp="2010-01-01T12:45:00Z"
    minElapsedTimeSec="120"
    minExecutionTimeSec="60"
    user="BJONES" />
    <result order="elapsedTime" maxRows="100"/>
</dwa:querySelection>
```

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure returns three result sets.

First result set (QUERY_INFO_CURSOR)

This result set includes an XML string that contains the query information that was extracted from the accelerator. The structure of this string conforms to that of the `queryList` element in the SAQTSAMP(AQTSXSD1) data set.

The result set is identified by a cursor named QUERY_INFO_CURSOR. The result set rows consist of the following columns:

1. SEQID INTEGER
2. QUERYINFO VARCHAR(32698)

The information in the QUERYINFO column is concatenated in ascending order of the values in the SEQID column. The column information is encoded in UTF-8.

Second result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Fourth result set

Each cell in the QUERY column contains query information in XML format. The information is provided in the form of XML attributes. These belong to the `<query>` element.

The `<query>` element has the following attributes:

applicationStallTime

This is the total time a query had to wait for its execution. The end of this period is marked by the return of the result set that was generated from the open query. This is when the query-submitting application became inactive, which is known as the application stall time.

prepareTimeSec

The time in seconds that was needed to prepare a SQL statement before the corresponding query could be processed.

sessionID

A number that identifies the session with the accelerator during which a query was processed.

timeoutsHTAP
The number of times queries had to be delayed because the WAITFORDATA delay period expired. This value is provided only if the accelerator was already configured to collect delay information related to WAITFORDATA settings.

waitTimeHTAPSec
The time a WAITFORDATA query had to wait for the completion of the next data replication process until it could be routed to the backend database on the accelerator. This value is provided only if the accelerator was already configured to collect delay information related to WAITFORDATA settings.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the /tmp directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The message parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_GET_QUERY_DETAILS

Deprecated (SYSPROC.ACCEL_GET_QUERY_DETAILS2 should be used with product version 7.1.0 or later). Retrieves the details of a past query or a currently running query. The stored procedure was kept for compatibility reasons. This stored procedure also works for queries against accelerator-only tables.

Details

This stored procedure retrieves information about past or active queries from an accelerator. The output is returned in a result set because the amount of data can be quite large. The result set contains the original SQL statement that was processed on the accelerator and internal accelerator information about the query. Unlike most other IBM Db2 Analytics Accelerator for z/OS stored procedures, SYSPROC.ACCEL_GET_QUERY_DETAILS does not return the result in the message output parameter.

This means that the information is stored in a global temporary table in Db2 for z/OS. The global temporary table is created by the following SQL statement during the installation of the stored procedures:

```
CREATE GLOBAL TEMPORARY TABLE DSNAQT.ACCEL_QUERY_INFO
(SEQID INTEGER, QUERYINFO VARCHAR(32698)) CCSID UNICODE)
```

The result set contains the SQL query code followed by a newline character and the plan text. All information is returned in a single, long VARCHAR string. If the length of this string exceeds 32698 characters, it is split into multiple result-set rows. To create a usable single string, the multiple result-set rows are concatenated in the order of the values in the SEQID column (ascending).

You can enable tracing for the SYSPROC.ACCEL_GET_QUERY_DETAILS stored procedure. If this is done, the trace information is also returned in a result set. Hence you might see two result sets. The first will always contain the query information whereas the second contains the trace information.

Syntax

```
CALL SYSPROC.ACCEL_GET_QUERY_DETAILS
(accelerator_name,
plan_id,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

plan_id
The identifier (plan ID) of the query that you want to obtain details about. You can run the SYSPROC.ACCEL_GET_QUERIES stored procedure to obtain a list of these identifiers. The plan IDs of past or currently active queries are shown in the *query_list* output XML string.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure returns three result sets:

First result set
See description under "Details" on page 46.

Second result set (SP_TRACE_CURSOR)
Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)
This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of <message> elements (no

truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

AQT10081E
A plan with an ID of *planid* cannot be found on the accelerator.

AQT20101E
The file integrity could not be verified for the *fileName* file. The file was transferred using the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage`** command. The hash value *callerHash* that was provided does not match the hash value computed on the accelerator. Checksum output of the accelerator: *computedHash*

SYSPROC.ACCEL_GET_QUERY_DETAILS2

The stored procedure retrieves the details of a past query or a currently running query. This stored procedure also works for queries against accelerator-only tables.

Details

This stored procedure retrieves information about past or active queries from an accelerator. The output is returned in a result set because the amount of data can be quite large. The result set contains the following information:

- Full SQL statement of the query, as it was routed to the accelerator
- Remote query execution plan in a compressed, base64-encoded format
- The internally used remote query execution plan in textual (db2exfmt) format

Syntax

```
CALL SYSPROC.ACCEL_GET_QUERY_DETAILS2
(accelerator_name,
plan_id,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

plan_id
The identifier (plan ID) of the query that you want to obtain details about. You can run the SYSPROC.ACCEL_GET_QUERIES2 stored procedure to obtain a list of these identifiers. The plan IDs of past or currently active queries are shown in the output XML string in the first result set of SYSPROC.ACCEL_GET_QUERIES2.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure returns five result sets:

First result set (QUERY_INFO_CURSOR)
This result set contains the original SQL statement. The result set is identified by a cursor named QUERY_INFO_CURSOR. The result set rows consist of the following columns:

1. SEQID INTEGER
2. QUERYINFO VARCHAR(32698)

The information in the QUERYINFO column is concatenated in ascending order of the values in the SEQID column. The column information is encoded in UTF-8.

Example:

```
Result set QUERY_INFO_CURSOR:
SEQID QUERYINFO
---------------
0     SELECT '123456789 223456789 323456789 423456789
      523456789 623456789 723456789 823456789 923456789
      A23456789 B23456789 C23456789 D23456789 ', COUNT(*) FROM
      BCKE.TQRNGE2
```

Second result set (QUERY_PLAN_CURSOR)
This result set contains the remote query execution plan in compressed base64 format. The result set is identified by a cursor named QUERY_PLAN_CURSOR. It consists of the following columns:

1. SEQID INTEGER
2. PLAN VARCHAR(32698)

The information in the PLAN column is concatenated in ascending order of the values in the SEQID column. The column information is encoded in UTF-8.

Example:

```
Result set QUERY_PLAN_CURSOR:
SEQID PLAN
----------
0     UEsDBBQACAgIABVhfkoAAAAAAAAAAAAAAAAjAAAAdGFwSW5mb18
      yMDE3XzAzXzMwXzEyXzA4XzQyXzQ5NS5hcGV9WVVQHGzPXdzdfX
      Ho0iKlpcWluLu7U9ytUNx ...
```

Third result set (QUERY_PLANTEXT_CURSOR)

This result set contains the remote query execution plan in textual (db2exfmt) format. The result set is identified by a cursor named QUERY_PLANTEXT_CURSOR. It consists of the following columns:

1. SEQID INTEGER
2. PLAN VARCHAR(32698)

The information in the PLAN column is concatenated in ascending order of the values in the SEQID column. The column information is encoded in UTF-8.

Example:

```
Result set QUERY_PLANTEXT_CURSOR
SEQID PLAN
----------
0     Connecting to the Database.
      Db2 Universal Database Version 11.1, 5622-044
      (c) Copyright IBM Corp. 1991, 2016
      Licensed Material - Program Property of IBM
      IBM DATABASE 2 Explain Table Format Tool
```

Fourth result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Fifth result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

AQT10081E
A plan with an ID of *planid* cannot be found on the accelerator.

AQT20101E
The file integrity could not be verified for the *fileName* file. The file was transferred using the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage`** command. The hash value *callerHash* that was provided does not match the hash value computed on the accelerator. Checksum output of the accelerator: *computedHash*

SYSPROC.ACCEL_GET_QUERY_EXPLAIN

Retrieves Db2 EXPLAIN information about a query from an accelerator. This stored procedure also works for queries against accelerator-only tables.

Details

The EXPLAIN information can be used to generate and display an access plan graph for the query, which provides valuable information for query optimization.

To identify a query, the stored procedure checks the entries in DSN_QUERYINFO_TABLE. Make sure that this table exists in your Db2 subsystem before using the stored procedure. For information on how to create this table, see the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

The SYSPROC.ACCEL_GET_QUERY_EXPLAIN stored procedure returns the EXPLAIN information in a result set, in the form of a base64-encoded, compressed archive.

You can enable tracing for the SYSPROC.ACCEL_GET_QUERY_EXPLAIN stored procedure. If this is done, the trace information is also returned in a result set. Hence you might see two result sets. The first will always contain the EXPLAIN information whereas the second contains the trace information.

Syntax

```
CALL SYSPROC.ACCEL_GET_QUERY_EXPLAIN
(query_number
query_block_number,
explain_time,
explain_schema,
message);
```

Options description

query_number
The number that identifies a query. The specified number must match an entry in the QUERY_NUMBER column of DSN_QUERYINFO_TABLE.

query_block_number
The number identifying the query block of the query.

explain_time
The time when the query was submitted. The specified number must match a timestamp in the EXPLAIN_TIME column of DSN_QUERYINFO_TABLE.

explain_schema
The table schema that DSN_QUERYINFO_TABLE belongs to.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure returns three result sets.

First result set
The first result is a block of XML code that shows the location of a compressed trace file (zip file). This trace file contains Db2 support information. It is the same trace file that you get when you run SYSPROC.ACCEL_CONTROL_ACCELERATOR with the <getAdditionalSupport> command parameter and the `traceCommand` attribute set to the value DB2SUPPORT. Running SYSPROC.ACCEL_GET_QUERY_EXPLAIN instead has the advantage that you need not identify the plan ID first because SYSPROC.ACCEL_GET_QUERY_EXPLAIN is already bound to a particular query (see the description of the query_number parameter).

Output example:

```
<?xml version="1.0" encoding="UTF-8" ?><aqt:controlResult
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
   <additionalSupport planID="0"
    absoluteAdditionalSupportZipFilePath="/head/dwa/var/log/additional_support_info/
    cdePerformanceAndDb2SupportForTask_314_PlanId_0.zip"
    fileSize="5463619"/>
```

In this example, a trace file called `cdePerformanceAndDb2SupportForTask_314_PlanId_0.zip` has been written to the `/head/dwa/var/log/additional_support_info` directory on the accelerator.

Second result set (SP_TRACE_CURSOR)
Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

SYSPROC.ACCEL_GET_TABLES_DETAILS

Collects information about a set of tables with regard to data changes (data consistency) or archiving operations with the High Performance Storage Saver. The set of tables can include accelerator-only tables. For referencing accelerator-only tables, which were created for federated access, it returns the table name, the schema, and the table type.

Details

For tables partitioned by growth or by range, information is returned for each partition of the table. If the table is not partitioned, information is returned for the whole table. The information is returned in a result set.

Syntax

```
CALL SYSPROC.ACCEL_GET_TABLES_DETAILS
(accelerator_name,
table_set,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_set

A list of the tables that you want to process. The XML string that you specify as input for this parameter must conform to the structure of the `tableSet` element in the SAQTSAMP(AQTSXSD1) data set.

The tables must be accelerator-shadow tables that were defined by the SYSPROC.ACCEL_ADD_TABLES stored procedure, or accelerator-only tables. The same table name must not appear more than once in the table set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <dwa:tableSet xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
      <table name="SALES" schema="SCHM" />
      <table name="PRODUCT" schema="SCHM" />
   </dwa:tableSet>
```

The sample XML code instructs the stored procedure to work on the SALES and PRODUCT tables.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure returns three result sets:

First result set (TABLES_DETAILS_CURSOR)

The first result set contains details about the detected table changes. It is identified by the cursor TABLES_DETAILS_CURSOR.

The table change information in the TRACEDATA column consists of an XML string. If the length of the string exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Third result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

The stored procedure returns different outputs for accelerator-shadow tables and for tables (partitions) that were archived with the help of the High Performance Storage Saver. The following details are returned:

Table 5. Information returned by the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure

Change information (accelerator-shadow tables compared with original DB2 tables)	**Information about accelerator-archive tables**
• Timestamp of last update for each table partition. This information is returned in the `lastLoadTimestamp` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. • Change category. The following categories exist: **NONE** No change occurred. Referencing accelerator-only tables, which were created for federated access, also fall into this category. **RELOAD_RECOMMENDED** Data changes have occurred that suggest a reload of the accelerator-shadow tables or partitions. **RELOAD_REQUIRED** The table or table partitions are out of sync. A reload is enforced.	• Timestamp of the archiving operation. This information is returned in the `archiveTimestamp` XML attribute of the `archiveInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. • Amount of archived data in MB. This information is returned in the `dataSizeInMB` XML attribute of the `archiveInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. • Names of the image-copy data-sets. This information is returned in the `type` attribute of the `backupImage` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. For more information about the elements and attributes in the SAQTSAMP(AQTSXSD1) data set, follow the link to the data-set transcript at the end of this topic.

Table 5. Information returned by the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure (continued)

Change information (accelerator-shadow tables compared with original DB2 tables)	**Information about accelerator-archive tables**
• **UNKNOWN** Data changes have occurred, but information about these cannot be obtained. The changes might be due to an unusual event or error. It is recommended that you manually review the data changes and then reload the accelerator-shadow tables or partitions in question. The marker is mostly set if one of the following events took place: – The REORG utility was run after the last load and the original table was not updated since then. The REORG utility resets the change indicators in the SYSIBM.SYSTABLESPACESTATS table so that changes cannot be detected. It is therefore recommended that you run REORG before you load or reload a table. **Important:** This situation is handled differently by DB2 11 for z/OS in so far as NONE rather than UNKNOWN is returned as the change information (support for LASTDATACHANGE). Exception: If REORG was run on a table space that is partitioned by growth and contains more than one partition, the situation is as described here, and a reload is recommended. – A DB2 table was not modified, but another DB2 table in the same table space was. • **UNKNOWN (continued)** – Relevant realtime statistics cannot be found in Db2 for z/OS. This happens, for example, if a table space was restored with the RECOVER utiltiy, but the REORG utility was not run afterwards. – The table space of a table is partitioned by growth, and the table was defined on the accelerator before migrating to version 3 of IBM Db2 Analytics Accelerator. In this case, you must remove the accelerator-shadow table from the accelerator and then redefine it to update the metadata of the table. This information is returned in the `category` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.	

Table 5. Information returned by the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure (continued)

Change information (accelerator-shadow tables compared with original DB2 tables)	**Information about accelerator-archive tables**
• Type of the data change (except when the change indicator is `UNKNOWN`). The following types of changes are detected: – SQL statements INSERT, UPDATE, DELETE, and TRUNCATE TABLE – Utility executions LOAD RESUME, LOAD REPLACE (for both LOG YES and LOG NO), and RECOVER – ADD PARTITION and ROTATE PARTITION – Partition boundary changes due to ALTER TABLE or REORG REBALANCE operations – Replacement of table data with data from a clone table (EXCHANGE DATA operation) – `FederatedReference` for referencing accelerator-only tables This information is returned in the `type` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. • Approximate amount of data to be transferred during a reload in MB (except when the change indicator is `UNKNOWN`). The estimate is based on the most recent Db2 statistics. The newer the statistics, the more accurate the estimate will be. **Note:** This figure is useful as a metric for reloads only if the corresponding table space contains just one table. If there are more tables, this figure gives you the total size of all tables in the table space. This information is returned in the `dataSizeInMB` XML attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set. The `changeInformation` element contains more attributes that return additional information. Follow the link to the transcript of the SAQTSAMP(AQTSXSD1) data set at the end of this topic for details.	

Important:

- Accelerator-shadow tables in *InitialLoadPending* or *Error* state will always be reported as changed because they do not appear to be loaded.
- If a table has been archived by the High Performance Storage Saver, it will be reported as changed by all accelerators on which it is defined. In such a case, the report will return RELOAD_REQUIRED in the change category and enforce a reload. The reload will then delete the table data from all accelerators,

except the accelerator that holds the accelerator-archive table. This is because a DB2 table can be archived to a single accelerator only. Other copies of the same table on different accelerators (accelerator-shadow tables) will reflect the fact that the archived data has been removed from the original DB2 table.

Information in the first result set

This is basically change information (accelerator-shadow tables compared with original Db2 tables):

- Timestamp of last update for each table partition.

  This information is returned in the `lastLoadTimestamp` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.
- Change category. The following categories exist:

  NONE
  Data changes did either not occur, or did occur, but information about these cannot be obtained. The changes might be due to an unusual event or error. It is recommended that you manually review the data changes and then reload the accelerator-shadow tables or partitions in question. The marker is mostly set if one of the following events took place:

  - The REORG utility was run after the last load and the original table was not updated since then. The REORG utility resets the change indicators in the SYSIBM.SYSTABLESPACESTATS table so that changes cannot be detected. It is therefore recommended that you run REORG before you load or reload a table.
  - A Db2 table was not modified, but another Db2 table in the same table space was.
  - Relevant realtime statistics cannot be found in Db2 for z/OS. This happens, for example, if a table space was restored with the RECOVER utiltiy, but the REORG utility was not run afterwards.

  This information is returned in the `category` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.

  RELOAD_RECOMMENDED
  Data changes occurred that suggest a reload of the accelerator-shadow tables or partitions.

  RELOAD_REQUIRED
  The table or table partitions are out of sync. A reload is enforced.

  UNKNOWN
  Data changes occurred, but information about these cannot be obtained. The changes might be due to an unusual event or error. It is recommended that you manually review the data changes and then reload the accelerator-shadow tables or partitions in question. The marker is mostly set if REORG was run on a table space that is partitioned by growth and contains more than one partition. In cases like this, a reload is recommended.

  This information is returned in the `category` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.
- Type of the data change (except when the change indicator is `NONE` or `UNKNOWN`). The following types of changes are detected:

  - SQL statements INSERT, UPDATE, DELETE, and TRUNCATE TABLE
  - Utility executions LOAD RESUME, LOAD REPLACE (for both LOG YES and LOG NO), and RECOVER
  - ADD PARTITION and ROTATE PARTITION
  - Partition boundary changes due to ALTER TABLE or REORG REBALANCE operations
  - Replacement of table data with data from a clone table (EXCHANGE DATA operation)

  A corresponding value or indicator is returned in the `type` attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.

If you add a column to a Db2 table or change a column definition, or drop a Db2 table and then re-create it, the corresponding accelerator-shadow table becomes invalid. You must therefore remove it from the accelerator and add it again before you can reload and use it for query acceleration.

- Approximate amount of data to be transferred during a reload in MB (except when the change indicator is `UNKNOWN`). The estimate is based on the most recent Db2 statistics. The newer the statistics, the more accurate the estimate will be.

  Note: This figure is useful as a metric for reloads only if the corresponding table space contains just one table. If there are more tables, this figure gives you the total size of all tables in the table space.

  This information is returned in the `dataSizeInMB` XML attribute of the `changeInformation` element, which is a child element of the `tableSetDetails` element in the SAQTSAMP(AQTSXSD1) data set.

The `changeInformation` element contains more attributes that return additional information. Follow the link to the transcript of the SAQTSAMP(AQTSXSD1) data set at the end of this topic for details.

Important: Accelerator-shadow tables in *InitialLoadPending* or *Error* state will always be reported as changed because they do not appear to be loaded.

Examples of the first result set (associated with TABLES_DETAILS_CURSOR)

The following sample result sets contain change information about a range-partitioned table. The entire information (XML) is written to the TABLES_DETAILS column in the result set table. The relevant information is found in the `<part>` elements. The first example contains change information about a table without archived partitions; the second information about a table with archived partitions.

Information about the change category (for tables with and without archived partitions) is found in the `category` attribute of the `<changeInformation>` element. The other information is found in the other attributes of the same element. That is, the timestamp of the last load is in the `lastLoadTimestamp` attribute, the change type in the `type` attribute, and the amount of data to be transferred during a reload in the `dataSizeInMB` attribute. The `sharedTablespace` attribute tells you whether the table space is shared between multiple tables.

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqt:tableSetChanges xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0"
    <table name="TEST_BY_RANGE" schema="MYDWA"/>

       <partInformation type="BY_RANGE"
          <column name="COL1"/>
       </partInformation>

       <part dbmsPartNr="2" logicalPartNr="1" endingAt="2011-10-31">
          <changeInformation category="RELOAD_RECOMMENDED"
           lastLoadTimestamp="2012-01-09T11:53:27.997141Z"
           type="DataChanged" sharedTablespace="false" dataSizeInMB="123"/>
       </part>

       <part dbmsPartNr="3" logicalPartNr="2" endingAt="2011-11-30">
          <changeInformation category="NONE"
           lastLoadTimestamp="2012-01-09T11:53:27.997141Z"
           sharedTablespace="false" dataSizeInMB="105"/>
       </part>

       <part dbmsPartNr="4" logicalPartNr="3" endingAt="2011-12-31">
         <changeInformation category="UNKNOWN"
          lastLoadTimestamp="2012-01-09T11:53:27.997141Z"
          type="ReorgWithNoSubsequentDataUpdate" sharedTablespace="false"
          dataSizeInMB="175" />
       </part>

       <part dbmsPartNr="1 logicalPartNr="4" endingAt="2011-01-31">
          <changeInformation category="RELOAD_REQUIRED"
           lastLoadTimestamp="2012-01-09T11:53:27.997141Z"
           type="PartitionAddedOrRotated" sharedTablespace="false"
           dataSizeInMB="142" />
       </part>
    </table>
```

For tables with archived partitions, you find the archiving information in the `<archiveInformation>` elements. There, you find the timestamp of the archiving operation in the `archiveTimestamp` attribute, the amount of transferred data in the `dataSizeInMB` attribute, and the name of the image copy inside the `<backupImage>` element:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:tableSetDetails xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
version="1.0">

   <table name="TEST_BY_RANGE" schema="MYDWA"/>

         <partInformation type="BY_RANGE"
            <column name="COL1"/>
         </partInformation>

         <part dbmsPartNr="2" logicalPartNr="1" endingAt="2011-10-31">
            <changeInformation category="NONE" type="NoChangeDetected"
             dataSizeInMB="0" />
         </part>

         <part dbmsPartNr="3" logicalPartNr="2" endingAt="2011-11-30">
            <changeInformation category="NONE"
             lastLoadTimestamp="2012-01-09T11:53:27.997141Z"
             dataSizeInMB="105"/>
         </part>

         <part dbmsPartNr="4" logicalPartNr="3" endingAt="2011-12-31">
            <archiveInformation archiveTimestamp="2012-03-26T17:27:00.000000Z"
             dataSizeInMB="135">
               <backupImage>ARCHIVE.DA11.DB000022.CUSTOMER.P0003</backupImage>
            </archiveInformation>
         </part>

         <part dbmsPartNr="1" logicalPartNr="4" endingAt="2011-01-31">
            <changeInformation category="RELOAD_REQUIRED"
             lastLoadTimestamp="2012-01-09T11:53:27.997141Z">
             type="PartitionAdded" dataSizeInMB="142" />
         </part>

         <part endingAt="2011-09-30">
            <archiveInformation archiveTimestamp="2012-03-26T17:27:00.000000Z"
             dataSizeInMB="13765">
               <backupImage>ARCHIVE.DA11.DB000022.CUSTOMER.P0001</backupImage>
            </archiveInformation>
         </part>

     </table>
```

If a table in the specified set is a referencing accelerator-only table, which has been created for federated access, the result for that table looks similar to the XML string in the following example:

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
<aqt:tableSetDetails
 xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.2"
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011">
  <table name="T1" schema="SYSADM">
     <changeInformation category="NONE" type="FederatedReference"/>
  </table>
</aqt:tableSetDetails>
```

Restrictions

Data changes might not be detected correctly under the following conditions:

- Db2 table data was modified by a non-IBM utility.
- A Db2 table space was restored by the DSN1COPY utility.
- The table space of the SYSIBM.SYSTABLESPACESTATS table was stopped.
- Db2 for z/OS ended abnormally without being able to return real-time statistics.
- The amount of archived data that is reported by the stored procedure in the `dataSizeInMB` XML attribute is the amount that was actually transferred. This size might differ from the amount that was reported elsewhere before the transfer because the data size of partitions that have not been archived is an estimate based on DB2 statistics.

- Furthermore, data changes might not be detected correctly under the following conditions:
  - DB2 table data was modified by a non-IBM utility.
  - A DB2 table space was restored by the DSN1COPY utility.
  - The table space of the SYSIBM.SYSTABLESPACESTATS table was stopped.
  - Db2 for z/OS ended abnormally without being able to externalize real-time statistics.
- The reported change information does not account for incremental updates. If data changes are detected for a table that is currently enabled for incremental updates, the reported change type (`DataChangeInReplicatedTable`) indicates that the changes might have already been transferred to the accelerator.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- Authorization to run the DISPLAY DATABASE command for the databases that hold the tables listed in the *`table_specification`* parameter
- Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics)
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure saves all runtime statistics to disk (if the required PTF is installed).

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_GET_TABLES_INFO

For all or a subset of the tables on an accelerator, this stored procedure returns the XML specifications that were used to define these tables using the SYSPROC.ACCEL_ADD_TABLES stored procedure. In addition, this stored procedure returns status information about the specified tables including accelerator-only tables.

Syntax

```
CALL SYSPROC.ACCEL_GET_TABLES_INFO
(accelerator_name,
table_set,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_set
A list of the tables that you want to process. The XML string that you specify as input for this parameter must conform to the structure of the `tableSet` element in the SAQTSAMP(AQTSXSD1) data set.

The tables must be accelerator-shadow tables that were defined by the SYSPROC.ACCEL_ADD_TABLES stored procedure, or accelerator-only tables. The same table name must not appear more than once in the table set.

You can set this parameter to NULL, in which case the stored procedure returns information about all tables that are defined in the SYSACCEL.SYSACCELERATEDTABLES table or that are present on the specified accelerator. This allows you to detect orphaned or incompletely deleted tables, which are listed with an integrity status of *ViolatedOnAccelerator* or *ViolatedInDatabaseManagementSystem* in the TABLE_INFORMATION element of the result set. You cannot restore such tables, but the rests of such tables can be cleaned up by using the ACCEL_REMOVE_TABLES stored procedure.

Note: An accelerator-shadow table that is enabled for replication by the IBM Integrated Synchronization function does not have to be removed when the original Db2 table is dropped. It is removed automatically. That is, there will not be an orphaned table on the accelerator.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <dwa:tableSet xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
      <table name="SALES" schema="SCHM" />
      <table name="PRODUCT" schema="SCHM" />
   </dwa:tableSet>
```

The sample XML code instructs the stored procedure to work on the SALES and PRODUCT tables.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

This stored procedure returns four result sets:

First result set (TABLE_INFO_SPEC_CURSOR)
The first result set contains a table specification that can be used as input for the *table_specification* parameter of the SYSPROC.ACCEL_ADD_TABLES stored procedure. The result set is identified by the cursor TABLE_INFO_SPEC_CURSOR. The result set can be empty if no tables exist or missing if an error occurred. It consists of the following columns:

- SEQID of type INTEGER
- TABLE_SPECIFICATION of type VARCHAR, with a maximum length of 32698 characters

The XML data in the TABLE_SPECIFICATION column conforms to the structure of the `tableSpecifications` element in the SAQTSAMP(AQTSXSD1) data set.

Second result set (TABLE_INFO_STATES_CURSOR)

The second result set contains status information about the set of tables on the accelerator. It is identified by the cursor TABLE_INFO_STATES_CURSOR. The result set can be empty if no tables exist or missing if an error occurred. It consists of the following columns:

- SEQID of type INTEGER
- TABLE_STATES of type VARCHAR, with a maximum length of 32698 characters

Each cell in the TABLE_STATES column contains table status information or table statistics in XML format. The information is provided in the form of XML attributes. These belong to the `<status>` and `<statistics>` elements.

The `<status>` element has the following attributes:

loadStatus

The value of the `loadStatus` is a table state. For more information, see *Table states* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*. Consider the peculiarities of the following states in case they are output by the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure:

FederatedTableOperational

The output is slightly different, depending on the Db2 subsystem that sent the stored procedure call: The result set includes the load status if the call comes from the Db2 subsystem that owns the tables; if the call comes from the referencing Db2 subsystem, the load status is always `FederatedTableOperational`.

Furthermore, the output includes the location name of the remote Db2 subsystem if you call the stored procedure from the referencing Db2 subsystem. This is not the case if you call the stored procedure from the remote Db2 subsystem because the tables are local tables if viewed from the perspective of the remote Db2 subsystem.

Operational

This value is reserved for accelerator-only tables, and indicates that the table is ready for processing.

The stored procedure does not list or create output for a table if the creation of the accelerator-only table has not been committed. Likewise, the procedure might still list and produce output for a table whose removal has not been committed.

type

The value of the `type` can be one of the following:

AcceleratorOnly

Indicates that the table is an accelerator-only table.

FederatedReference

Indicates that the table is a referencing accelerator-only table that was created for federated access.

Regular

Indicates that the table is an accelerator-shadow table.

accelerationStatus

An that shows whether acceleration is enabled for a table. A value of 1 means enabled; 0 means disabled.

integrityStatus

An that indicates whether the table deviates structurally from its counterpart in Db2 for z/OS. To this end, the information about the table in SYSACCEL.SYSACCELERATEDTABLES is compared to that information about the same table in the catalog of the accelerator.

replicationStatus

An that shows whether incremental updates are enabled for the table. A value of 1 means enabled; 0 means disabled.

replicationDetails
Information about the currently ongoing replication process for the table. It indicates whether replication was progressing as expected at the time the stored procedure returned the results or whether errors were encountered up to that point in time.

archiveStatus
An attribute that shows whether the table has been archived by the High Performance Storage Saver on the specified accelerator or on another accelerator. The term *specified accelerator* refers to the accelerator that you specified as an input parameter for this stored procedure. A value of 1 means *archived*; a value of 0 *not archived*.

archiveProblemsDetected
This attribute indicates whether problems occurred while archiving the table, such as partitions that have been incompletely archived or restored. You can obtain more detailed information about the problems by running the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure. Some problems can only be detected by SYSPROC.ACCEL_GET_TABLES_DETAILS, so a value of *false* does not necessarily mean that no problems exist. It just means that no problems have been detected by SYSPROC.ACCEL_GET_TABLES_INFO.

archiveSynchronizationStatus
This attribute contains a more detailed description of the **`archiveStatus`**. Its value is **`archiveSynchronizationStatusType`**, which takes on one of numerous other values detailing the archiving state. If you are interested in these values, search for `archiveSynchronizationStatusType` in the Appendix F, "Transcript of the SAQTSAMP(AQTSXSD1) data set member," on page 237.

The `<statistics>` element has the following attributes:

usedDiskSpaceInMB
The amount of disk space that is taken up by the table in MB.

skew
The skew value is a metric for the distribution imbalance of table rows across the worker nodes. The value is the difference between the smallest and the largest portion of the table data in megabytes (MB).

organizedPercent
The percentage of the table data that is organized based on the specified organizing keys. For organized tables, this value is typically 100, and 0 for tables that are not organized.

lastLoadTimestamp
Timestamp that indicates the last successful load of the table. Because accelerator-only tables are not loaded by IBM Db2 Analytics Accelerator functions, the attribute shows the table state (for example *Organized*) for this type of table.

archiveBackendStatisticsCollectionTimestampCollected
An attribute that indicates whether statistics were (ever) collected for an archived table. A value of 1, indicating that statistics were in fact collected, can occur only for tables that have been archived by the High Performance Storage Saver. Otherwise, the value is 0.

archiveBackendStatisticsCollectionTimestamp
A timestamp attribute that shows the last time statistics were collected for an archived table.

tableBackendStatisticsCollectionTimestampCollected
An attribute that indicates whether statistics about the table were (ever) collected. A value of 1 means yes; 0 means no.

tableBackendStatisticsCollectionTimestamp
A timestamp attribute that shows the last time statistics were collected for the table.

lastAccessTimestamp
An attribute that shows the last time the table was used (accessed) by a query. The attribute is not returned if the table has just been loaded, but not yet used by a query.

accessCount

An attribute that shows how many times the table was accessed by queries since its definition on the accelerator. The attribute is not returned if the table has just been loaded, but not yet used by a query.

capturePointTimestamp

Shows the time of when the last replication update started.

archiveDiskSpaceInMB

The amount of disk space occupied by an archived table.

archiveRowCount

The number of archived table rows.

The output XML document conforms to the structure of the `tableInformation` element in the SAQTSAMP(AQTSXSD1) data set.

Third result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Fourth result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Example of first result set (TABLE_INFO_SPEC_CURSOR)

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:tableSpecifications xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
  <table name="CUSTOMER" schema="ABCTS1">

  </table>
  <table name="TIME24" schema="ABCTS1" /> <!-- sub-elements are optional -->
</aqttables:tableSpecifications>
```

What you see is the content or value of the *table_specifications* parameter that was used when you defined the table by using the SYSPROC.ACCEL_ADD_TABLES stored procedure. This example shows the original XML specifications for two tables, CUSTOMER and TIME24. Both tables belong to the ABCTS1 schema.

Example of second result set (TABLE_INFO_STATES_CURSOR)

```
<aqt:tableInformation xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
     version="1.5">
    <table schema="ABCTS1" name="CUSTOMER">
        <status loadStatus="Loaded" type="Regular"
         accelerationStatus="1" integrityStatus="Unimpaired"
         replicationStatus="0" archiveStatus="0" />
        <statistics usedDiskSpaceInMB="5" skew="0" organizedPercent="0"
            lastLoadTimestamp="2018-03-09T08:23:19.710632Z"
            tableBackendStatisticsCollectionTimestampCollected="1"
            archiveBackendStatisticsCollectionTimestampCollected="0"
            tableBackendStatisticsCollectionTimestamp="2022-11-21T12:17:52.888358Z"
            lastAccessTimestamp="2023-01-23T09:14:06.300476Z"
            accessCount="748"
            archiveDiskSpaceInMB="0" >
            <statisticsProfile>RUNSTATS ON TABLE "ABCXYZ11"."CUSTOMER-ID_180794-V2"
                ON ALL COLUMNS WITH DISTRIBUTION ON ALL COLUMNS AND SAMPLED
                DETAILED INDEXES ALL
            </statisticsProfile>
        </statistics>
    </table>

    <table schema="ABCTS1" name="TIME24">
        <status loadStatus="ReplicationInProgress" type="Regular"
         accelerationStatus="1" integrityStatus="Unimpaired"
         replicationStatus="1" replicationDetails="Active"
         archiveStatus="0" />
        <statistics usedDiskSpaceInMB="8" skew="0.713489"
         organizedPercent="0"
         lastLoadTimestamp="2018-03-08T12:27:52.688616Z"
         tableBackendStatisticsCollectionTimestampCollected="1"
         archiveBackendStatisticsCollectionTimestampCollected="0"
         tableBackendStatisticsCollectionTimestamp="2022-11-21T12:17:52.888358Z"
         lastAccessTimestamp="2023-01-23T09:14:06.300476Z"
         accessCount="748"
         capturePointTimestamp="2018-03-08T12:27:52.688616Z"
         archiveDiskSpaceInMB="0" />
       </table>

   <table schema="ABCTS1" name="TEST3A">
        <status accelerationStatus="1" archiveStatus="0"
         integrityStatus="Unimpaired"
         loadStatus="FederatedTableOperational"
         replicationStatus="0" type="FederatedReference"/>
        <statistics organizedPercent="0.000"
         skew="0.000" usedDiskSpaceInMB="0"
         tableBackendStatisticsCollectionTimestampCollected="1"
         archiveBackendStatisticsCollectionTimestampCollected="0"
         tableBackendStatisticsCollectionTimestamp="2021-11-21T12:17:52.888358Z"
         lastAccessTimestamp="2022-08-22T14:04:21.543614Z"
         accessCount="242" />
    </table>

</dwa:tableInformation>
```

The example shows status information about three tables, CUSTOMER, TIME24, and TEST3A.

In the information about the CUSTOMER table, the `<statisticsProfile>` element contains the information that was passed in by use of the `<passThrough>` element in connection with the SYSPROC.ACCEL_COLLECT_STATISTICS stored procedure.

The `loadStatus` of the CUSTOMER table is *Loaded*. The acceleration status is *1*, which means the table can be used for accelerated queries in its current state.

The integrity status is *Unimpaired*, meaning that the information about this table in the SYSACCEL.SYSACCELERATEDTABLES table in Db2 is consistent with the information about the same table in the catalog of the accelerator. Were the table not mentioned in the accelerator catalog, but marked as eligible for accelerated queries in the SYSACCEL.SYSACCELERATEDTABLES table, its integrity status would be *ViolatedOnAccelerator*. If, on the other hand, the table were listed on the accelerator, but not marked as eligible for accelerated queries, its status would be *ViolatedInDatabaseManagementSystem*. The latter can happen if rows were accidentally deleted from the SYSACCEL.SYSACCELERATEDTABLES table or if a backup of the Db2 catalog was restored that did not contain the accelerator information at the time when the backup was created.

The table uses 5 MB of disk space. The skew value of `0` indicates an even distribution of rows across the processing nodes. A day in November 2022 was the last time statistics were collected for the table. The table has not yet been archived by the High Performance Storage Saver. The last query that accessed the table was run in January 2023. In total, the table was accessed by queries 748 times.

Note: The number and the data size of High Performance Storage Saver rows are computed asynchronously so that the display of changes might be delayed for some time after the High Performance Storage Saver operations.

You can interpret the information about the TIME24 table analogously. The main difference is that this table is enabled for incremental updates. For that reason, the `replicationEnabled` attribute shows the value 1. The `capturePointTimestamp` attribute shows when the last update started.

The TEST3A table is a referencing accelerator-only table, which has been created for federated access. You can see that its load status is `FederatedTableOperational` and that its type is `FederatedReference`.

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Messages

The following messages are likely to occur in connection with the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure:

AQT10000I

Success message indicating that the stored procedure could provide the requested table information in the result sets.

AQT10076W

Information about tables in Db2 for z/OS and on the accelerator is out of sync. As a result, one or more tables cannot be detected on either side. To solve the problem, remove the tables in question from the accelerator and then redefine these if they still exist in the Db2 subsystem.

AQT10507W

Original tables were altered so that the column definitions in Db2 for z/OS and on the accelerator might be out of sync.

This warning is not issued if columns were dropped from a Db2 for z/OS table. Because the dropping of columns results in the deletion of column metadata from the Db2 for z/OS catalog, the change cannot be tracked anymore.

AQT10134E

The SYSPROC.ACCEL_GET_TABLES_INFO stored procedure failed because one or more of the specified tables are not found in the Db2 for z/OS catalog table SYSIBM.SYSACCELERATEDTABLES.

The following scenario shows you which messages to expect under which conditions.

Table 6. Disparate table sets in the database and on the accelerator

Database	Accelerator
T1	

Table 6. Disparate table sets in the database and on the accelerator (continued)

Database	Accelerator
T2	T2
	T3

Assume that you have the following set of tables in your database and on the accelerator:

In this case, you will receive the following messages, according to the input value of the *table_set* parameter:

Table 7. Error messages resulting from disparate table sets

Value of *table_set*	Output returned by *table_specifications* and *table_information* for tables	Messages
NULL	T1, T2, T3	AQT10000I Warning AQT10076W for table T1, which cannot be found on the accelerator Warning AQT10076W for table T3. The table exists on the accelerator, but since it has not been specified as input (*table_set* is NULL), AQT10076W is returned instead of AQT10134E.
T1	T1	AQT10000I Warning AQT10076W because table T1 cannot be found on the accelerator
T2	T2	AQT10000I
T3	No output	Error AQT10134E for table T3, which is not found in SYSIBM.SYSACCELERATEDTABLES
T4	No output	T4 does not exist at all. In cases like this, just error message AQT10134E is issued although this only indicates that the table is not found in SYSIBM.SYSACCELERATEDTABLES.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

AQT10076W

The *tableSchema*.*tableName* table on accelerator *acceleratorName* cannot be found in the catalog of the *unknownOnSide* or has been altered, such that the table definitions on the accelerator are not longer in synch with the definitions in the database.

AQT10134E
The *acceleratedTableSchema*.*acceleratedTableName* table does not exist on the *acceleratorName* accelerator.

AQT10507W
At least one column of the *tableSchema*.*tableName* table has been altered, added, or removed in Db2 for z/OS after defining the table on the *acceleratorName* accelerator.

SYSPROC.ACCEL_GRANT_TABLES_REFERENCE

Grants a referencing Db2 subsystem the right to access tables that belong to a different, remote Db2 subsystem.

Details

The referencing Db2 subsystem must be paired with the same accelerator as the remote Db2 subsystem that contains the requested tables.

The referencing Db2 subsystem acquires the right to access the tables by means of SELECT statements.

The grant is necessary, but not sufficient to reference the tables in queries. To reference the tables in queries from the referencing Db2 subsystem, you must also run the ACCEL_CREATE_REFERENCE_TABLES stored procedure from that subsystem.

Syntax

```
CALL SYSPROC.ACCEL_GRANT_TABLES_REFERENCE
(accelerator_name,
remote_location_name,
table_set
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

remote_location_name
Location name of the referencing Db2 subsystem that will access the tables. Location name means: the name of the subsystem as listed in the SYSIBM.LOCATIONS table. You can display the location name by running the `-DISPLAY DDF` Db2 command.

Example:

```
-DISPLAY
DDF

********************************* Top of Data *********************************
DSNL080I  -DB11 DSNLTDDF DISPLAY DDF REPORT FOLLOWS:
DSNL081I STATUS=STARTD
DSNL082I LOCATION           LUNAME            GENERICLU
DSNL083I XWB1701            DEIBMIPS.IPXWB17  -NONE
DSNL084I TCPPORT=11511 SECPORT=0     RESPORT=11611 IPNAME=-NONE
DSNL085I IPADDR=::9.137.18.112
.
.
```

In this example, the location name is `XWB1701`.

The referencing Db2 subsystem must be paired to the accelerator specified by the *`accelerator_name`* parameter. The owning Db2 subsystem might use a different name for that accelerator.

table_set

An XML string that lists the tables to which you want to grant access. The tables must have been defined by running the SYSPROC.ACCEL_ADD_TABLES stored procedure, or, for accelerator-only tables, by running the `CREATE TABLE ... IN ACCELERATOR` command. Names must not occur more than once in the list. The list or string must not be empty.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<dwa:tableSet xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
  <table name="NATION" schema="TPCH"/>
  <table name="ORDERS" schema="TPCH"/>
</dwa:tableSet>
```

The tables in the set must have been defined on the accelerator specified by the *`accelerator_name`* parameter. They must exist in the owning Db2 subsystem from which this stored procedure is run.

`message`

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *`message`* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

Ensure that the proper Db2 for z/OS program temporary fix (PTF) is installed. The catalog table SYSACCEL.SYSACCELERATEDTABLESAUTH must exist.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure

- EXECUTE on the SYSACCEL.* packages
- INSERT on SYSACCEL.SYSACCELERATEDTABLESAUTH
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
- One of the following privileges:
  - SELECT privilege with GRANT option on every table in the *table_set* that is used with the stored procedure.
  - Access to the predefined RACF profile DB2ACCEL.FEDERATION. To this end, it is required that:
    1. The stored procedure runs in a WLM environment with an APF-authorized address space. That is, the load module for the stored procedure must reside in an APF-authorized library.
    2. A RACF profile DB2ACCEL.FEDERATION exists in the RACF resource class for Db2 (DSNR).

       Example:

       ```
       -- Setup profile DB2ACCEL.FEDERATION with group FEDADMG
       SETR CLASSACT(DSNR)
       RDEFINE DSNR (DB2ACCEL.FEDERATION) OWNER(DB2OWNER) UACC(NONE)
       PERMIT DB2ACCEL.FEDERATION CLASS(DSNR) ID(FEDADMG) ACCESS(READ)

       SETR RACLIST(DSNR) REFRESH
       ```

       The `ID` can be a group (here: `FEDADMG`), a single user, or a list of users. A sample job including the profile definition is provided in the SAQTSAMP.AQTSFEDR data set member.

Accessed data and systems

During execution, the stored procedure carries out the following tasks:

- It connects to the accelerator specified by the *accelerator_name* parameter and, to this end, uses the information in the Db2 for z/OS communications database.
- For each table listed in the table set, it creates an entry in the SYSACCEL.SYSACCELERATEDTABLESAUTH catalog table.
- It verifies that each new entry in SYSACCEL.SYSACCELERATEDTABLESAUTH. is associated with the accelerator specified by the `accelerator_name` parameter.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

SYSPROC.ACCEL_CREATE_REFERENCE_TABLES
Creates table metadata in the catalog of a referencing Db2 subsystem and pointers (references) to accelerator-shadow tables or accelerator-only tables, so that tables defined in a remote Db2 subsystem can be queried.

SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE
Revokes the access to a set of tables that was granted by running the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure.

SYSPROC.ACCEL_LOAD_TABLES

Loads data from the source tables in Db2 into the corresponding accelerator-shadow tables.

Details

The stored procedure supports the following load scenarios:

- Load of a previously defined accelerator-shadow table in *InitialLoadPending* state
- Reload of table partitions or non-partitioned accelerator-shadow tables only if the Db2 tables have been modified since the last load
- Partial reload, that is, if partitions of a Db2 table have been moved with the High Performance Storage Saver, reload of those partitions that have not been moved.
- Complete update of the data in an already loaded accelerator-shadow table with more recent Db2 data
- Update of the data in selected partitions of range-partitioned accelerator-shadow tables, provided that the partitioning key and the partition boundaries have not changed
- Deletion of data in accelerator-shadow tables that originate from range-partitioned tables after a ROTATE PARTITION operation
- Addition of new partition data to existing accelerator-shadow tables after a ROTATE PARTITION operation or ADD PARTITION operation on range-partitioned tables in Db2
- Addition of data from partitions that were appended to partitioned-by-growth tables in Db2
- Addition of data from partitions that were appended to partitioned-by-growth tables in Db2 InitialLoadPendingInitialLoadPending

Queries against already loaded accelerator-shadow tables go against the old data until the load operation is completed.

Generally, all table data is transferred to the accelerator during an initial load. Partial reloads are possible for range-partitioned tables and tables partitioned by growth. However, you must explicitly specify the partitions that you want to reload (update). Partial loads are only possible under the following conditions:

- If table partitions have been moved by the High Performance Storage Saver, you can load (initial load) or reload only those partitions that have not been moved. Moved partitions are always ignored.
- For range-partitioned tables and tables partitioned by growth, you can reload (update) the data of selected partitions only.

If the IBM Db2 Analytics Accelerator catalog in Db2 for z/OS does not indicate that a table has been defined on the specified accelerator or if accelerator-shadow tables are in a state that disallows a load operation, the SYSPROC.ACCEL_LOAD_TABLES stored procedure returns an error message and does not take any action. Valid states are:

- *InitialLoadPending*
- *Error*
- *Loaded*

If the stored procedure runs successfully, the state of all loaded accelerator-shadow tables is changed to *Loaded*.

During a load, IBM Db2 Analytics Accelerator automatically selects a suitable distribution key.

If the stored procedure fails, all table modifications are rolled back on the accelerator, and the accelerator-shadow tables return to their previous states (*InitialLoadPending*, *Error*, or *Loaded*) and the IBM Db2 Analytics Accelerator catalog remains unmodified.

A situation in which the tables are modified, but the timestamps of these tables do not change in the catalog can only occur if an error surfaces after the completion of operations on the accelerator, but before the end of the stored-procedure run. The diverging timestamps in the catalog and on the accelerator have no functional consequences; the accelerator-shadow tables will contain the most recent data and will be available for processing.

Parallel processing

Partitions of a partitioned table (partitioned by range or by growth) and non-partitioned tables can be loaded in parallel. By default, up to four entities of both types (table partitions, non-partitioned tables, or a mixture of both) are loaded in parallel. You can change the default by setting the AQT_MAX_UNLOAD_IN_PARALLEL environment variable to a different value.

The order in which partitions or non-partitioned tables are loaded can vary and is not predictable. The reason is that before each run, the accelerator calculates how to distribute the items most efficiently to the available parallel threads It is therefore very likely that the order will deviate from the order in which the entities are listed in the *table_load_specification*, which is the parameter used to specify the items to be processed.

If it is very important to you to know or keep the order, consider not to use parallel processing at all or to use an external loader. Such a loader can be called by using the `<externalTool>` element in connection with the `enforceTableLoadSequence` attribute in XML input for the *table_load_specification* parameter. See the following example:

```
<externalTool enforceTableLoadSequence="true" />
```

During execution, the stored procedure can increase or decrease the degree of parallelism at any time to optimize the performance of the current and of future load jobs. If that happens, indicating messages (`AQT10418I` and `AQT10419I`) are written to the SYSLOG. These messages are also returned by the stored procedure when processing has finished.

SYSPROC.ACCEL_LOAD_TABLES calls the Db2-supplied SYSPROC.DSNUTILU stored procedure, which unloads the data from the Db2 tables. The Workload Manager (WLM) policies might prohibit the invocation of SYSPROC.DSNUTILU, due to specific service class definitions or a current system overload.

In such a situation, SYSPROC.ACCEL_LOAD_TABLES allows you to try running SYSPROC.DSNUTILU again with a configurable timeout period between the retries. Before each retry, information about the DSNUTILU error `00E79002` is written to the WLM job log of the SYSPROC.ACCEL_LOAD_TABLES stored procedure. This gives operators suitable information for corrective actions (for example, running the **`-STA SYSPROC.DSNUTILU`** command or resuming the WLM application environment for DSNUTILU). You can configure the retries by setting the following environment variables, for example:

- `AQT_MAX_RETRIES_DSNUTILU=2`
- `AQT_SECONDS_BEFORE_RETRY_DSNUTILU=60`

These settings instruct SYSPROC.ACCEL_LOAD_TABLES to make three attempts at starting DSNUTILU and to wait for 60 seconds between the retries.

Additionally, you can set the AQT_SKIP_UNLOAD_EMPTY_PARTS environment variable to improve the performance of the load process. If this variable is set, SYSPROC.ACCEL_LOAD_TABLES does not call the SYSPROC.DSNUTILU utility for empty partitions. The detection of empty tables and partitions relies on Db2 real-time statistics.

Important: If the `<externalTool>` element is used as part of the *`table_load_specification`* parameter for SYSPROC.ACCEL_LOAD_TABLES, the Db2 UNLOAD utility is always called, irrespective of the setting of this variable.

An auto-cleanup mechanism is in place. If SYSPROC.ACCEL_LOAD_TABLES was ended prematurely or abnormally, for example by the **`-CAN THREAD`** Db2 command, Db2 UNLOAD utilities that have been started by the stored procedure remain in the *STOPPED* state and related resources are left in the `/tmp/aqtpipes/V7/<user>` directory, where `<user>` is the ID that was used to start the stored procedure. During the next run of the SYSPROC.ACCEL_LOAD_TABLES stored procedure with the same user ID (for any table), the Db2 UNLOAD processes are ended, and the leftovers are deleted.

Restrictions

- You cannot invoke the SYSPROC.ACCEL_LOAD_TABLES procedure more than once for the same accelerator-shadow table at the same time. While the procedure is running, the state of all specified accelerator-shadow tables changes to *LoadInProgress*, which prevents the start of another instance of SYSPROC.ACCEL_LOAD_TABLES that works on the same tables. The stored procedure can be invoked again after a successful completion of the previous run. Otherwise, an error message is issued.

- You can only use this stored procedure to load or update accelerator-only tables if an external tool is employed for this task. If such a tool is not available, you must use the `INSERT ... INTO TABLE ... IN ACCELERATOR` SQL statement. For information on how to call external tools, see the example at the end of the description of the *table_load_specification* parameter.
- If you change the partitioning type of the table space of a source table in Db2 for z/OS (for example, from *segmented* to *partitioned by growth*), you can no longer load the corresponding accelerator-shadow table. The operation will fail. Therefore, you must remove the table from the accelerator and redefine it after changing the partitioning type.
- Referencing accelerator-only tables, which have been created for federated access, cannot be loaded with an external tool because their content must, at any time, reflect the content of the underlying original table in the remote Db2 subsystem. If such a table were loaded by an external tool, it would be possible for the table content to diverge from the content of the underlying "master" table. Hence the access to referencing accelerator-only tables is blocked, and an attempt to load these with an external tool will fail.
- After a load, a replication-enabled accelerator-shadow table goes into *Suspended* state if someone has stopped replication during the load.

  Suspended means that the table is skipped when incremental updates are applied. To make this table part of the incremental update process again, you must reload it.

Multiple accelerators

If multiple accelerators are connected to the same Db2 subsystem and a table exists on more than one accelerator, you can load the various instances of this table in parallel, provided that the lock mode permits this (NONE and ROW do). Each accelerator contains a snapshot of Db2 data at the point of the last load operation. There is no mechanism that guarantees the consistency of the table data on all accelerators.

Syntax

```
CALL SYSPROC.ACCEL_LOAD_TABLES
(accelerator_name,
lock_mode,
table_load_specification,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

lock_mode

Controls the protection level while accelerator-shadow tables are being loaded. The protection level ensures a consistent state of the data with respect to the specified entity. Valid values:

TABLESET

Protects the Db2 source tables of to-be-loaded accelerator-shadow tables against changes during the load operation.

Data availability

Newly loaded data becomes available for queries when the load job has been finished for the entire table set.

Error behavior

If an error occurs while loading one of the tables in the set, the load job for that table is aborted, and the job proceeds with the next table in the set. Message AQT20121E is issued in case of a load error.

TABLE

Protects just the Db2 source table of the accelerator-shadow tables that is currently being loaded.

Data availability

Newly loaded data in a table becomes available for queries when the load job for that table has been finished and when the entire operations has moved on to the next table in the set.

Error behavior

If an error occurs while loading one of the tables in the set, the load job for that table is aborted, and the job proceeds with the next table in the set. Messages:

- Message AQT20121E is returned if the load of all tables in the set failed.
- Message AQT10420W is issued for each single table after a failure to load that table

PARTITIONS

Protects the original Db2 table-space partition of the accelerator-shadow table partition to be loaded against changes during the load. With this setting, a non-partitioned table is always locked completely.

Note: Locks at partition level are not supported for Db2 for Linux®, UNIX, and Windows.

Data availability

Newly loaded data in a table partition becomes available for queries when the load job for that partition has been finished and when the entire operations has moved on to the next partition or table.

Error behavior

If an error occurs while loading a table partition, the load job for that partition is aborted, and the job proceeds with the next partition to be loaded. An error has no effect on other partitions or tables to be loaded; the job continues. Messages:

- Message AQT20121E is returned if the load of all tables or all partitions of a partitioned table in the set failed.
- Message AQT10421W is issued for each partition after a failure to load that partition.

Special cases

To avoid partially loaded tables, lock mode TABLE is applied during an initial load, even if you specified lock mode PARTITIONS. For details, see the description of lock mode TABLE.

If the reload of a partition that is enabled for incremental updates fails, incremental updates are disabled for the entire table even if the reload of the other table partitions succeeded.

NONE

No locking at all. However, only committed data is loaded into the accelerator-shadow tables because the Db2 data is unloaded with isolation level CS and SKIP LOCKED DATA.

Data availability

Data availability is the same as for lock mode PARTITIONS.

Error behavior

The error behavior is the same as for lock mode PARTITIONS.

ROW

Protects at least the row or page that is being loaded against updates. Whether a row, a page, or an entire table is locked is determined by the LOCKSIZE defined for the table space of the source tables. (For more information, see the Db2 documentation for the CREATE TABLESPACE command. You find a link at the end of this topic. When lock mode ROW is set, Db2 data is unloaded at isolation level CS (cursor stability), but in contrast to lock mode NONE, rows locked by an application are not skipped. This is the recommended choice if incremental updates are enabled for the tables to be loaded.

The lock mode **Row** ensures that all incremental update changes not yet applied are written to a spill queue, which is emptied after the load. This means that changes that occurred before the setting of the capture point are applied at a later time.

If you want to use this lock mode in a different context, contact IBM support before doing so.

Data availability
Data availability is the same as for lock mode PARTITIONS.

Error behavior
The error behavior is the same as for lock mode PARTITIONS.

Important: If multiple instances of a table exist in a data sharing group, the lock mode ROW might lead to a situation in which some tables are not captured. This might happen, for example, if an update of a table in member A has not yet been mirrored in the same table in member B. If SYSPROC.ACCEL_LOAD_TABLES receives the table data from member B, the data in the corresponding accelerator-shadow table might deviate from the content of the original Db2 for z/OS table. This effect can sometimes be observed if all of the following conditions apply:

- The lock mode **Row** is used in conjunction with the setting detectChanges="DATA" (see "Automatic detection of modified data" on page 80 for more information).
- Incremental updates are being applied to the tables during the load process.
- Transactions that issue SQL statements on the tables, such as INSERT, DELETE, TRUNCATE, or UPDATE, have started before the load process and are running concurrently with this process.

To avoid an inconsistency between the tables, a temporary lock is set on the Db2 for z/OS table or partition to be unloaded. The user ID that runs SYSPROC.ACCEL_LOAD_TABLES must have the SELECT privilege on the affected tables. Otherwise, the lock cannot be set successfully, and the stored procedure returns SQL error -551. The frequency and the duration of such locks are controlled by the environment variables AQT_MAX_RETRIES_FOR_LOCKS and AQT_SECONDS_BEFORE_RETRY_LOCKS. These locks, however, might have the following side effects:

- The load performance decreases.
- You run into timeout situations.
- The execution of other applications is compromised.

You can suppress these locks by setting the following environment variable in the AQTENV data set as shown:

```
AQT_SKIP_LOCK_LOCKMODE_ROW = SET
```

Doing so restores the behavior of older product versions (no such locks).

table_load_specification
An XML input string that lists the accelerator-shadow tables to be loaded. The string must conform to the structure of the tableSetForLoad element in the SAQTSAMP(AQTSXSD1) data set. A table name must not occur more than once in the input string.

For *range-partitioned tables* and *tables partitioned by growth*, it is possible to update only the data in selected partitions. This is done by specifying a partition list in the XML input string. Other tables are always completely reloaded. An error is returned if a partition list is specified for a table that is neither range-partitioned, nor partitioned by growth.

You specify partitions by their logical or physical partition numbers, separated by commas (,). Ranges can be specified by using a colon (:) between the range boundaries. The range boundaries are inclusive. Using negative values, you can specify partitions by counting backwards from the upper end of the entire range of partition numbers. In this context, -1 is used to identify the last partition of a table. In a sorted sequence of partition numbers, the position of a partition in the sequence can be determined according to the following formula:

```
(number of last partition) + 1 - n
```

where n is the number of partitions to be deducted from the upper end.

Table 8. Examples of specifications for partitions and partition ranges

XML specification for partitions	Description
`<partitions>1</partitions>`	Includes only the first partition
`<partitions>1:10</partitions>`	Includes the first 10 partitions of a table
`<partitions>-10:-1</partitions>`	Includes the last 10 partitions of a table
`<partitions>1:-11</partitions>`	Includes all partitions except for the last 10
`<partitions>1,10:13,-1</partitions>`	Includes the partitions 1, partitions 10 through 13, and the last partition of a table

Example A:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:tableSetForLoad xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
  usePhysicalPartitionNumbers="true" version="1.0">
    <table name="SALES" schema="BCKE">
        <!-- explicitly specified partition numbers -->
        <partitions>1,5:10,20</partitions>
    </table>
    <table name="CUSTOMER" schema="BCKE" >
        <!-- negative partition numbers count from the end, -1 is the last
             element example for "all partitions except last 10" -->
        <partitions>1:-11</partitions>
    </table>
    <table name="ORDER2009" schema="BCKE">

    </table>
</dwa:tableSetForLoad>
```

The sample XML string includes the attribute setting `usePhysicalPartitionNumbers="true">`, which means that the specified partition numbers are regarded as *physical* partition numbers.

The sample XML string lists three accelerator-shadow tables to be loaded: SALES, CUSTOMER, and ORDER2009. For the SALES table, the partitions 1, 5 through 10, and 20 are loaded.

All partitions of the CUSTOMER table are loaded, except for the last 10 partitions. Assume that this table has 20 partitions. According to the formula, this gives you 20 + 1 - 11=10.

The behavior is different for ORDER2009: Because partitions are not specified, the stored procedure loads just the partitions that have changed in the original Db2 table (newly added partitions or changed partition boundaries).

Example B:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:tableSetForLoad
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.1">
     <table name="SALES" schema="BCKE" >
          <partitions>1,5:10,20</partitions>
     </table>
     <table name="CUSTOMER" schema="BCKE"
      detectChanges="PARTITIONS" />
     <table name="PRODUCT" schema="BCKE" />
</aqttables:tableSetForLoad>
```

The sample XML string lists three accelerator-shadow tables to be loaded or reloaded: SALES, CUSTOMER, and PRODUCT. Assume that these tables are partitioned by range.

The SALES table contains a `<partitions>` element, which lists the partitions to be loaded. These are the partitions with the IDs 1, 5, 6, 7, 8, 9, 10, and 20. If partitions were added as the result of an ADD or ROTATE operation, then these partitions will also be loaded.

For the CUSTOMER table, the `detectChanges` attribute is set to `PARTITION`, but a partition list has not been specified, so only tables that were added by an ADD or ROTATE operation since the last load will be loaded.

For the PRODUCT table, neither the `detectChanges` attribute, nor a partition list has been specified, so all partitions are loaded or reloaded.

The `<externalTool>` element is not intended for users or administrators, but for tool vendors who use the SYSPROC.ACCEL_LOAD_TABLES stored procedure, but have replaced the Db2 Unload Utility (DSNPROC.DSNUTILU) with another program. The element allows the tool vendor to specify the name of the program and the processing order of the tables.

In addition, the `<externalTool>` element can provide access to a callback interface (a sub-element with the name `<callback />`), which can be used to specify callback functions provided by the external application. When specified, these functions are called at various stages during the execution of the external program.

In the appendix of this book, you find sample header files for applications in the C or C++ programming language. Use these headers to define your own callback interfaces.

Example C:

```
<?xml version= "1.0" encoding= "UTF-8" ?>
<aqttables:tableSetForLoad
 xmlns:aqttables = "http://www.ibm.com/xmlns/prod/dwa/2011"
 version = "1.2" >
   <externalTool name = "IBM DB2 Analytics Accelerator Loader for z/OS, V1.1 (5639-OLA)"
    enforceTableLoadSequence = "true">
          <callback
                callerid = "0x4e353456"
                library = "HLOCALLB"
                versionFunction="VersionCallback"
                initFunction="InitCallback"
                loadFunction="LoadCallback"
                terminateFunction="TerminateCallback"
          />
   </externalTool>
   <table name = "UNPPARTITIONED3" schema = "TEST" />
   <table name = "PARTITIONED4" schema = "TEST" />
</aqttables:tableSetForLoad>
```

In this example, the `<externalTool>` element specifies IBM Db2 Analytics Accelerator Loader for z/OS as the external tool. The attribute setting `enforceTableLoadSequence="true"` ensures that the tables are loaded in the same order as shown.

The following list describes the `<callback />` specification. What you see are attribute and function names provided by the product IBM Db2 Analytics Accelerator Loader for z/OS. A different product certainly provides different attributes and functions, which means that the following set of instructions will have to be replaced accordingly.

callerid

A string that identifies the external tool or program. This string is passed to the callback functions for the following purposes:

- To check if the external program qualifies as a calling application.
- To take specific actions based on the received `callerid`.

library

The z/OS library member that contains the callback functions. This member must be included in the external library that was specified in the `//STEPLIB` statement during the Workload Manager (WLM) setup for the IBM Db2 Analytics Accelerator stored procedures.

For more information, see the appropriate WLM setup section in the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide*:

- *Setting up a WLM application environment for IBM Db2 Analytics Accelerator*
- *Setting up a WLM application environment for different product versions*

versionFunction

In this example, the `versionFunction` attribute is used to call a callback function with the name `VersionCallback`. According to the specifications of the external tool provider, this function is run once to check if the currently installed IBM Db2 Analytics Accelerator stored procedures are compatible with the version of the external program.

initFunction

The `initFunction` attribute is used to call the `initCallback` function. This function is also run once per program execution. The external program uses this function to capture the load information from the XML specification as provided by the stored procedure. In return, the stored procedure receives instructions regarding program execution, such as the number of parallel threads to be started.

loadFunction

The `loadFunction` attribute specifies the main program to be run, that is, the program that will replace the Db2 Unload Utility to extract the load data from the Db2 for z/OS source tables.

terminateFunction

The `terminateFunction` attribute specifies a function that ends the main program (`loadFunction`) if errors occur during the execution. The termination callback function issues an error message with return code 8 (`rc=8`) in such cases.

Another sub-element of the `<externalTool>` element is the `<rerouteDataCallback />` element. Its main purpose is to prevent that the same data is unloaded multiple times in cases where copies of accelerator-shadow tables exist on different accelerators. With the `<rerouteDataCallback />` element, you can address a callback interface that will unload the data to a re-readable pipe, from where it can be picked up by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. This way, the unload process can be skipped during subsequent runs of the stored procedure. The following example shows callback instructions that are valid if the IBM Db2 Analytics Accelerator Loader for z/OS is used as the external program. For other external programs, the names of attributes and callback functions will have to be adjusted.

Example D:

```
<?xml version= "1.0" encoding= "UTF-8" ?>
<aqttables:tableSetForLoad
 xmlns:aqttables = "http://www.ibm.com/xmlns/prod/dwa/2011"
 version = "1.2" >
   <externalTool name = "IBM DB2 Analytics Accelerator Loader for z/OS, V1.1 (5639-OLA)"
    enforceTableLoadSequence = "true">
           <rerouteDataCallback
                 callerid = "0x4e353456"
                 library = "HLOCALLB"
                 versionFunctionRDC = "VersionCallbackRDC"
                 initFunctionRDC = "InitCallbackRDC"
                 preUnloadFunctionRDC = "preUnloadCallbackRDC"
                 postUnloadFunctionRDC = "postUnloadCallbackRDC"
           />
   </externalTool>
   <table name = "UNPPARTITIONED3" schema = "TEST" />
   <table name = "PARTITIONED4" schema = "TEST" />
</aqttables:tableSetForLoad>
```

The callback specification in detail:

callerid

See the description of this attribute for the `<callback />` element.

library

See the description of this attribute for the `<callback />` element.

versionFunctionRDC

See the description of the `versionFunction` attribute for the `<callback />` element. It much works in the same way.

initFunctionRDC

See the description of the `initFunction` attribute for the `<callback />` element. It much works in the same way.

preUnloadFunctionRDC

The callback function specified by this attribute receives the name of the pipe that the Db2 Unload Utility writes the load data to (Db2 Unload pipe). This function also receives the name of the pipe that IBM Db2 Analytics Accelerator reads the load data from (Db2 load pipe). Both pipes are managed and owned by SYSPROC.ACCEL_LOAD_TABLES.

If this callback function returns an error code greater than 4, the unload utility and the callback function specified by the `postUnloadFunctionRDC` attribute are not started.

postUnloadFunctionRDC

The callback function specified by this attribute is run when the Db2 Unload Utility or its replacement finishes processing. It returns a success message.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Automatic detection of modified data

The SYSPROC.ACCEL_LOAD_TABLES stored procedure can detect modified data automatically and then reload only the tables or partitions that have changed since the last load. This is useful if you want to run an automated batch process repeatedly that refreshes the data in your accelerator-shadow tables at regular intervals, for example daily, and you do not know which data has changed in Db2. Through a special attribute in the XML input, you can synchronize your accelerator data and at the same time ensure that just a minimum amount of data is transferred during the process.

You can set the `detectChanges` to the following values:

detectChanges="DATA"

Basically, this setting causes a reload of partitions or entire tables if changes were detected in the original partitions or tables since the last load.

detectChanges="PARTITIONS"

Basically, this setting causes a load or reload of partitions on the accelerator that were added or rotated in Db2 for z/OS since the last load.

However, the actual effect of these settings also depends on the table type and whether a partition list has been specified, so the outcome is not always straightforward. See the following table:

Table 9. Effect of the `detectChanges` attribute in connection with other settings

Attribute setting	Partition list specified	Table type	Effect
Not set or empty value	No	Partitioned by range	Full table reload
		Partitioned by growth	Full table reload
		Non-partitioned	Full table reload
	Yes	Partitioned by range	Reload of selected partitions plus added or rotated partitions
		Partitioned by growth	Reload of selected partitions plus added or rotated partitions
		Non-partitioned	Error AQT10403E (invalid attribute combination)

Table 9. Effect of the `detectChanges` attribute in connection with other settings (continued)

Attribute setting	Partition list specified	Table type	Effect
`detectChanges= "DATA"`	No	Partitioned by range	Reload of changed partitions including added or rotated partitions
		Partitioned by growth	Reload of changed partitions including added or rotated partitions
		Non-partitioned	Full table reload if changes were detected
	Yes	Partitioned by range	Reload of changed partitions including added or rotated partitions. Warning `AQT10148W` (partition list ignored)
		Partitioned by growth	Reload of changed partitions including added or rotated partitions. Warning `AQT10148W` (partition list ignored)
		Non-partitioned	Error `AQT10403E` (invalid attribute combination)
`detectChanges= "PARTITION"`	No	Partitioned by range	(Re)load of just the added or rotated partitions
		Partitioned by growth	(Re)load of just the added or rotated partitions
		Non-partitioned	Error `AQT10403E` (invalid attribute combination)
	Yes	Partitioned by range	Reload of selected partitions plus added or rotated partitions
		Partitioned by growth	Reload of selected partitions plus added or rotated partitions
		Non-partitioned	Error `AQT10403E` (invalid attribute combination)

Important:

- Accelerator-shadow tables in *InitialLoadPending* or *Error* state are always fully reloaded.
- You cannot specify a partition list for non-partitioned tables.
- Partitions that have been removed from the beginning of a range-partitioned table due to a ROTATE operation are automatically detected and deleted from the corresponding accelerator-shadow table.
- All partition changes in range-partitioned tables except for ADD and ROTATE lead to an error if you try to reload the corresponding accelerator-shadow tables.
- The `forceFullReload` attribute is deprecated because `detectChanges="PARTITIONS"`.without partition list has the same effect as `forceFullReload="false"`, and omitting the `detectChanges` attribute and a partition list has the same effect as `forceFullReload="true"`.
- If you use lock mode `NONE` and `detectChanges="DATA"`, it can happen that more changes are detected than actually transferred to the accelerator. This is caused by the SKIP LOCKED DATA option, which is used implicitly by the Db2 Unload Utility. If parts of the data were locked by other processes,

then this data is ignored and not unloaded to the accelerator. To ensure that all changes are finally transferred, a partition in an accelerator-shadow table is always fully reloaded if it was previously loaded with lock mode `NONE` and `detectChanges="DATA"`. A reload is started even if the partition had no further changes since then.

Example

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:tableSetForLoad
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.1">
   <table name="UNPARTITIONED3" schema="TEST" detectChanges="DATA"/>
   <table name="PARTITIONED4" schema="TEST" detectChanges="PARTITIONS"/>
</aqttables:tableSetForLoad>
```

Suppose that the table `UNPARTITIONED3` is a non-partitioned table, and that `PARTITIONED4` is partitioned by range. The accelerator-shadow table of the non-partitioned table will be fully reloaded if changes were detected since the last load. The accelerator-shadow table of the partitioned table will be reloaded only if partitions were added or rotated since the last load. The table will not be fully reloaded. Only the new partitions will be added.

If very short synchronization intervals are required, such as periods of minutes for example, consider using the incremental update function of IBM Db2 Analytics Accelerator.

SYSPROC.ACCEL_LOAD_TABLES in connection with the `detectChanges="DATA"` setting invokes the same change detection mechanism as the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure. That is, SYSPROC.ACCEL_LOAD_TABLES reloads all partitions or non-partitioned tables for which SYSPROC.ACCEL_GET_TABLES_DETAILS returns a change category other than NONE. The following restrictions apply:

- Under specific conditions, for example if table data was modified by a non-IBM utility, table changes might not be correctly detected. In such a case, you must manually reload the affected partitions or the entire accelerator-shadow table.
- If the detection of data changes is not possible, that is, in cases where the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure returns the change category `UNKNOWN`, tables or partitions are reloaded even if the accelerator data is still in sync with the original Db2 data.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.
- The accelerator-shadow table must exist on the accelerator that the *accelerator_name* parameter specifies.
- Because SYSPROC.ACCEL_LOAD_TABLES starts one or more instances of SYSPROC.DSNUTILU, DSNUTILU must be installed and configured properly on z/OS. This includes a correct setup of the WLM application environment for Db2-supplied stored procedures and IBM Db2 Analytics Accelerator stored procedures. For more information, see *Defining WLM performance goals for IBM Db2 Analytics Accelerator for z/OS stored procedures* in the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide*.
- To avoid naming conflicts on z/OS, do not run Db2 utilities whose names start with *AQT* at the same time as this stored procedure.

Authorizations for z/OS

The required authorizations for this stored procedure are listed in the following section.

Important: The ID used to run this stored procedure must be exactly the same as the ID used to run the Db2 for z/OS UNLOAD utility DSNUTILU.

Therefore, do not let applications change this ID by use of a Db2 authorization exit, such as DSN3@ATH or DSN3@SGN.

The list of required authorizations:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Authorization to read the original Db2 base tables of the accelerator-shadow tables that are listed in the *table_load_specification* parameter
- DISPLAY authorization (for calling -DIS GROUP)
- Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics)
- Authorization to execute the RUNSTATS utility on the databases that contain the tables to be loaded, such as the STATS authorization.
- Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved.
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure (UNIX System Services pipes are created in this directory)
- Privilege to connect to Db2 for z/OS via RRSAF.

  Important: If you want to grant the required Db2 privileges to secondary authorization IDs, for example groups, make sure that the Db2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.

- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_LOAD_TABLES stored procedure locks Db2 objects as defined by the *lock_mode* parameter. After a successful load of all accelerator-shadow tables, it changes the value of the REFRESH_TIME column for these tables in the SYSACCEL.SYSACCELERATEDTABLES catalog table of Db2 for z/OS.

Messages that indicate a change in the degree of parallelism (AQT10418I and AQT10419I) are written to the SYSLOG. You can suppress the logging of these messages by *commenting out* the following line in the AQTENV data set:

```
AQT_DO_SYSLOGGING = SET
```

On the accelerator, the stored procedure updates the IBM Db2 Analytics Accelerator catalog and, in case of an initial load, creates corresponding accelerator-shadow tables.

The dynamic statement cache in Db2 is invalidated for initial loads. All plans that are affected by the cache invalidation must therefore be rebuilt by Db2. This might increase the preparation time of the affected queries slightly because a new plan must be prepared during the first execution of a query. This, however, is a onetime operation. For all subsequent preparations of the same query, the dynamic statement cache can be used again. The initial prolongation of the preparation time is in most cases more than outweighed by the reduction that sets in when successfully accelerated queries are submitted repeatedly because such queries are immediately routed to the accelerator.

Related reference

The message parameter
The message parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Environment variables
The job control language (JCL) for the configuration of the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures contains a data definition (DD) "AQTENV". This data definition includes a data set in which environment variables are defined. These variables control the behavior of several stored procedures.

Sample C/C++ header for external loader callback interface
Use the following C/C++ header to define an interface for callback functions to be run by an external load application called by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. Make sure to compile your interface as a 31-bit XPLINK application.

Sample C/C++ header for rerouting callback interface
Use the following C/C++ header to define an interface for callback functions to be run by an external rerouting application called by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. Make sure to compile your interface as a 31-bit XPLINK application.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10201E
Information about the *acceleratorName* accelerator in the IBM Db2 Analytics Accelerator for z/OS catalog tables is inconsistent.

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

CREATE TABLESPACE

SYSPROC.ACCEL_REMOVE_ACCELERATOR

Removes an accelerator from the system configuration.

Details

The removal includes Db2 catalog entries for the accelerator and all accelerator data that is related to the corresponding Db2 subsystem. The accelerator is also disabled for this Db2 subsystem by means of a `-STOP ACCEL` command.

Syntax

```
CALL SYSPROC.ACCEL_REMOVE_ACCELERATOR
(accelerator_name,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

- Tables must have been deleted from the accelerator before SYSPROC.ACCEL_REMOVE_ACCELERATOR is run. The SYSACCEL.SYSACCELERATEDTABLES catalog table must be void of entries for the specified accelerator, that is, the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure must return an empty list.
- It must be possible to contact the specified accelerator. Otherwise, it is not removed.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

This stored procedure removes entries (rows) pertaining to the specified accelerator from the IBM Db2 Analytics Accelerator catalog table SYSACCEL.SYSACCELERATORS and from the following tables in the Db2 communications database:

- SYSIBM.LOCATIONS
- SYSIBM.IPNAMES
- SYSIBM.USERNAMES

Entries pertaining to the specified accelerator are removed from the SYSACCEL.SYSACCELERATORS table of the catalog only if all tables have been removed from the accelerator before, and no more tables exist.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES

Removes accelerator-only tables that were created by the SYSPROC.ACCEL_CREATE_REFERENCE_TABLES stored procedure for access by a referencing Db2 subsystem.

Details

The stored procedure initiates the removal from the referencing Db2 subsystem. For that reason, the referencing Db2 subsystem must be paired with the same accelerator as the Db2 subsystem that contains the original tables. You can only remove the reference tables from the accelerator if access to these tables has been granted previously by running the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure, and if this access privilege still exists.

Syntax

```
CALL SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES
(accelerator_name,
remote_location_name,
table_reference_set
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

remote_location_name
Location name of the remote Db2 subsystem owning the original tables. Location name means: the name of the subsystem as listed in the SYSIBM.LOCATIONS table. You can display the location name by running the `-DISPLAY DDF` Db2 command. For an example, see remote_location_name.

The remote Db2 subsystem must be paired with the accelerator specified by the *accelerator_name* parameter. The referencing Db2 subsystem might use a different name for that accelerator.

table_set
An XML string that lists the names of the accelerator-only tables that you want to remove. The tables must have been created by running the SYSPROC.ACCEL_CREATE_REFERENCE_TABLES stored procedure. Names must not occur more than once in the list. The list or string must not be empty.

Example:

```
<dwa:tableSet xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <table name="ORDER" schema="TPCD2PCT"/>
    <table name="PARTSUPP" schema="TPCD2PCT"/>
    <table name="SUPPLIER" schema="TPCD2PCT"/>
    <table name="NATION" schema="TPCD2PCT"/>
</dwa:tableSet>
```

This example lists the accelerator-only tables ORDER, PARTSUPP, SUPPLIER, and NATION for removal.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)
Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

The accelerator that is specified by the `accelerator_name` parameter must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR stored procedure.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- Ownership of the accelerator-only tables in the table set. Alternatively, you can use DBADM, SYSADM, SYSCTRL, or system DBADM authority.
- If tables reside in implicitly defined databases, authority to remove tables from these databases.
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- SELECT and DELETE on SYSACCEL.SYSACCELERATEDTABLES
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

During execution, the stored procedure carries out the following tasks:

- It connects to the accelerator specified by the *`accelerator_name`* parameter and, to this end, uses the information in the Db2 for z/OS communications database.
- It verifies for each table in the set that it exists on the accelerator specified by the `accelerator_name` parameter and that it is a reference table pointing back to an accelerator-shadow table or accelerator-only table in the owning (remote) Db2 subsystem.
- It removes the entries for the reference tables in the table set from the SYSYCCEL.SYSACCELERATEDTABLES table (for the specified accelerator).
- It removes the reference tables themselves.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE
Revokes the access to a set of tables that was granted by running the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure.

SYSPROC.ACCEL_CREATE_REFERENCE_TABLES
Creates table metadata in the catalog of a referencing Db2 subsystem and pointers (references) to accelerator-shadow tables or accelerator-only tables, so that tables defined in a remote Db2 subsystem can be queried.

SYSPROC.ACCEL_REMOVE_TABLES

Removes (drops) accelerator-shadow tables and accelerator-archive tables, but not accelerator-only tables and deletes the corresponding entries from the IBM Db2 Analytics Accelerator catalog in Db2 for z/OS.

Details

The procedure first removes tables from the accelerator, then the corresponding entries from the SYSACCEL.SYSACCELERATEDTABLES catalog table. In case of an error, such as an only partial removal of a table, you can rerun the procedure.

If you remove a table from Db2 for z/OS, you also have to remove the corresponding accelerator-shadow table from all of your accelerators. That is, you have remove each copy of the accelerator-shadow table in a separate, manual step, either by using the removal function in your administration client, or by running this stored procedure.

The only exception are tables that are replicated by the IBM Integrated Synchronization function. In this case, the accelerator-shadow tables are removed automatically when you remove the original Db2 tables.

Attention:

- In general, the SYSPROC.ACCEL_REMOVE_TABLES stored procedure removes tables regardless of their state. This means that running operations (table load, for example) will be interrupted and ended prematurely. Running queries will be canceled.
- If you want to remove a table whose original data has partly or entirely been archived on an accelerator by the High Performance Storage Saver, the operation is not carried out and a warning is returned. This happens because the data on the accelerator would become inaccessible after removing the table. It is possible to override this default behavior by specifying the `force="true"` option in the input XML string. If you do that, you must recover the partition data manually from the Db2 for z/OS image copies. Note that the table spaces of the affected partitions might have been left in a restricted state like PRO (partition read-only). If so, you must remove the restricted states manually in Db2 for z/OS before you can insert new data into the partitions.
- If you remove a table whose data has been archived by the High Performance Storage Saver, you also lose information about the image copies. That is, you lose information telling you in which image copies the table data resides. It might prove hard to find the proper image copy so that you can restore the data if needed. So before you remove such a table, run SYSPROC.ACCEL_GET_TABLE_DETAILS to obtain the information about the image copy. Store this information in a safe place.

- As indicated at the beginning, this stored procedure does not remove accelerator-only tables. The reason is that after removing it from the accelerator, a table without data is left in Db2. Such a table cannot be used anymore. Therefore, you must run the Db2 DROP command to remove an accelerator-only table.

  There is one exception though: If the accelerator does not receive information about the committal of a DROP TABLE statement, an orphaned accelerator-only table remains. You can use SYSPROC.ACCEL_REMOVE_TABLES to remove such orphaned tables. To identify such tables, you can run the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure. The output of this procedure lists such tables with a type of (`acceleratorOnly`) and an integrity status of (`ViolatedInDatabaseManagementSystem`).

  If, on the contrary, an accelerator-only table is orphaned in Db2 for z/OS, you must submit a DROP TABLE statement to remove the leftovers. Such a situation can occur if a table creation has been committed, but the information flow (DRDA connection) between Db2 and the accelerator has been disrupted. In this case, the table creation on the accelerator is rolled back, leaving an orphaned table in Db2. Again, you can identify such tables by running the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure. In the output, such tables are listed with a type of (`acceleratorOnly`) and an integrity status of (`ViolatedOnAccelerator`).

- The stored procedure always attempts to delete a table from the specified accelerator, even if it is not listed in the catalog or does not exist in Db2 for z/OS because it is the only way to remove (the rests of) tables for which the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure returns an integrity status of *ViolatedInDatabaseManagementSystem*.
- The stored procedure continues with a warning if a table is not found on the accelerator or in the SYSACCEL.SYSACCELERATEDTABLES table.
- In the (unlikely) case of an error, the stored procedure ends. Tables that have already been removed at this point cannot be restored (no rollback). Tables that were not yet processed at the time of the error are not removed. Rerun the stored procedure to remove these tables. As you do this, warnings will be issued for the tables that have already been removed before the error occurred.

Impact on federated access

To be on the safe side, remove any federated access to tables before removing the tables themselves. That is, run the following stored procedures in the order indicated or the corresponding functions in your administration client:

1. SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES
2. SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE

The implications are as follows if you remove a table, but forget to remove the federated access before:

- Related entries are removed from the SYSIBM.SYSACCELERATEDAUTH table and existing grants are revoked.
- You end up with a referencing accelerator-only table whose reference object (the original table) is gone. This leaves an orphaned entry in the catalog of the referencing Db2 subsystem. `SQLCODE -204` (object not found) is returned when you run a query against this table. If you run the SYSPROC.ACCEL_GET_TABLES_INFO stored procedures to obtain information about tables, it will report a `Violated on accelerator` error for accelerator-only tables with dangling references.

Syntax

```
CALL SYSPROC.ACCEL_REMOVE_TABLES
(accelerator_name,
table_set,
message
);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_set

A list of the tables that you want to process. The XML string that you specify as input for this parameter must conform to the structure of the `<aqttables:tableSet … >` or `<aqttables:tableSetForRemove … >` element in the SAQTSAMP(AQTSXSD1) data set. The latter allows you to specify the `force="true"` option.

The tables must have been defined by the SYSPROC.ACCEL_ADD_TABLES stored procedure. The same table name must not appear more than once in the table set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <aqt:tableSetForRemove xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
      <table name="SALES" schema="SCHM" />
      <table name="PRODUCT" schema="SCHM" />
   </aqt:tableSetForRemove>
```

The sample XML code instructs the stored procedure to work on the SALES and PRODUCT tables.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Messages

The following messages are likely to occur in connection with the SYSPROC.ACCEL_REMOVE_TABLES stored procedure:

AQT10076W

Is returned, along with a success message, if a table could only be found on the accelerator or in the Db2 for z/OS catalog table SYSIBM.SYSACCELERATEDTABLES.

The orphaned table or catalog entry is not removed, but processing for the other tables continues.

AQT10077W

If a table could neither be found on the accelerator, nor in the Db2 for z/OS catalog table SYSIBM.SYSACCELERATEDTABLES.

Processing for the other tables continues after such a warning.

AQT10411E

Is returned if you try to remove an accelerator-only table by using SYSPROC.ACCEL_REMOVE_TABLES.

AQT10505W

Is returned, along with a success message, if you try to remove a table archived with the High Performance Storage Saver.

The archived table is not removed, but processing for the other tables continues.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- DELETE on SYSACCEL.SYSACCELERATEDTABLESAUTH
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.

Accessed data and systems

The SYSPROC.ACCEL_REMOVE_TABLES stored procedure deletes rows from the SYSACCEL.SYSACCELERATEDTABLES catalog table in Db2 for z/OS.

On the accelerator, it updates the IBM Db2 Analytics Accelerator catalog and removes tables from the accelerator database. If the last table is removed from the accelerator, the accelerator database is also removed (dropped).

Related reference

The message parameter

The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member

Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E

The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

AQT10076W

The *tableSchema.tableName* table on accelerator *acceleratorName* cannot be found in the catalog of the *unknownOnSide* or has been altered, such that the table definitions on the accelerator are not longer in synch with the definitions in the database.

AQT10077W

The *tableSchema.tableName* table is unknown on the *acceleratorName* accelerator.

SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES

Restores the data of archived partitions to their original locations. The stored procedure works on partitions that were archived with the High Performance Storage Saver (SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure).

Details

During the process, the archived data is restored to Db2. Rather than deleting the data from the accelerator, it is made available for regular accelerated queries, that is, queries that use the setting `GET_ACCEL_ARCHIVE=NO` as a special register or ZPARM, or do not use this special register or ZPARM at all. The following scenarios are supported:

1. Restoring the data of one or more explicitly specified partitions
2. Restoring the data of an entire table

To restore the data to the original Db2 table, the stored procedure uses the image copies that were created at the time of the archiving operation (the execution of the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure).

Restore without active accelerator

The ACCEL_RESTORE_ARCHIVE_TABLES procedure can only restore data from a table that has been defined on an accelerator. The accelerator must be active (status *online*). If the accelerator is unavailable or if an accelerator-archive table was accidentally removed without restoring the archived data before, you can only recover the Db2 data manually from the Db2 image copies.

When partition data is archived with IBM Db2 Analytics Accelerator for z/OS Version 4, the PRO state is set on the partition, which prevents that further image copies are created for the partition. If that state has been properly maintained, it is guaranteed that a **`RECOVER TO LASTCOPY`** operation will restore the partitions from the image copies that were created when the data was archived. IBM Db2 Analytics Accelerator for z/OS provides a sample JCL job that demonstrates the procedure. You can find this sample job in the SAQTSAM(AQTSJI04) data set.

Parallelism and error behavior

Parallel processing is applied on the partition level, that is, multiple partitions are handled concurrently. Per default, data from one up to four partitions (same default as for SYSPROC.ACCEL_LOAD_TABLES) can be restored concurrently. You can change this default by setting the AQT_MAX_UNLOAD_IN_PARALLEL environment variable to a different value.

If an operation fails on a single partition, only the failing partition is affected. All changes to previously processed partitions are kept and are not rolled back. For those partitions that were in the process of being restored while the error occurred, you must call the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES procedure again (preferably after fixing detected problems) to resume processing. You can use the **Restore partitions to Db2** function in your administration client to display the restore (error) status of each partition. Alternatively, you can invoke the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure to detect tables with incompletely restored partitions and the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure to obtain information about partitions that must be processed again because the restore operation was interrupted.

If an error occurs, indexes are not rebuilt because this is potentially costly and time-consuming and would have to be repeated when the procedure is called again to resume processing for the failed partitions. If indexes could not be rebuilt, they are set to the *rebuild-pending* state at the end of the restore operation. This might degrade the performance of queries exploiting these indexes and cause static SQL queries to fail due to an unavailable resource.

For all utility calls via SYSPROC.DSNUTILU, return code `-471` with reason code `00E79002` might be returned. In many cases, this indicates that the dispatching priority assigned by the Workload Manager (WLM) to the procedure was too low, so that the WLM could not assign the request to a TCB before the expiration of the time limit that was defined at installation time. As with the SYSPROC.ACCEL_LOAD_TABLES stored procedure, the call to DSNUTILU is repeated. The number of retries is controlled by the AQT_MAX_RETRIES_DSNUTILU environment variable, whose default value is 2, and the AQT_SECONDS_BEFORE_RETRY_DSNUTILU environment variable, whose default value is 60. The same behavior occurs when the RECOVER utility cannot obtain a drain lock in time.

Restrictions

- Do not run utilities on a table space while you are restoring data to tables in the table space.
- Since the RECOVER and the REBUILD INDEX utilities are used to restore data, queries should not be run at the same time because these are likely to end in a *resource unavailable* error (`SQLCODE -904`).

  Even if queries do not fail, there is no guarantee that the results include any of the restored data. Queries that were routed to the accelerator might or might not include restored data if the Db2 special register `CURRENT GET_ACCEL_ARCHIVE` is set to `NO` (it does not matter how this value was set, that is, implicitly or explicitly). On the contrary, the restored data will be available for queries that are routed to the accelerator with `SET CURRENT GET_ACCEL_ARCHIVE = YES`.
- While the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure is processing a table, do not start the procedure again for the same table. Also, do not start SYSPROC.ACCEL_ARCHIVE_TABLES for this table as long as SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES is running.

Db2 Utilities

During a recovery of archived partitions, the following Db2 utilities are run in the order indicated:

RECOVER

The RECOVER utility restores the original Db2 data from the image copies that were created at the time the partitions were archived. The utility works at partition level. Several threads can be run in parallel if multiple partitions are specified.

The following happens in case of an error:

1. All concurrently running threads restoring other partitions than the one that has caused the failure run to completion. After that, the process is stopped and any remaining partitions or tables are not recovered.
2. The partition that caused the error is set to the PRO state again to protect it against additional copies and changes.
3. Unless the previous run was already the last retry according to the setting of the AQT_MAX_RETRIES_DSNUTILU environment variable in the AQTENV data set, the action is retried if the error was caused by the Workload Manager (WLM) environment (`-471 0E790002`) or by a drain lock timeout situation.
4. The RESTORE INDEX utility will not be executed for the table.
5. The CHECK DATA utility will not be executed for the table.

If the RECOVER utility fails, solve the reported issue and run the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure again to complete the restore operation (including the indexes).

REBUILD INDEX

Rebuilds all indexes of the tables that were recovered (and are in *rebuild-pending* state). The use of these indexes, for example in queries, is not possible before the indexes have been completely rebuilt. Queries can thus not be processed as fast as usual as long the indexes are unavailable. The utility is run once per table, and rebuilds all data-partitioned indexes as well as non-partitioned indexes.

If the utility fails, the indexes are left in *rebuild-pending* state, and an error message is issued to explain the problem. You must take the following actions after that:

1. Correct the problem that has caused the utility to fail.

2. Re-invoke SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES to resume index rebuilding. The RECOVER step is skipped in this case.

CHECK DATA

The CHECK DATA utility is invoked automatically for partitions that are in *check-pending* state after a restore operation.

The utility checks the consistency of the restored data. This might be necessary if data has been restored to a child table in a foreign-key relationship. The environment variable AQT_SORTDEVT allows you to explicitly specify a device name, such as SYSDA, which is used in the REBUILD INDEX and CHECK DATA utility statements for sort data sets.

If the CHECK DATA utility fails or detects an inconsistency, the following chain of actions is triggered:

1. The table is left in *check-pending* state. The same happens if the utility itself fails. The utility output is included in an error message only, and will not be written to an extra data set, which would contain inconsistent records.
2. The process is stopped and any remaining partitions or tables are not recovered.
3. Indexes are not rebuilt because this is potentially costly and time-consuming and would have to be repeated when the procedure is called again to resume processing for the failed partitions. If indexes could not be rebuilt, they are set to the *rebuild-pending* state at the end of the restore operation. This might degrade the performance of queries exploiting these indexes and cause static SQL queries to fail due to an unavailable resource.

If the CHECK DATA utility fails, solve the reported issue and run the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure again.

Syntax

```
CALL SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES
(accelerator_name,
table_restore_specification,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

table_restore_specification

An XML string containing a list of the tables to be restored. This XML input string must conform to the structure of the `tableSetForRestoreArchiving` element in the SAQTSAMP(AQTSXSD1) data set.

If you specify tables for which archived partitions do not exist, the operation has no effect. An error message is not issued in such a case.

If you do not specify partitions explicitly, all archived partitions of the specified tables are restored. In addition, you can only restore the data of partitions that still exist in Db2 for z/OS.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:tableSetForRestoreArchiving
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
 usePhysicalPartitionNumbers="true"> version="1.0">
    <!-- restore selected partitions for table SALES -->
    <table name="SALES" schema="BCKE">
       <partitions>1,5:10,20</partitions>
    </table>
    <!-- restore all partitions of table CUSTOMER -->
    <table name="CUSTOMER" schema="BCKE" />
</dwa:tableSetForRestoreArchiving>
```

The attribute `usePhysicalPartitionNumbers` allows you to specify whether partition numbers are to be treated as logical or physical partition numbers. In this example, the attribute has been set to `true`, which means that the specified partition numbers are regarded as *physical* partition numbers.

`message`
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Prerequisites

Recommended setting for large data partitions

Before you restore large data partitions with the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure, comment out and customize the AQT_UTILITY_TMP environment variable in the AQTENV data-set member, so that its value suits your z/OS storage environment. The variable sets a name template which includes the high level qualifier (HLQ) of the work data sets (WORKDDN) that are used by the Db2 utilities that are invoked by the stored procedure. If a Db2 utility work storage pool exists in your z/OS environment, specify the HLQ of this pool.

Otherwise, associate a sufficiently large temporary pool with the HLQ by using an appropriate storage-management-subsystem (SMS) data-class, in a similar way as it is shown for the archive copy data sets in the "Prerequisites" on page 14 section of the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure.

If the AQT_UTILITY_TMP variable is not set, which is the installation default, the HLQ is always `<USERID>.AQT`, where `<USERID>` is the user ID of the caller. This setting might be unsuitable for the allocation of large WORKDDN data sets in your particular environment.

Alternatively, you can specify the variable AQT_SORTDEVT. This variable allows you to explicitly specify a sort device name (SORTDEVT), as it is used in some environments to specify a sort pool. See also AQT_UTILITY_TMP and AQT_SORTDEVT in Appendix A, "Environment variables," on page 201.

Space requirements for large data partitions

- The REBUILD INDEX utility is invoked to rebuild all affected indexes of a table at once. The utility allocates all required sort work data sets dynamically, thereby exploiting the highest possible degree of parallelism on your system. Therefore, a space in bytes up to the following maximum is required for sort work data sets:

  Non-partitioned indexes × *number of table rows* × 2 ×
  (*longest non-partitioned index key* + 10) + *data-partitioned indexes* ×
  number of restored rows × 2 × (*longest data-partitioned index key* + 10) +
  ST01WK*nn*(*data-partitioned sequential indexes*)

  where *longest non-partitioned index key* and *longest data-partitioned index keys* stand for the maximum lengths of these keys, and ST01WK*nn*(*data-partitioned sequential indexes*) is a value obtained by applying the formula in the section *Calculating the size of the ST01WKnn data sets*, in in the *Db2 11 for z/OS 11.0.0 Utility Guide and Reference: Data sets that REBUILD INDEX uses* (see link under **Related information**).

  Note: The formulas in this and the following section are valid for Db2 11 for z/OS and Db2 10 for z/OS. It is unpredictable at this point whether these formulas can be applied to future versions of Db2 for z/OS.

- The CHECK DATA utility is invoked to check all table constraints and foreign-key constraints in all restored partitions. The utility allocates two WORKDDN data sets dynamically that have the following total size in bytes:

  2 × *restored rows* × *number of constraints* × (*longest foreign key* + 18)

  where *longest foreign key* is the maximum length of the longest foreign key of the restored table, as defined in *Defining work data sets* in the *Db2 11 for z/OS 11.0.0 Utility Guide and Reference: Data sets that CHECK DATA uses* (see link under **Related information**).

Tip:

- If the table to be restored has a high number of indexes or a high number of constraints in combination with very large partitions or a high number of partitions, one of the space requirements might exceed your available temporary storage space. In this case, do not specify all partitions of the table in a single stored-procedure call, but reduce the number of partitions, if necessary to just one partition per call.
- If the index sort size becomes too large on your system, drop one or more indexes of the table and rebuild these indexes manually after the restore operation.
- If the WORKDDN size becomes too large, drop one or more constraints and re-create these constraints manually after the restore operation.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Authorizations for z/OS

The user ID under which this stored procedure is invoked must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics).
- Authorization to run the -DISPLAY DATABASE command for all databases that contain table partitions to be restored
- Authorization to run the **`-START DATABASE`** command on all table spaces that the tables belong to (for the utility lock that is used during the restore operation)
- Authorization to run the **`-STOP DATABASE`** command on all table spaces that the tables belong to
- Authorization to run the REPAIR utility on all tables (to reset the `DB2 RO_ACCEL` state)
- Authorization to run the RECOVER utility on all tables
- Authorization to run the CHECK DATA utility on all tables
- Authorization to run the REBUILD INDEX utility on all indexes of the restored tables

- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure (temporary files are created there)
- Privilege to connect to Db2 for z/OS via RRSAF.

  Important: If you want to grant the required DB2 privileges to secondary authorization IDs, for example groups, make sure that the DB2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests.
- Privilege to create data sets with the qualifier `<USERID>.AQT`, where `<USERID>` is the user ID of the person invoking the stored procedure
- Read access in RACF to COPY data sets that serve as the basis for the restore operation
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

When partition data is being restored, the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure creates, reads, or modifies the following tables:

- Original Db2 tables are modified (recovered).
- The table spaces of the partitions to be restored are stopped and are restarted in access mode UT to avoid changes during the data recovery (see **`START DATABASE Db2`** command). Finally, the access mode is set back to RW.

  In case of an error (for example if the stored procedure run is ended through an external **`-CAN THREAD`** command), the UT access mode persists. It is reset to RW when the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure is called again for the same partition and if processing can be completed successfully.
- The REPAIR utility is called to remove the PRO state from the restored partitions.
- The partition data is restored from the image copies that were created at the time of the archiving operation, but the image-copy data sets are not deleted.
- On the accelerator, the declaration of the partition data changes, so that it is classified as regular table data rather than archived partition data.
- The Db2 utilities create temporary data sets named `<USERID>.AQT`, where `<USERID>` is the user ID of the caller.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Environment variables
The job control language (JCL) for the configuration of the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures contains a data definition `(DD) "AQTENV"`. This data definition includes a data set in which environment variables are defined. These variables control the behavior of several stored procedures.

SYSPROC.ACCEL_ARCHIVE_TABLES

Archives Db2 for z/OS table partitions in a storage area on an accelerator. The stored procedure is not suitable for accelerator-only tables.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

DB2 11 for z/OS: Utility Guide and Reference

SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE

Revokes the access to a set of tables that was granted by running the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE stored procedure.

Syntax

```
CALL SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE
(accelerator_name,
remote_location_name,
table_set
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

remote_location_name
Location name of the referencing Db2 subsystem (the system that was granted access). Location name means: the name as listed in the SYSIBM.LOCATIONS table. You can display the location name by running the `-DISPLAY DDF` Db2 command. For an example, see remote_location_name.

The referencing Db2 subsystem must be paired to the accelerator specified by the *accelerator_name* parameter. The owning (remote) Db2 subsystem might use a different name for that accelerator.

table_set
An XML string that lists the tables not to be accessed from the referencing location anymore. The tables must have been added by the SYSPROC.ACCEL_ADD_TABLES command. Names must not occur more than once in the list. The list or string must not be empty. For an example, see table_set.

The tables in the set must have been defined on the accelerator specified by the *accelerator_name* parameter. They must exist in the owning (remote) Db2 subsystem from which this stored procedure is run.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)
Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

Ensure that the proper Db2 for z/OS program temporary fix (PTF) is installed. The catalog table SYSACCEL.SYSACCELERATEDTABLESAUTH must exist.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- DELETE on SYSACCEL.SYSACCELERATEDTABLESAUTH
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
- In addition, the caller of the stored procedure needs *one* of the following privileges:
  - SECADM authorization
  - ACCESSCTRL authorization

  Either of these authorizations is unnecessary if *one* of the following conditions is met:

  - Matching user IDs: For each table in your *`table_set`*, look up the ID in the GRANTOR column of SYSACCEL.SYSACCELERATEDTABLESAUTH. Compare these IDs with the ID of the user who runs the stored procedure. The GRANTOR IDs of all tables in the *`table_set`* must match.
  - Access to the predefined RACF profile DB2ACCEL.FEDERATION. To this end, it is required that:
    1. The stored procedure runs in a WLM environment with an APF-authorized address space. That is, the load module for the stored procedure must reside in an APF-authorized library.
    2. A RACF profile DB2ACCEL.FEDERATION exists in the RACF resource class for Db2 (DSNR).

       Example:

       ```
       -- Setup profile DB2ACCEL.FEDERATION with group FEDADMG
       SETR CLASSACT(DSNR)
       RDEFINE DSNR (DB2ACCEL.FEDERATION) OWNER(DB2OWNER) UACC(NONE)
       PERMIT DB2ACCEL.FEDERATION CLASS(DSNR) ID(FEDADMG) ACCESS(READ)

       SETR RACLIST(DSNR) REFRESH
       ```

The ID can be a group (here: FEDADMG), a single user, or a list of users. A sample job including the profile definition is provided in the SAQTSAMP.AQTSFEDR data set member.

Accessed data and systems

During execution, the stored procedure carries out the following tasks:

- It connects to the accelerator specified by the *accelerator_name* parameter and, to this end, uses the information in the Db2 for z/OS communications database.
- It verifies that an entry exists in the SYSACCEL.SYSACCELERATEDTABLES catalog table for each table in the table set.
- It verifies that for each table in the table set, the SYSACCEL.SYSACCELERATEDTABLESAUTH table contains a valid entry for the specified accelerator.
- For each table in the table set, it removes the entry from the SYSACCEL.SYSACCELERATEDTABLESAUTH catalog table.

Related reference

The message parameter
The message parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES
Removes accelerator-only tables that were created by the SYSPROC.ACCEL_CREATE_REFERENCE_TABLES stored procedure for access by a referencing Db2 subsystem.

SYSPROC.ACCEL_GRANT_TABLES_REFERENCE
Grants a referencing Db2 subsystem the right to access tables that belong to a different, remote Db2 subsystem.

SYSPROC.ACCEL_SET_TABLES_ACCELERATION

Enables or disables query acceleration for accelerator-shadow tables by changing their state. The state is reflected in the corresponding IBM Db2 Analytics Accelerator system tables and determines whether Db2 routes a query to an accelerator.

Details

Technically, the SYSPROC.ACCEL_SET_TABLES_ACCELERATION stored procedure sets a flag in the ENABLE column of the SYSACCEL.SYSACCELERATEDTABLES catalog table in Db2 for z/OS.

Only accelerator-shadow tables in the following states can be enabled for query acceleration:

- *ArchivingInProgress* (if the tables are loaded)
- *Loaded*
- *ReplicationInProgress*
- *UpdateInProgress*

If a table cannot be enabled, the stored procedure stops processing, so that the remaining tables will not be enabled either.

In contrast, enabled tables can always be disabled, no matter what their current state is. A disablement does not affect already running queries. The setting takes effect only on queries that have not yet started.

An accelerator-shadow table can be enabled for query acceleration on multiple accelerators. If all the tables that are relevant for a query are loaded and enabled on more than one accelerator, Db2 for z/OS selects one of the possible candidates for executing the query.

The query routing process does not care whether the data in the tables is up-to-date. Therefore, if you enable tables on multiple accelerators, you are responsible for keeping the data in sync because any accelerator might be chosen for query execution.

Impact on federated access

If query acceleration is disabled for an accelerator-shadow table, a referencing Db2 subsystem should also not be able to run queries against the referencing accelerator-only tables because one can assume that a disabled table is also not updated anymore. That means that the data in the accelerator-shadow table gets more and more obsolete as time passes. This, in turn, would also make the referencing accelerator-only table obsolete, meaning that queries from a referencing Db2 subsystem would not reflect recent data changes in the results.

This is critical because the users behind the queries from the referencing Db2 subsystem might be unaware of this circumstance. However, there is currently no mechanism that blocks the queries. They will continue to run and return results. Therefore, if your queries require data recency, disable referencing queries manually. To this end, run the SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE stored procedure or the equivalent function in your administration client against the tables in question. This will disable accelerated queries against the referencing accelerator-only tables.

Restrictions

You cannot use this stored procedure on accelerator-only tables because these tables are always enabled for query acceleration, and this cannot be changed. The data of an accelerator-only table exists on an accelerator only. Thus, the alternative to process a query in DB2 does not exist. And to use this type of table for queries that are not accelerated does not make sense.

Syntax

```
CALL SYSPROC.ACCEL_SET_TABLES_ACCELERATION
(accelerator_name,
on_off,
table_set,
message);
```

Options description

accelerator_name
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

on_off
Enables or disables query acceleration for one or more accelerator-shadow tables. When set to "ON", this parameter enables the specified tables. When set to "OFF", the specified tables are disabled.

table_set
A list of the tables that you want to process. The XML string that you specify as input for this parameter must conform to the structure of the `tableSet` element in the SAQTSAMP(AQTSXSD1) data set.

The tables must be accelerator-shadow tables that were defined by the SYSPROC.ACCEL_ADD_TABLES stored procedure, or accelerator-only tables. The same table name must not appear more than once in the table set.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <dwa:tableSet xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
      <table name="SALES" schema="SCHM" />
      <table name="PRODUCT" schema="SCHM" />
   </dwa:tableSet>
```

The sample XML code instructs the stored procedure to work on the SALES and PRODUCT tables.

message
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

What to do after changing a table or table space in Db2 for z/OS
In a number of cases, an ALTER TABLE or ALTER TABLESPACE statement in Db2 for z/OS does not involve maintenance work on your accelerator-shadow tables. There are cases, however, in which it does. See which steps you might have to take.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_SET_TABLES_REPLICATION

Enables or disables incremental updates for one or more accelerator-shadow tables. The stored procedure is not suitable for accelerator-only tables or accelerator-archive tables.

You can start the update of accelerator-shadow tables with changed data when the tables have been loaded.

Incremental updates can only be enabled for accelerator-shadow tables in the *InitialLoadPending* or *Loaded* state. If you try to run the SYSPROC.ACCEL_SET_TABLES_REPLICATION stored procedures on tables in another state, the operation ends immediately and no tables are enabled.

If you enable an already loaded table, the following steps are carried out:

1. The data in the accelerator-shadow table is deleted.
2. The accelerator-shadow table is reset to the *InitialLoadPending* state.
3. Query acceleration is turned off for the table.

Hence a reload of the accelerator-shadow table is required before the incremental update process sets in again. This mechanism was introduced to guarantee data consistency. If data changes occurred after the last load of the table, but before its enablement for incremental updates, these changes would not be captured and propagated to the accelerator.

Attention:

- Tables to be included must have a unique constraint (primary key or primary index). If such a key does not exist in the accelerator-shadow table or cannot be determined, you must redefine the table and specify such a key. You cannot do this from your administration client; you must run the corresponding stored procedure (SYSPROC.ACCEL_ADD_TABLES) with a special option. The columns that you choose for the key must contain unique values or form such values when they are combined. The values of the selected key columns will then be written to a hidden column in the accelerator-shadow table. This column, called *informational unique constraint*, then serves as the primary key or primary index. For more information, see the entry about the `table_specifications` parameter of the SYSPROC.ACCEL_ADD_TABLES stored procedure in the *IBM Db2 Analytics Accelerator for z/OS: Stored Procedures Reference*.
- When lock mode ROW is set, Db2 data is unloaded at isolation level CS (cursor stability). Whether a row, a page, or an entire table is locked is determined by the LOCKSIZE defined for the table space of the source tables. To avoid deadlocks during the draining of the spill queue, set this table space to `LOCKSIZE ROW` if you want to use lock mode ROW for your incrementally updated tables.
- If you update Db2 for z/OS tables by running the LOAD utility rather than an INSERT, UPDATE, or DELETE operation, you must set the following parameters (for the LOAD utility):
  - `SHRLEVEL CHANGE`
  - `LOG YES`

  Otherwise, the changes that were made by the LOAD utility are not detected by the incremental update function, and will thus not be reflected in your accelerator-shadow tables.
- The Db2 source tables of your accelerator-shadow tables have an attribute named DATA CAPTURE. The attribute can carry the value Y or N (default), for *yes* or *no*. When incremental updates are enabled for a table, the DATA CAPTURE attribute of the table is set to the value Y. Once set, this attribute value persists, even if the table is disabled at a later time. Bear this in mind, especially if you run applications that use the DATA CAPTURE attribute.

The DATA CAPTURE attribute is set by an ALTER TABLE statement, which is run as part of the SYSPROC.ACCEL_SET_TABLES_REPLICATION stored procedure. However, the attribute can only be set successfully if the ID of the user who runs the stored procedure has ALTER TABLE authorization. If this is not the case, a database administrator must set the attribute for all tables in Db2 for z/OS.

- You might have to reload or even remove tables from an accelerator after the execution of an ALTER TABLE or ALTER TABLESPACE statement in Db2 for z/OS. For more information, see *What to do after changing a table or table space in Db2 for z/OS*in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.
- IBM InfoSphere Change Data Capture for z/OS uses a metadata table called DMMD_SIGNALS. This table is a kind of an interface for the entire process. It does not make sense to select it for incremental updates. Ensure that you do not select it inadvertently as doing so will lead to unpredictable and unwanted side-effects.
- If a table is dropped from Db2 for z/OS, you normally have to remove all related accelerator-shadow tables manually from the attached accelerators to avoid catalog mismatches. There is one exception: If the tables are enabled for replication, and replication is carried out by the IBM Integrated Synchronization function, the accelerator-shadow tables are removed automatically.

Restrictions:

- It is not possible to enable tables if the table name or the schema name contains GB18030 characters (Simplified Chinese) of Unicode plane 2 (U+20000-U+2FFFF: Supplementary Ideographic Plane). Trying to do so results in an error.
- You cannot enable incremental updates for accelerator-only tables tables because this type of table has no data in Db2 that could be propagated to an accelerator.

Impact on federated access

Enabling an accelerator-shadow table for incremental updates has implications if federated access is configured for this table.

The table state changes from *Loaded* to *InitialLoadPending* to enforce a full reload of the table. This ensures that the data in both tables is the same before the incremental update process starts, so that updates cannot go amiss. However, a table in state *InitialLoadPending* is empty. Hence it cannot return correct results when it receives a query. To avoid this situation, referencing accelerator-only tables created for federated access are removed when incremental updates are enabled for the original table. If a referencing Db2 subsystem sends a query against the table, the query fails and `SQLCODE -204` (object not found) is returned. If you run the SYSPROC.ACCEL_GET_TABLES_INFO stored procedures to obtain information about tables, it will report a `Violated on accelerator` error for the missing reference table.

Syntax

```
CALL SYSPROC.ACCEL_SET_TABLES_REPLICATION
( accelerator_name,
  on_off,
  table_set,
  message
);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

on_off

The value `ON` enables incremental updates for all accelerator-shadow tables that are specified by the *`table_set`* parameter. The value `OFF` does the opposite.

table_set

A list of the accelerator-shadow tables that you want to process. The same table must not appear more than once in the table set. The XML string that you specify as input for this parameter must conform to the structure of one of the following elements in the SAQTSAMP(AQTSXSD1) data set:

Examples:

tableSet

See this piece of sample XML code, which instructs the stored procedure to work on the SALES and PRODUCT tables:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <aqt:tableSet xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011"
    version="1.0">
      <table name="SALES"   schema="BCKE" />
      <table name="PRODUCT" schema="BCKE" />
   </aqt:tableSet>
```

tableSetForSetTablesReplication

The `tableSetForSetTablesReplication` element is an extension of `tableSet`. It allows you to set an attribute (`overrideExistingKeys`), which, as the name suggests, allows you to override certain settings. This leads to an overall performance improvement for earlier product versions. If you only use this or a later product version, you can ignore this element.

```
<?xml version="1.0" encoding="UTF-8" ?>
   <aqt:tableSetForSetTablesReplication
    xmlns:aqt="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0"
    overrideExistingKeys="true">
      <table name="SALES" schema="BCKE" />
      <table name="PRODUCT" schema="BCKE" />
   </aqt:tableSetForSetTablesReplication>
```

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

As a result of calling the SYSPROC.ACCEL_SET_TABLES_REPLICATION stored procedure, IBM InfoSphere Change Data Capture for z/OS carries out certain steps to enable incremental updates on the accelerator.

In addition, the stored procedure updates the IBM Db2 Analytics Accelerator for z/OS catalog on the accelerator.

Related reference

What to do after changing a table or table space in Db2 for z/OS
In a number of cases, an ALTER TABLE or ALTER TABLESPACE statement in Db2 for z/OS does not involve maintenance work on your accelerator-shadow tables. There are cases, however, in which it does. See which steps you might have to take.

SYSPROC.ACCEL_TEST_CONNECTION

Tests the connection between a Db2 subsystem and an accelerator. A pairing or authentication of the accelerator is not required before running this procedure.

Details

The SYSPROC.ACCEL_TEST_CONNECTION stored procedure allows you to check:

- Whether the mainframe computer can contact (ping) the accelerator over the network
- Whether the network path from Db2 to the accelerator has been properly configured
- Whether the connection between the Db2 subsystem and the accelerator works after the pairing process
- The data transfer rate (network speed) between Db2 and the accelerator
- The versions of software components on a particular accelerator and other information

Syntax

```
CALL SYSPROC.ACCEL_TEST_CONNECTION
(diagnostic_specification,
diagnostic_output,
message);
```

Options description

diagnostic_specification

An XML input string that specifies which tests are to be run. The XML string must conform to the structure of the `diagnosticCommand` element in the SAQTSAMP(AQTSXSD1) data set. The string can contain several XML elements. Each of these elements starts a certain test. You can run all the tests separately. If more than one test is specified and one of them fails, the remaining tests are completed. The tests are carried out in the same order as listed here:

<ping ip="*ip_address*" />

Tests the IP connection between the stored procedures and a specific accelerator. For this test, the **`ping`** command in IBM UNIX System Services is used.

<connect accelerator="name" />

Tests the connectivity including authenticated access between the stored procedures and a specific accelerator. The stored procedure uses the handshake mechanism and the data of the Communications database and the SYSACCEL.SYSACCELERATORS table.

<mtupath ip="*ip_address*" port="*number*" />

Tests the validity of the maximum transmission unit (MTU) that has been chosen for the network path between the z/OS system and the accelerator. The MTU is the largest possible unit of data that can be sent on a given physical medium in a single frame. This test can also be used for installation verification. In addition to the IP address, specify the DRDA port number (`1400` by default).

<networkSpeed [...] />

Tests the data transfer rate of the network connecting the z/OS system and the accelerator. Use this test to analyze network performance bottlenecks that might be the cause of load performance problems. Specify the following attributes:

- The name of the accelerator (`accelerator = "`*`name`*`"`)
- The total size of the data chunk to be tested (`totalNumberBytes="`*`value`*`"`). Set this to a value of `10000000` (10 MB) or higher.
- The size of the individual data units that make up the total data size (`dataBatchSize="`*`size`*`"`). Set this to a value of `320000` or higher.
- The number of parallel connections (`parallelConnections="`*`number`*`").` Set this to the value of the AQT_MAX_UNLOAD_IN_PARALLEL environment variable. The `totalNumberBytes` value is multiplied by this value. The product is the total number of bytes that are transferred between Db2 for z/OS and the accelerator throughout the test.
- An identifier for the sample (`seed="`*`identifier`*`"`). You can use any value that is easily recognizable, for example `12345`.

<version [...] />

Returns the version and other information about the software that is installed on an accelerator. You can submit the version element with different attributes:

<version ip="*ip_addr*" port="1400" />

This XML string can be used to probe an accelerator that has not yet been authenticated for use with a Db2 subsystem or data sharing group (no pairing). Since a name has not been registered yet, you must provide the IP address and the network port.

<version accelerator="*name*" />

This XML string can be used to probe an authenticated (paired) accelerator.

`diagnostic_output`

The XML output string that contains the test results. The structure of this string conforms to that of the `diagnosticOutput` element in the SAQTSAMP(AQTSXSD1) data set. If all tests were completed successfully, the stored procedure returns SQL code `0` and message `AQT10000I`. If one or more tests fail, you receive an error SQL code and message `AQT10075E`.

`message`

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *`message`* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Example A

The following XML input string (value of the *`diagnostic_specification`* parameter) instructs the stored procedure to test the IP connectivity, connectivity with authenticated access, MTU settings, and the network performance:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:diagnosticCommand
xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
<ping ip="203.0.113.8" />
<connect accelerator="TESTACC1" />
<mtupath ip="203.0.113.8" port="1400" />
<networkSpeed accelerator="TESTACC1" totalNumberBytes="1000000000"
dataBatchSize="320000" parallelConnections="20" seed="12345"/>
</aqttables:diagnosticCommand>
```

The output looks similar to the following XML block (the output changes frequently and might not be exactly as shown here) :

```
<?xml version="1.0" encoding="UTF-8" ?>
<testConnect:diagnosticOutput
 xmlns:testConnect="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">

<diagnostics  action="ping" success="true">

    CS V1R13: Pinging host 203.0.113.8

     Ping #1 response took 0.000 seconds.
          Ping #2 response took 0.000 seconds.
          Ping #3 response took 0.000 seconds.
          Ping #4 response took 0.000 seconds.

    Ping test returned with rc: 0

</diagnostics>

<diagnostics  action="connect" success="true">
</diagnostics>

<diagnostics  action="networkSpeed" success="true">

     Server output:
     -------------------- Thread performance counters ------------------

      +++++ Performance counter: thread0 +++++

        thread-0: Total data received:            1000000000 bytes

        thread-0: Single buffer size:             320000 bytes
        thread-0: Receive calls:                  3125

        thread-0: Elapse time for receive calls:   3651 ms
        thread-0: Send buffer size:                262144 bytes
        thread-0: Receive buffer size:             16777216 bytes
        thread-0: Transfer rate:                   261.209070503 MB/sec

      +++++ Performance counter: thread1 +++++

        thread-1: Total data received:             1000000000 bytes
        thread-1: Single buffer size:              320000 bytes
        thread-1: Receive calls:                   3125
        thread-1: Elapse time for receive calls:   3747 ms
        thread-1: Send buffer size:                262144 bytes
        thread-1: Receive buffer size:             16777216 bytes
        thread-1: Transfer rate:                   254.516764453 MB/sec

      +++++ Performance counter: thread2 +++++

        thread-2: Total data received:             1000000000 bytes
        thread-2: Single buffer size:              320000 bytes
        thread-2: Receive calls:                   3125
        thread-2: Elapse time for receive calls:   3576 ms
        thread-2: Send buffer size:                262144 bytes
        thread-2: Receive buffer size:             16777216 bytes
        thread-2: Transfer rate:                   266.687448659 MB/sec

      +++++ Performance counter: thread3 +++++

        thread-3: Total data received:             1000000000 bytes
        thread-3: Single buffer size:              320000 bytes
        thread-3: Receive calls:                   3125
        thread-3: Elapse time for receive calls:   3702 ms
        thread-3: Send buffer size:                262144 bytes
        thread-3: Receive buffer size:             16777216 bytes
        thread-3: Transfer rate:                   257.61056629 MB/sec

     -------------------Overall performance counter --------------------

        Number of sender threads:               20

        Total number of bytes received:         4000000000 bytes

        Total elapse time for receiving data:   3747 ms

        Transfer rate End-to-End:               1018.06705781 MB/sec

     -------------------- Connector details ----------------------------

       DRDA socket connector details:

       Socket details

       socket number: 25

         .
         .
         .

       Total performance counter:

           @perf: Single thread buffer size:                 320000 bytes
           @perf: Number of sender threads:                  20
           @perf: Total number of bytes sent:                4000000000 bytes
           @perf: Total elapse time for send calls with ack: 3.74664 sec
           @perf: Transfer rate:                             1067623542.06 bytes/sec

       Test finished.

</diagnostics>

<diagnostics  action="mtupath" success="true">

      z/OS max segment size: 8952 byte

      Accelerator max segment size: 8952 byte

      Accelerator -> z/OS ping output:

      PING 203.0.113.9 (203.0.113.9) 8952(8980) bytes of data.

      8960 bytes from 203.0.113.9: icmp_seq=1 ttl=64 time=0.171 ms
      8960 bytes from 203.0.113.9: icmp_seq=2 ttl=64 time=0.194 ms
      8960 bytes from 203.0.113.9: icmp_seq=3 ttl=64 time=0.176 ms
      8960 bytes from 203.0.113.9: icmp_seq=4 ttl=64 time=0.166 ms
      8960 bytes from 203.0.113.9: icmp_seq=5 ttl=64 time=0.170 ms
      8960 bytes from 203.0.113.9: icmp_seq=6 ttl=64 time=0.181 ms
      8960 bytes from 203.0.113.9: icmp_seq=7 ttl=64 time=0.188 ms
      8960 bytes from 203.0.113.9: icmp_seq=8 ttl=64 time=0.170 ms
      8960 bytes from 203.0.113.9: icmp_seq=9 ttl=64 time=0.179 ms
      8960 bytes from 203.0.113.9: icmp_seq=10 ttl=64 time=0.182 ms


      ---203.0.113.9 ping statistics ---

      10 packets transmitted, 10 received, 0% packet loss,
      time 9001ms
      rtt min/avg/max/mdev = 0.166/0.177/0.194/0.017 ms



      z/OS -> Accelerator ping output:
      CS V1R13:
      Pinging host 203.0.113.8

      Ping #1 response took 0.000 seconds.
      Ping #2 response took 0.000 seconds.
      Ping #3 response took 0.000 seconds.
      Ping #4 response took 0.000 seconds.
      Ping #5 response took 0.000 seconds.
      Ping #6 response took 0.000 seconds.
      Ping #7 response took 0.000 seconds.
      Ping #8 response took 0.000 seconds.
      Ping #9 response took 0.000 seconds.
      Ping #10 response took 0.000 seconds.


      Socket details

       socket number: 0

       .
       .
       .

       TCP option - maximum segment size for outgoing
       TCP packets: 8952 byte

       TCP option - no delay: yes
       IP option - time to live: 0

       IP option - multicast time to live: 0

       IP option - multicast packets should be looped back to the local
       sockets: no

      Ping returned with rc: 0

</diagnostics>
</testConnect:diagnosticOutput>
```

Note that in this case, the `success` attribute always carries the value `"true"`.

If, on the other hand, the tests did not run successfully, the output might look like this:

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:diagnosticOutput
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
    <diagnostics action="ping" success="false">
        CS V1R10: Pinging host 203.0.113.8
        Ping #1 timed out
        Ping #2 timed out
        Ping #3 timed out
        Ping #4 timed out
    </diagnostics>
</aqttables:diagnosticOutput>
```

In this case, the IP connectivity test failed (`success="false"`). The specified accelerator is unreachable.

Example B

The `<version>` element is used in the XML input string (value of the *`diagnostic_specification`* parameter) :

```
<?xml version="1.0" encoding="UTF-8" ?>
<aqttables:diagnosticCommand
 xmlns:aqttables="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
    <version ip="162.108.0.102" port="1400"/>
</aqttables:diagnosticCommand>
```

The output looks similar to the following extract:

```
<?xml version="1.0" encoding="UTF-8" ?>
<testConnect:diagnosticOutput
 xmlns:testConnect="http://www.ibm.com/xmlns/prod/dwa/2011"
 version="1.0">
<diagnostics  action="version" success="true">
<accelerator>
    <versionInformation>
      <component name="Accelerator" version="7.1.0.201711071105" />
      <component name="Access Server" version="" />
      <component name="BackendDBS" version="11.1.9.0 [Build 1709041600]" />
      <component name="Linux Operating System" version="CentOS Linux
       release 7.4.1708 (Core)" />

    </versionInformation>
    <DRDAProtocolLevel>AQT_0000000000060001_0000000000000041_7800000000000000
    </DRDAProtocolLevel>
    <highestSupportedCompatibilityLevel>65</highestSupportedCompatibilityLevel>
</accelerator><storedProcedure>
     <versionInformation>
         <component name="version" version="7.1.0" />
         <component name="interface-version" version="65" />
         <component name="build-label" version="20171109-0814" />
         <component name="build-timestamp" version="2017-11-09 09:21:21" />
         <component name="build-platform" version="OS/390 SCLM 25.00 04 2827" />
         <component name="support-level" version="5" />
     </versionInformation>
     <DRDAProtocolLevel>AQT_0000000000060001_0000000000000041_6000000000000000
     </DRDAProtocolLevel>
     <highestSupportedCompatibilityLevel>65</highestSupportedCompatibilityLevel>
     <setting name="AQT_MAX_UNLOAD_IN_PARALLEL" value="20" />
     <setting name="AQT_MAX_RETRIES_DSNUTILU" value="2" />
     <setting name="AQT_SECONDS_BEFORE_RETRY_DSNUTILU" value="60" />
     <setting name="AQT_QUOTEFIX1026" value="SET" />
     <setting name="AQT_SKIP_VALIDATE_XML" value="SET" />
     <setting name="AQT_UTILITY_TMP" value="ISAOV2.&amp;USERID..AQT.&amp;UNIQ." />
     <setting name="AQT_RTS_EXTERNALIZATION_AUTH_FAILURE" value="ERROR" />
     <setting name="AQT_DO_SYSLOGGING" value="SET" />
     <setting name="AQT_KEEPIDLE_INTERVAL" value="60" />
</storedProcedure>
```

Prerequisites

For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment.

- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related concepts

Error handling
The SQL CALL statement invoking a stored procedure returns an *sqlstate* field value, an *sqlcode* field value, and diagnostic information (*sqlerrmc*) in the SQL communication area (SQLCA).

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

AQT10075E
The *failingTest* test failed.

SYSPROC.ACCEL_UPDATE_CREDENTIALS

Changes the authentication token that Db2 and the stored procedures use to communicate with an accelerator.

Use this stored procedure if you must comply with security policies that require a regular change of all authentication information, such as passwords. You can run the **`<getAcceleratorInfo>`** command of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure to return a timestamp that verifies regular changes of authentication information.

Tip: Updating the authentication token does not affect subsequent operations of the accelerator, but queries running at the same time might fail occasionally. It is therefore recommended that you disable the accelerator before you call the SYSPROC.ACCEL_UPDATE_CREDENTIALS stored procedure.

Syntax

```
CALL SYSPROC.ACCEL_UPDATE_CREDENTIALS
(accelerator_name,
message);
```

Options description

`accelerator_name`
The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

`message`
For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- Updating the credentials requires an accelerator that was successfully added to the system configuration by running the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure (successful pairing process).

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E

The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

SYSPROC.ACCEL_UPDATE_SOFTWARE2

Transfers software updates from the z/OS UNIX file system (z/FS) to an accelerator, lists available software versions, activates versions that are available on an accelerator, and removes obsolete software packages from an accelerator.

Details

Update packages to be transferred by the SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure must be located on an IBM Z server or in a Global Temporary Table (GTT). Both targets can be addressed when you download packages from IBM Fix Central.

If you prefer to install, transfer, and activate software updates by running this stored procedure rather than the corresponding functions in your administration client, you must extract the hash value for each package from the checksum files and submit the hash value as part of the XML input string for the stored procedure.

Download packages on IBM Fix Central usually come with a checksum file for every supported algorithm. Supported algorithms are SHA256, SHA1 and MD5. The corresponding files on IBM Fix Central are:

- `sha256sums.txt`
- `sha1sums.txt`
- `md5sums.txt`

Any of these files contains one line for each package. Every line consists of two values. The first value is the hash used for verification (the value you have to extract); the second value is the corresponding package name.

You can set the `AQT_HOST_PACKAGE_DIRECTORY` environment variable in the `AQTENV` data set to specify the default location for update packages. The variable must specify an absolute directory path. The SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure looks into this directory only. Subdirectories are not searched.

Restriction: In product version 7, you no longer find the `hostPackageDirectory` attribute to override the corresponding path setting in the AQTENV data set. Hence you must modify the path in the AQTENV data set if you want to change the package location.

More than one version can be transferred at a time. However, you cannot install more than one version on the same accelerator. This implies that you can also not switch between different versions.

If an error occurs during a package transfer, the erroneous package and all subsequent packages will not be transferred.

The stored procedure internally sets the accelerator to the *Quiesced* state before a new version is activated. Running queries and administration tasks are completed before this is done.

During the activation or rollback of a software update, the TCP/IP connection between the stored procedure and the accelerator is interrupted (in most cases, this does not apply to updates of the Access Server or the replication engine for incremental updates). To verify that the version change was successful, the stored procedure tries to reconnect to the accelerator. A timeout mechanism is in place in case the accelerator takes too long to respond.

If SYSPROC.ACCEL_UPDATE_SOFTWARE2 cannot activate the specified version, it reactivates the version that was previously used. Thus, the stored procedure will always be able to reconnect to a running accelerator.

Finally, SYSPROC.ACCEL_UPDATE_SOFTWARE2 calls SYSPROC.ACCEL_CONTROL_ACCELERATOR with the **`GetActivationLog`** subcommand. The log that is produced shows the versions of the installed software packages and progress information about running installations. Mind that an installation might involve a downtime of the accelerator, during which such a log cannot be produced.

Syntax

```
CALL SYSPROC.ACCEL_UPDATE_SOFTWARE2
(accelerator_name,
update_command,
update_result,
message);
```

Options description

accelerator_name

The unique name of the accelerator. This accelerator must have been defined by the SYSPROC.ACCEL_ADD_ACCELERATOR2 stored procedure.

update_command

An XML input string that initiates the code transfer to the accelerator, instructs the stored procedure to list available software versions on an accelerator, in the proper z/FS directory, or change the active software version. The string must conform to the structure of the `softwareUpdate` element in the SAQTSAMP(AQTSXSD1) data set. Use the following XML elements to start the sub-functions of the stored procedure:

<listDeployedPackages>

To list the software versions (transferred packages) that are currently available on an accelerator.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdate
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.1">
   <listDeployedPackages/>
</dwa:softwareUpdate>
```

<listZPackageDirectory>

To list the software packages that are currently available in the zFS of your z/OS data server. You can use the output of this option as input for the `<deployPackage>` option.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdate
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.1">
  <listZPackageDirectory />
</dwa:softwareUpdate>
```

This input string instructs the SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure to list the available software packages in the host package directory of the zFS. This is the directory that the `AQT_HOST_PACKAGE_DIRECTORY` environment variable points to. This variable is set in the `AQTENV` data set.

<deployPackage>

To transfer one or more packages from the zFS of your z/OS data server to the accelerator. This option does not produce any output.

Example A:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdate
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
   <deployPackage
    fileName="iias-1.0.3.0-20171120090514b50_accelerator.tar.gz"
    source="FILE"
    hash="ce07195b659e75f4e1db43552860070061f156a98bb37b672b101ba6e3ddf30c"
    algorithm="SHA256" />
</dwa:softwareUpdate>
```

In this example, an update package called `iias-1.0.3.0-20171120090514b50_accelerator.tar.gz` is transferred from the host package directory in the zFS (as set by the `HOST_PACKAGE_DIRECTORY` environment variable) to the specified accelerator.

Example B:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdate
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
   <deployPackage
    fileName="iias-1.0.3.0-20171120090514b50_accelerator.tar.gz"
    source="GTT"
    hash="ce07195b659e75f4e1db43552860070061f156a98bb37b672b101ba6e3ddf30c"
    algorithm="SHA256" />
</dwa:softwareUpdate>
```

In this example, an update package that was stored in a global temporary table is transferred to the specified accelerator.

`<activateDeployedPackage>`

To activate a particular package:

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdate
 xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.0">
   <activateDeployedPackage fileName="SailfishFin1.rpm" essentialsOnly="true" />
</dwa:softwareUpdate>
```

The XML code in this example instructs SYSPROC.ACCEL_UPDATE_SOFTWARE2 to activate a software package named `SailfishFin1.rpm`.

The `essentialsOnly` attribute is set to the value `"true"`, which means that not all updates included in the package are applied. The activation is limited to the most essential components, such as the accelerator software, the database engine, the web console, and the DRDA gateway. Updates of less important components, such as the lower-level firmware and operating system components are skipped. This reduces the time needed for the activation, during which the accelerator cannot be used. On average, the activation period is reduced to a time span between one and two hours.

<removeDeployedPackage>

To remove a particular software package (version) from an accelerator

Example:

```
<dwa:softwareUpdate
   xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.1">
      <removeDeployedPackages>
         <package
          fileName="iias-1.0.3.0-20171112090514b50_accelerator.tar.gz"/>
      </removeDeployedPackages>
</dwa:softwareUpdate>
```

The XML code in this example instructs SYSPROC.ACCEL_UPDATE_SOFTWARE2 to remove the package `iias-1.0.3.0-20171112090514b50_accelerator.tar.gz`.

You can only remove packages that are not active anymore. That is, if a package to be removed is still active, you must first activate a different package or version.

update_result

An XML output string that is created as a result of using the `<listDeployedPackages>` or `<listZPackageDirectory>` element in the *`update_command`* input string.

For `<listDeployedPackages>` and `<listZPackageDirectory>`, the output parameter conforms the structure of the `<softwareUpdate>` element in the SAQTSAMP(AQTSXSD1) data set. For all other functions, this output parameter is NULL.

Example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<dwa:softwareUpdateResult version="1.0">
  <ZPackageDirectoryListing directoryName="/u/download/fixCentral">
    <deployablePackage source="FILE"
      fileName="iias-1.0.3.0-20171112090514b50_accelerator.tar.gz"
         fileSizeInBytes="913042363"
      hash="7de0b93ed997941dd1b6cde2c87625b2afd7b72759a5631c86bbc6c4aee61e82"
         algorithm="SHA256" />
    <deployablePackage source="FILE"
      fileName="iias-1.0.3.1-20171214060618b60_accelerator.tar.gz"
         fileSizeInBytes="9530186651"
      hash="3412b93ed997941dd1b6cde2c87625b2afd7b72759a5631c86bbc6c4aee61e82"
         algorithm="SHA256" />
  </ZPackageDirectoryListing>
</dwa:softwareUpdateResult>
```

This XML code is the result of using the `<listZPackageDirectory>` command as input. It lists two packages in the host package directory:

- `iias-1.0.3.0-20171112090514b50_accelerator.tar.gz`
- `iias-1.0.3.1-20171214060618b60_accelerator.tar.gz`

It is assumed that this directory is `/u/download/fixCentral`.

message

For the description, follow the appropriate link under **Related reference** at the end of this topic.

Result sets

The stored procedure always returns two result sets:

First result set (SP_TRACE_CURSOR)

Depending on the trace configuration in the *message* input parameter, this result set is empty or contains trace information about the stored procedure execution. It is identified by the cursor SP_TRACE_CURSOR and contains the following columns:

- SEQID of type INTEGER
- TRACEDATA of type VARBINARY, with a maximum length of 32698 characters

The information in the TRACEDATA column is encoded in UTF-8. It is intended for analysis by IBM support. If the length of a value exceeds 32698 characters, it is split into multiple result set rows. The rows are concatenated in ascending order of SEQID.

Second result set (MESSAGES_CURSOR)

This result set contains an XML string that contains the same messages as the MESSAGE output parameter. In contrast to the MESSAGE output parameter, the result set does not have a 64 KB size limitation. Therefore, it always contains the whole set of `<message>` elements (no truncation). The structure of the XML string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTSXSD1) data set. The result set is identified by a cursor named MESSAGES_CURSOR and contains the following columns:

- SEQID of type INTEGER
- MESSAGES of type VARBINARY, with a maximum length of 32698 characters

The rows in the result set are concatenated in ascending order of the values in the SEQID column. The information in the MESSAGES column is encoded in UTF-8.

Prerequisites

- For the user ID that calls the stored procedures on z/OS, you must define an OMVS segment in the Resource Access Control Facility (RACF).
- The *accelerator_name* parameter must specify an accelerator name that is listed in the catalog tables of the product.

Authorizations for z/OS

On z/OS, the user ID under which this stored procedure is run must have the following privileges:

- EXECUTE on the stored procedure
- EXECUTE on the SYSACCEL.* packages
- MONITOR1 authorization
- Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
- For the functions **`listZPackageDirectory`** and **`deployPackage`**: Read access to the directory that the AQT_HOST_PACKAGE_DIRECTORY environment variable points to.
- RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
- Sub-commands of the `update_command` option might require further authorizations.
- EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Accessed data and systems

The SYSPROC.ACCEL_UPDATE_SOFTWARE2 stored procedure reads the IBM Db2 Analytics Accelerator for z/OS catalog tables in your database management system.

In addition, it executes requests on the accelerator. This might include a reboot of the IBM Db2 Analytics Accelerator for z/OS server.

Related reference

The message parameter
The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Transcript of the SAQTSAMP(AQTSXSD1) data set member
Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

Related information

AQT10402E
The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.

The `message` parameter

The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

The `message` parameter is always used as an output parameter and produces an XML string with information about the success or failure of a stored-procedure run. The structure of this string conforms to that of the `messageOutput` element in the SAQTSAMP(AQTXSD1) data set.

You can use the `message` parameter as an *input* parameter for a variety of purposes.

- “Controlling the trace behavior” on page 120
- “Setting the compatibility level” on page 120
- “Setting or overriding environment variables” on page 122

The input string must conform to the structure of the `<messageControl>` element in the SAQTSAMP(AQTSXSD1) data set. The `message` parameter is defined as an INOUT character large object (CLOB) with a maximum size of 64 KB. If the 64 KB CLOB is unable to contain all `<message>` elements, some of these elements are omitted. Warning AQT10004W is issued in cases like this; it shows the number of skipped messages. To see all messages in cases where the 64 KB CLOB does not suffice, open the result set identified by the MESSAGES_CURSOR. This result set always contains the whole set of messages.

Important: NULL, an empty string, or space characters are allowed as an input value for the message parameter. In this case, the stored procedure uses a default value.

The default values are defined in the SAQTSAMP(AQTDEF6) data set member. This data set member is loaded by the `AQTDEF6  DD` statement in the Workload Manager (WLM) startup job.

Controlling the trace behavior

To control the trace behavior and write trace information about the stored procedure to a result set or file. To do so, you specify an input XML string that uses the `<traceConfig>` element:

Example A:

```
<?xml version= "1.0" encoding= "UTF-8" ?>
<spctrl:messageControl xmlns:spctrl = "http://www.ibm.com/xmlns/prod/dwa/2011"
 version = "1.2" >
   <language>en_US</language> <!-- currently ignored -->
   <traceConfig location = "/tmp/spcreate.trace" keepTrace = "true"
    traceFileSizeInMB = "10" >

      <component name = "PROCEDURE" level = "DEBUG" />
   </traceConfig>
</spctrl:messageControl>
```

If you use this example, stored-procedure trace information is written to the `/tmp/spcreate.trace` file in the z/OS UNIX file system (zFS) because the `location` attribute is set to this value. The trace file is allowed to grow to a size of 10 MB. When this size is exceeded, the oldest entries are removed from the trace file to free up space for the newest entries. The trace information will not be deleted after the stored-procedure run (`keepTrace=True`). The information is collected with a detail level of DEBUG.

Example B:

```
<?xml version= "1.0" encoding= "UTF-8" ?>
<spctrl:messageControl xmlns:spctrl = "http://www.ibm.com/xmlns/prod/dwa/2011"
 version = "1.2" >
   <language>en_US</language> <!-- currently ignored -->
   <traceConfig location = "/u/myTraceDirectory/"
    keepTrace = "onError" traceFileSizeInMB = "10" >
      <component name = "PROCEDURE" level = "DEBUG" />
   </traceConfig>
</spctrl:messageControl>
```

In contrast to the previous example, the trace information is written to the `/u/myTraceDirectory/` directory. A file name has not been specified, so the default name is chosen. The trace information is deleted if no error occurs. If an error causes the procedure to fail, the trace information is kept at the specified location (`keepTrace=onError`).

Tip: In an automated setup, it is favorable to configure tracing for stored procedures as shown in **Example B**, that is:

- Specify a writable location, for example `location="/u/someDirectory/"` so that the trace information is written to a file. Trace files are much easier to read and interpret than result sets.

  You can also specify `location="GENERATED"`. In this case, the trace file is written to the `/tmp` directory of the z/OS UNIX file system. The file name will be chosen at random.
- Keep trace files in case of errors only by using the `keepTrace="onError"` attribute on the `message` input parameter of the stored procedures.

Setting the compatibility level

- To set the compatibility level for the stored-procedure output. This way, you can exclude newer elements or attributes from the XML output, which might be necessary to keep the output compatible with self-written applications that were developed for earlier product versions.

Example:

```
<dwa:messageControl xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011" version="1.2">
  <compatibilityLevel>77</compatibilityLevel>
</dwa:messageControl>
```

In this example, the compatibility level has been set to the value 77. This compatibility level makes the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure produce version 7.5.8 output. This output does not include the `last_access_timestamp` and `access_count` attributes because these attribute were introduced later, with product version 7.5.11.

Table 10 on page 121 lists recent compatibility levels and the (additional) attributes that were introduced with these levels.

Table 10. Compatibility levels and attributes introduced with these levels

Compatibility levels	**Additional output attributes by stored procedure**	**Introduced with stored procedure version**
76	**SYSPROC.ACCEL_GET_QUERIES2** – `waitTimeHTAPSec` – `timeoutsHTAP`	7.5.7
77	**SYSPROC.ACCEL_GET_QUERIES2** – `prepareTimeSec` – `sessionID` **SYSPROC.ACCEL_GET_TABLES_INFO** – `load_backend_statistics_collection_timestamp` – `archive_backend_statistics_collection_timestamp`	7.5.8
79	**SYSPROC.ACCEL_GET_QUERIES2** – `applicationStallTime` **SYSPROC.ACCEL_GET_TABLES_INFO** – `last_access_timestamp` – `access_count`	7.5.11

Note: The compatability levels are cumulative. That is, if you specify 79, you also get the output attributes that were introduced with earlier compatibility levels, such as 76 and 77.

For a description of the listed output attributes, see the appropriate reference chapter about the stored procedure:

– SYSPROC.ACCEL_GET_QUERIES2: “Fourth result set” on page 45

  - SYSPROC.ACCEL_GET_TABLES_INFO: “Second result set (TABLE_INFO_STATES_CURSOR)” on page 63

To ensure that the stored-procedure output is compatible with future product versions, proceed as follows:

1. Determine the current interface level of the stored procedures by running any IBM Db2 Analytics Accelerator stored procedure. For the `message` parameter, specify just the `versionOnly` attribute, with a value of `true`. The input XML code for the `message` looks like this:

   ```
   <?xml version= "1.0" encoding= "UTF-8" ?>
   <spctrl:messageControl xmlns:spctrl = "http://www.ibm.com/xmlns/prod/dwa/2011"
   version = "1.0" versionOnly = "true" ></spctrl:messageControl>
   ```

2. Make a note of the interface version number that is returned as the query result.
3. Include this number as the value of the `compatibilityLevel` attribute in the `message` input parameter when your custom applications call IBM Db2 Analytics Accelerator stored procedures.

Setting the `compatibilityLevel` attribute to the correct value, you need not change your custom applications after installing a newer version of the stored procedures.

Setting or overriding environment variables

To set or override environment variables temporarily on a per-call basis. Settings will be valid only for the duration of a stored-procedure call. The following environment variables can be set in this way:

- AQT_MAX_UNLOAD_IN_PARALLEL
- AQT_ARCHIVE_COPY1
- AQT_ARCHIVE_COPY2
- AQT_ARCHIVE_RECOVERYCOPY1
- AQT_ARCHIVE_RECOVERYCOPY2

Example:

```
<?xml version= "1.0" encoding= "UTF-8" ?>
<aqttables:messageControl
 xmlns:aqttables = "http://www.ibm.com/xmlns/prod/dwa/2011" version = "1.2">
   <traceConfig>
      <component name = "PROCEDURE" level = "INFO" />
   </traceConfig>
   <environment>
      <environmentVariable name = "AQT_MAX_UNLOAD_IN_PARALLEL" value = "2" />
   </environment>
</aqttables:messageControl>
```

Running stored procedures from a batch job

To administer IBM Db2 Analytics Accelerator for z/OS by using the MVS™ job control language (JCL), the stored procedures must be called from a z/OS environment. You can achieve this by using one of the methods presented here.

Using the Db2 command line processor from UNIX System Services

To start a batch job, you can use the Db2 command line processor from UNIX System Services. In this case, the z/OS BPXBATCH utility is used to invoke the command line processor.

Before you begin

Make sure that the following conditions apply:

- The Db2 libraries for IBM Db2 Analytics Accelerator for z/OS have been installed.
- The IBM Db2 Analytics Accelerator for z/OS stored procedures have been installed.

- The Db2 command line processor has been installed.

  Note: The Db2 command line processor is a Java™ application that requires IBM Data Server drivers for JDBC.
- The accelerator has been successfully added to the Db2 subsystem from your administration client.

About this task

To create a suitable batch job with minimum effort, you can copy and customize one or more of the Db2 command-line scripts and UNIX System Services shell scripts that are included in the SAQTSAMP data set.

Procedure

1. Create a separate Db2 command line processor properties-file for the user who will run the batch job. Make sure that the user uses this properties file, for example by adding the CLPPROPERTIESFILE environment variable to the `.profile` of this user. Remove read access to this file for everybody else. This way, you can safely add a connection alias for the accelerator network connection to the properties file, including the password. Use the following syntax to add this information to the properties file:

   ```
   <connection_alias>=<db2host>:<db2port>/<db2location>,<uid>,<password>
   ```

   where

   `<db2host>`
   Is the host name or IP address of the IBM Z server on which Db2 runs

   `<db2port>`
   Is the port for network connections between the Db2 command-line client and the Db2 host. To identify the Db2 port, run the following Db2 command:

   ```
   - <ssid> DIS DDF
   ```

   where `<ssid>` is the Db2 subsystem ID.

   `<db2location>`
   Is the location of the Db2 subsystem that is supposed to interact with an accelerator.

   `<uid>`
   Is the ID of the user running the batch job

   `<password>`
   The password of the user running the batch job
2. Copy the following Db2 command-line scripts or UNIX System Services shell scripts and JCLs.

   The scripts are included in the SAQTSAMP data set, which is delivered with IBM Db2 Analytics Accelerator for z/OS.

   - AQTSCI02
   - AQTSJI02
   - AQTSSCHK
   - AQTSSCPY
   - Sample XML input strings depending on the stored procedures that you want to test. For example, AQTSXTCO is a sample XML input string for the SYSPROC.ACCEL_TEST_CONNECTION stored procedure. The SAQTSAMP data set contains such sample XML input strings for most IBM Db2 Analytics Accelerator for z/OS stored procedures.

   For a description, follow the **Related reference** link at the end.
3. Customize the following scripts according to your needs:

   - AQTSCI02
   - AQTSJI02

- AQTSCHK
- Members containing sample XML input strings, such as AQTSXTCO.

4. Submit AQTSJI02.

   AQTSJI02 uses AQTSSCPY to copy AQTSCI02 and various XML files to the `/tmp/ivp` directory in the z/OS UNIX file system, starts the command line processor with the copied files as input, and finally runs AQTSSCHK to verify the output.

Related reference

Members of SAQTSAMP
The SAQTSAMP data set contains various samples for installing, configuring or running the product's stored procedures. The following table lists all these members and provides brief descriptions of their functions.

Using the DSN Db2 command processor for TSO

You can also start the batch job from the DSN Db2 command processor for Time Sharing Option (TSO). With the RUN subcommand of the processor, you can start an application that contains SQL CALL statements invoking the stored procedures.

Before you begin

Make sure that the following conditions apply:

- The Db2 libraries for IBM Db2 Analytics Accelerator for z/OS have been installed.
- The IBM Db2 Analytics Accelerator for z/OS stored procedures have been installed.
- The z/OS XL C/C++ compiler has been installed.
- The accelerator has been successfully added to the Db2 subsystem from your administration client.
- The Db2 for z/OS JCL procedure DSNHC is available and is properly configured.

About this task

To create a suitable batch job with minimum effort, you can copy and customize the sample JCL members, which are included in the SAQTSAMP data set. The SAQTSAMP data set is delivered as part of the product.

Procedure

1. Copy the following job members of the SAQTSAMP data set:
   - AQTSJI03
   - AQTSCALL
2. Customize the copies as needed, that is, replace the !SAQTSAMP! and !SAMPDBRM! placeholders with appropriate data-set names.

   AQTSCALL is a sample application program in the C programming language. A single invocation of AQTSCALL calls a single stored procedure. You specify the procedure to be called by providing one of the input parameters in Table 11 on page 124.

Table 11. Input parameters for AQTSCALL

Input parameter	Called stored procedure
ADDTABLES	SYSPROC.ACCEL_ADD_TABLES
ALTERTABLES	SYSPROC.ACCEL_ALTER_TABLES
ARCHIVETABLES	SYSPROC.ACCEL_ARCHIVE_TABLES
CTRLACCEL	SYSPROC.ACCEL_CONTROL_ACCELERATOR

Table 11. Input parameters for AQTSCALL (continued)

Input parameter	Called stored procedure
LOADTABLES	SYSPROC.ACCEL_LOAD_TABLES
REMOVETABLES	SYSPROC.ACCEL_REMOVE_TABLES
REPLTABLES	SYSPROC.ACCEL_SET_TABLES_REPLICATION
SETTABLES	SYSPROC.ACCEL_SET_TABLES_ACCELERATION
TESTCON	SYSPROC.ACCEL_TEST_CONNECTION

The AQTSCALL program reads the input for the stored procedures from the DD names AQTP1, AQTP2, AQTP3 and AQTP4 for up to four input parameters. The DD names are defined in the AQTSJI03 job, which calls the sample application. Depending on the SQLSTATE that is set by the stored procedures, the AQTSJI03 job ends with one of the following return codes:

Table 12. Return codes of AQTSJI03

Return code	Description
0	Success
4	Warning
8	Error
16	Call failed or information could not be returned

AQTSJI03 is a sample JCL that compiles and links the C program AQTSCALL and that invokes it as a subcommand of DSN. It contains a sample invocation for each stored procedure. Instructions on how to customize AQTSJI03 are included in the header.

3. Submit AQTSJI03.

What to do next

For future invocations of AQTSCALL, remove the compile, link and binds steps from the AQTSJI03 job before submitting it the next time. If, at some point, you really need to recompile the AQTSCALL program, proceed as follows:

1. Remove the data sets that were created by AQTSJI03.

   Remember: During customization, you replaced the placeholders !SAQTSAMP! and !SAMPDBRM! with actual data-set names.

2. Resubmit the AQTSJI03 job.

Related reference

Members of SAQTSAMP
The SAQTSAMP data set contains various samples for installing, configuring or running the product's stored procedures. The following table lists all these members and provides brief descriptions of their functions.

Chapter 2. Error handling

The SQL CALL statement invoking a stored procedure returns an *sqlstate* field value, an *sqlcode* field value, and diagnostic information (*sqlerrmc*) in the SQL communication area (SQLCA).

You find error information in the MESSAGE output parameter only. It cannot be deduced from the *sqlstate* or the *sqlcode*. The *sqlcode* value is always +466 and the *sqlstate* value is always `0100C` because result sets are returned by the procedures unless the procedure ends prematurely.

Table 13. Result set characteristics by stored procedure

Name	Cursor name of result set	Result set columns and data types
SYSPROC.ACCEL_ ADD_ACCELERATOR2	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ ALTER_TABLES	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ ADD_TABLES	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ CONTROL_ACCELERATOR **Note:** This stored procedure returns a result set if you specify the `<getTraceData>` element in the XML input value of the *command* parameter to obtain trace information about an accelerator.	ACCELERATOR_TRACE_ CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ GET_QUERIES	QUERY_INFO_CURSOR	**SEQID** INTEGER **QUERYINFO** VARCHAR(32698)
SYSPROC.ACCEL_ GET_QUERIES2	QUERY_INFO_CURSOR	**SEQID** INTEGER **QUERYINFO** VARCHAR(32698)
SYSPROC.ACCEL_ GET_QUERY_DETAILS	QUERY_INFO_CURSOR	**SEQID** INTEGER **QUERYINFO** VARCHAR(32698)

Table 13. Result set characteristics by stored procedure (continued)

Name	Cursor name of result set	Result set columns and data types
SYSPROC.ACCEL_ GET_QUERY_DETAILS2	1. Cusor name of first result set: QUERY_INFO_ CURSOR 2. Cursor name of second result set: QUERY_PLAN_ CURSOR 3. Cursor name of third result set: QUERY_PLANTEXT_ CURSOR 4. Cursor name of fourth result set: SP_TRACE_CURSOR	1. First result set: **SEQID** INTEGER **QUERYINFO** VARCHAR(32698) 2. Second result set: **SEQID** INTEGER **PLAN** VARCHAR(32698) 3. Third result set: **SEQID** INTEGER **PLAN** VARCHAR(32698) 4. Fourth result set: **SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ GET_QUERY_EXPLAIN	QUERY_INFO_CURSOR	**SEQID** INTEGER **QUERYINFO** VARCHAR(32698)
SYSPROC.ACCEL_ GET_TABLES_DETAILS	1. Cusor name of first result set: TABLES_DETAILS_ CURSOR 2. Cursor name of second result set: SP_TRACE_CURSOR	1. First result set: **SEQID** INTEGER **TABLES_DETAILS** VARCHAR(32698) 2. Second result set: **SEQID** INTEGER **TRACEDATA** VARBINARY(32698)

Table 13. Result set characteristics by stored procedure (continued)		
Name	**Cursor name of result set**	**Result set columns and data types**
SYSPROC.ACCEL_ GET_TABLES_INFO **Note:** This stored procedure returns at least two result sets when run successfully (without errors).	1. Cusor name of first result set: TABLES_INFO_ SPEC_CURSOR 2. Cursor name of second result set: TABLES_INFO_ STATES_CURSOR 3. Cursor name of third result set: SP_TRACE_CURSOR	1. First result set: **SEQID** INTEGER **TABLE_SPECIFICATION** VARCHAR(32698) 2. Second result set: **SEQID** INTEGER **TABLES_STATES** VARCHAR(32698) 3. Third result set: **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ LOAD_TABLES	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ REMOVE_ACCELERATOR	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ REMOVE_TABLES	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ SET_TABLES_ ACCELERATION	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ SET_TABLES_ REPLICATION	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ TEST_CONNECTION	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)
SYSPROC.ACCEL_ UPDATE_CREDENTIALS	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)

Table 13. Result set characteristics by stored procedure (continued)

Name	Cursor name of result set	Result set columns and data types
SYSPROC.ACCEL_ UPDATE_SOFTWARE2	SP_TRACE_CURSOR	**SEQID** INTEGER **TRACEDATA** VARBINARY(32698)

Related information

Description of SQLCA fields

Chapter 3. What to do after changing a table or table space in Db2 for z/OS

In a number of cases, an ALTER TABLE or ALTER TABLESPACE statement in Db2 for z/OS does not involve maintenance work on your accelerator-shadow tables. There are cases, however, in which it does. See which steps you might have to take.

NOT Programming Interface information

ALTER TABLE

- If a column was added, changed, or deleted by an ALTER TABLE operation on a Db2 table, queries against this table are not accelerated if they reference the new, changed, or deleted column (reason code 14). Therefore, you must complete the following steps after an ALTER TABLE operation:
  1. Remove the corresponding accelerator-shadow table from all accelerators on which it is defined.

     In most cases, a REORG operation is not required before the table can be redefined on the accelerator. However, a REORG is required in the following cases, although this seems to be just a recommended action, as the advisory *REORG-pending* (AREO) state suggests:

     – The ALTER TABLE operation added a column.

     – The ALTER TABLE operation changed column data types.

     – The ALTER TABLE operation dropped columns in *Db2 11 for z/OS*.
  2. Redefine the table on the accelerators from which you previously removed it.
  3. Load the redefined table on all accelerators.
  4. Enable query acceleration for the redefined table on all accelerators.

ALTER TABLESPACE

If a table space was changed in Db2 for z/OS (ALTER TABLESPACE), and the tables of that table space also exist on an accelerator, actions are required under the following conditions:

- If the maximum number of partitions has changed (MAXPARTITIONS), you only have to reload the new or changed partitions. It is not required to reload existing partitions that have not changed.
- If a table space was changed from *segmented* to *partitioned by growth*, you must remove the accelerator-shadow tables of that table space from all accelerators, then redefine and re-prepare these for accelerated queries. Follow steps “1” on page 131 through “4” on page 131 to do that.

LOAD from DD DUMMY input data set

If you have emptied a table space or a table-space partition by overwriting its content through a **`LOAD REPLACE`** operation with **`DD DUMMY`** as the SYSREC data set, and the table has been enabled for incremental updates using IBM Integrated Synchronization, you need not reload the table or partition on the accelerator. The content changes are mirrored automatically in the corresponding accelerator-shadow table.

REORG DISCARD

If you have emptied a partition of a *range-partitioned* table through a **`REORG DISCARD`** operation without affecting any other partition of this table, and if the table has been enabled for incremental updates using IBM Integrated Synchronization, you need not reload the partition on the accelerator. The content changes are mirrored automatically in the corresponding accelerator-shadow table.

End of NOT Programming Interface information

Chapter 4. Messages

The messages that might appear when you run IBM Db2 Analytics Accelerator for z/OS stored procedures are listed here in numerical order.

AQT10000I **The operation was completed successfully.**

Explanation:
Success message for the XML MESSAGE output parameter of each stored procedure.

AQT10001I **Product version: *version*, Build label: *buildLabel*, Build timestamp: *buildTimeStamp*, Build platform: *buildPlatform*, Support level: *supportLevel*, DRDA protocol level: *communicationProtocolLevel***

Explanation

version
Version of the IBM Db2 Analytics Accelerator for z/OS stored procedure

buildLabel
Build label of the stored procedure

buildTimeStamp
Time of the build

buildPlatform
Platform on which the procedures were built

supportLevel
The support level of the stored procedure

communicationProtocolLevel
DRDA protocol level of the stored procedure

Problem determination:
To include version information about IBM Db2 Analytics Accelerator for z/OS in the MESSAGE output parameter as shown in the message text, you can set the version attribute in the XML MESSAGE input parameter for any IBM Db2 Analytics Accelerator for z/OS stored procedure. To do so, set the versionOnly attribute to the value true.

AQT10002W **Warnings were issued during this operation, but a corresponding output message could not be generated. The highest severity code was: *highestSeverity* Additional diagnostic data: *additionalDiagnostic***

Explanation

highestSeverity
The highest internal severity code that was issued during the stored-procedure run

additionalDiagnostic
Additional diagnostic data

Problem determination:
This message is issued when the generation of an output message failed although the stored procedure could complete the requested operation. This indicates that warnings, but not errors, have been issued during the operation.

User action:
Contact IBM support.

AQT10003E **An error caused the operation to fail. A corresponding output message could not be generated. The highest severity code was: *highestSeverity* Additional diagnostic data: *additionalDiagnostic***

Explanation

highestSeverity
The highest internal severity code that was issued during the run of the stored procedure

additionalDiagnostic
Additional diagnostic data

Problem determination:
This message is issued when the requested operation could not be executed and when the generation of a corresponding output message failed.

User action:
Contact IBM support.

AQT10004W **Several messages could not be included in the MESSAGE output. The number of skipped messages is *numberOfSkippedMessages*.**

Explanation

numberOfSkippedMessages
Number of messages that could not be written to the MESSAGE output parameter

Problem determination:
This message is issued as part of the MESSAGE output. It is displayed if some of the messages could not be included in the output object because the size limit of 64 KB would have been exceeded.

User action:

- Check the messages that are actually included in the output object. If these messages indicate a situation that requires further analysis, rerun the stored procedure with a trace level of DEBUG. This will cause all messages to be written to a trace file.
- Moreover, you can retrieve the full set of messages from the last result set of the procedure (identified by MESSAGES_CURSOR).

AQT10005W **A character in the MESSAGE output parameter was substituted due to an XML 1.0 character set violation.**

Problem determination:
This message is issued as part of the MESSAGE output to indicate the replacement of an invalid XML 1.0 character in the output.

User action:
Check the warnings that are included in the output. Substituted bytes are replaced with an asterisk (*). If the warnings indicate a situation that requires further analysis, rerun the stored procedure with a trace level of DEBUG. This causes all warnings to be written to a trace file.

AQT10050E **An internal error occurred on the *acceleratorName* accelerator: *diagnosticMessage***

Explanation

acceleratorName
Name of the accelerator

diagnosticMessage
Diagnostic information returned by the accelerator

Problem determination:
This error occurs when an unhandled error was detected on the accelerator. Additional information is returned in the diagnostic message.

User action:
If this message was returned by the SYSPROC.ACCEL_LOAD_TABLES stored procedure, you might receive the message 04E-00E4007 abend LOC=DSNUT101.DSNUGSDA for those Db2 UNLOAD utilities processes or threads that have already unloaded data to the accelerator. You can ignore this message and delete the corresponding dump files.

AQT10052W **The *acceleratorName* rejected the cancelation request for the *taskId* task. Reason: *cancelationErrorReason*.**

Explanation

acceleratorName
Name of the accelerator

taskId
ID of the task that could not be canceled

cancelationErrorReason
Diagnostic information from the accelerator (internal message that explains why the cancelation failed)

Problem determination:
This warning occurs when it is not possible to cancel a task. The task might no longer exist because it finished recently or the accelerator might have refused the cancelation. In either case, the accelerator remains unchanged and operations continue.

User action:
Check the list of running tasks and make sure that a task with the reported ID really exists.

AQT10053W **A task *taskId* of type *taskType* is running on the accelerator *acceleratorName* using table *tableSchema.tableName.* The operation ACCEL_REMOVE_TABLES is completed, but the identified concurrent task may fail due to that.**

Explanation

acceleratorName
Name of the accelerator that was accessed

tableSchema
Schema name of the table that was removed

tableName
Name of the table that was removed

taskId
ID of the concurrent accelerator task that is using the table

taskType
Type of the concurrent accelerator task that is using the table

Problem determination:
This warning occurs when you drop an accelerator table that is still used by a process or task. The table will be dropped, but the task or process might fail thereafter.

User action:
No action required. You can ignore error messages that are issued by related processes running in parallel.

AQT10055E **One or more tables prevent the removal of the *acceleratorName* accelerator because they have not been removed or are still listed in the SYSACCEL.SYSACCELERATEDTABL**

ES catalog table of your Db2 subsystem (database). In this particular case, the tables were found in or on the *existOnSide*.

Explanation

acceleratorName
Name of the accelerator

existOnSide
Table location, that is, ACCELERATOR or DATABASE

Problem determination:
Remaining tables or orphaned catalog entries prevent the removal of an accelerator.

User action

Follow these steps:

1. Find the remaining tables or orphaned catalog entries by running the SYSPROC.ACCEL_GET_TABLES_INFO stored procedure.
2. Run the SYSPROC.ACCEL_REMOVE_TABLES stored procedure or the corresponding function in your administration client to remove these tables and entries.
3. Remove the accelerator by running the SYSPROC.ACCEL_REMOVE_ACCELERATOR stored procedure or the corresponding function in your administration client.

AQT10061W **The maximum size of compressed trace data that is specified in the *traceProfileName* trace profile for the *acceleratorName* accelerator has been changed from *userSpecifiedSize* MB to *adjustedSize* MB.**

Explanation

acceleratorName
Name of the accelerator that the trace information is written to

traceProfileName
Name of the trace profile

userSpecifiedSize
Maximum size of the compressed trace file as specified by the user in the trace profile

adjustedSize
Adjusted size of the compressed trace file

Problem determination:
This warning occurs when the allowed maximum size of compressed trace data, as specified in the trace profile, has been modified because it was outside the allowed limits, or because no size limit was specified in the profile. The maximum size in the trace profile is the total data size in MB for accelerator traces that can be returned by the **`getTraceData`** command of the the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

User action:
Check the file size that is displayed by the adjustedSize parameter in the warning. The next time that you specify a trace profile, choose a bigger size than this.

AQT10062W **The accelerator was added succesfully. Since it had been added to the same Db2 subsystem before, obsolete data from this subsystem (*dbmsSystemName*) was found on the accelerator. This obsolete data has been deleted during the process.**

Explanation

dbmsSystemName
Location name of the Db2 subsystem that the accelerator is added to

Problem determination:
An accelerator can be added only once to a Db2 subsystem. To add it again, it must have been removed before. However, there are situations in which an accelerator was not properly removed, but the authentication token and other information that is required for using the accelerator is missing from the Db2 catalog tables. When this information has disappeared from the catalog tables, the accelerator can be added again. However, it probably contains data that was unloaded from the Db2 subsystem at an earlier time. Since the original authenication token is lost, the data cannot be accessed anymore. Adding the accelerator once more to the Db2 subsystem removes the orphaned data and thus allows you to start over from a clean setup. This warning just indicates that obsolete data from the Db2 subsystem was found on the accelerator and has been deleted.

User action:
No action required. Add and load tables to the accelerator as needed.

AQT10064E **The stored procedure cannot connect to the *acceleratorName* accelerator because the authentication failed. Diagnostic information: *reasonText*.**

Explanation

acceleratorName
Name of the accelerator

reasonText
Additional diagnostic information about the authentication failure

Problem determination:
This exception occurs when an authentication problem prevents the DRDA component running inside a stored procedure from connecting to the accelerator. This can occur in connection with any IBM Db2 Analytics Accelerator for z/OS stored procedure.

User action

- Check whether the catalog tables in the Db2 communication database were accidentally modified or replaced with a backup after adding the accelerator. If this is the case, restore the catalog tables to the correct state.
- If this exception occurred very shortly after the authentication token was renewed by the SYSPROC.ACCEL_UPDATE_CREDENTIALS stored procedure, or by the equivalent function in your administration client, rerun the stored procedure or repeat your action from the administration client.

AQT10065W **A trace profile with the name *traceProfileName* already exists on the *acceleratorName* accelerator. The existing profile was not changed.**

Explanation

acceleratorName
Name of the accelerator

traceProfileName
Name of the trace profile that already exists in the catalog

Problem determination:
This warning occurs when a trace profile cannot be created or changed on an accelerator because a trace profile with the same name already exists.

User action:
Change the name of the new trace profile or remove the existing trace profile.

AQT10066W **A trace profile with the name *traceProfileName* does not exist on the *acceleratorName* accelerator, and therefore could not be removed.**

Explanation

acceleratorName
Name of the accelerator

traceProfileName
Name of the trace profile that does not exist in the catalog

Problem determination:
This warning occurs when a trace profile cannot be removed because it is not listed in the catalog of the accelerator.

AQT10067W **The trace profile *traceProfileName* cannot be removed from the *acceleratorName* accelerator because it is a predefined (system) profile rather than a user-defined profile.**

Explanation

acceleratorName
Name of the accelerator

traceProfileName
Name of the trace profile that cannot be removed

Problem determination:
This warning is issued when you try to remove a predefined trace profile from an accelerator. Predefined trace profiles cannot be removed.

AQT10068W **The previously active trace profile *removedTraceProfileName* has been removed from the *acceleratorName* accelerator. For that reason, the *activeTraceProfileName* profile is used as the active profile.**

Explanation

acceleratorName
Name of the accelerator

removedTraceProfileName
Name of the previously active trace profile, which has been removed

activeTraceProfileName
Name of the new active trace profile

Problem determination:
This warning is issued when the accelerator resorts to the default trace profile because the previously active profile has been removed.

AQT10069E **The trace profile *traceProfileName* could not be created on the *acceleratorName* accelerator because an incorrect trace level *levelName* was specified.**

Explanation

acceleratorName
Name of the accelerator

traceProfileName
Name of the trace profile with an incorrect trace level

levelName
The incorrect trace level

Problem determination:
This error occurs when you specify an incorrect trace level for the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure. You might have misspelled the trace level or accidentally specified a level that does not exist.

User action:
Correct the trace level in the COMMAND input parameter for the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

AQT10070E **The trace profile *traceProfileName* could not be created on the *acceleratorName* accelerator because an incorrect trace component name *componentName* was specified.**

Explanation

acceleratorName
Name of the accelerator

traceProfileName
Name of the trace profile in which the unknown component is defined

componentName
Name of the wrong or unknown trace component

Problem determination:
This error occurs when you specify an incorrect trace component for the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure. You might have misspelled the component name or accidentally specified a component that does not exist.

User action:
Correct the trace component name in the COMMAND input parameter for the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

AQT10071E **The accelerated table *tableSchemaName.tableName* for the accelerator *acceleratorName* is in *currentTableState* state, which does not allow the attempted operation. Diagnostic information: operation: *diagnosticHintOperation***

Explanation

tableSchemaName
Schema name of the table that is out of sync

tableName
Name of the table that is out of sync

acceleratorName
Name of the accelerator

currentTableState
Current table state, which prevents the execution of the operation

diagnosticHintOperation
Diagnostic information about the failed operation

Problem determination:
This error occurs when the state of an accelerator table prohibits a certain type of operation. For the SYSPROC.ACCEL_SET_TABLES_ACCELERATION(ON) operation, the allowed states are *Loaded* and *UpdateInProgress*.

User action:
Make sure that all tables involved are in a state that permits the operation. Then retry running the procedure. To run SET_TABLES_ACCELERATION(ON) successfully, call SYSPROC.ACCEL_LOAD_TABLES before. You cannot run SYSPROC.ACCEL_SET_TABLES_ACCELERATION(ON) successfully as long as SYSPROC.ACCEL_LOAD_TABLES is executing an initial load. You must wait for the completion of the initial load.

AQT10072E **Query acceleration cannot be enabled for the *tableSchemaName.tableName* table on the *acceleratorName* accelerator because it is already enabled on another accelerator named *dupAcceleratorName*.**

Explanation

tableSchemaName
Schema name of the table that is enabled on another accelerator

tableName
Name of the table that is enabled on another accelerator

acceleratorName
The specified accelerator (accelerator on which the operation could not be executed)

dupAcceleratorName
The other accelerator that already contains an enabled copy of the table

Problem determination:
This error occurs when you try to enable query acceleration for a table that exists on multiple accelerators and has already been enabled on one of these accelerators.

User action:
Disable query acceleration for the table on the accelerator that you do not want to use from your administration client or by running the SYSPROC.ACCEL_SET_TABLES_ACCELERATION stored procedure with the value OFF. Then enable the table on the proper accelerator.

AQT10073W **The accelerator *acceleratorName* was added to a Db2 subsystem, but the system date or time of the subsystem differs from the system date or time of the accelerator. The system date and time of the *referenceTimeSubsystemName* subsystem serves as a reference for the accelerator. A time difference imposes limitations on the use of certain SQL statements.**

Explanation

referenceTimeSubsystemName
Name of the Db2 subsystem that sets the reference time

acceleratorName
Name of the accelerator

Problem determination:
The system time and date of an accelerator must be in sync with the system time and date of the Db2 subsystem to which it is attached. Otherwise queries that include time-related SQL expressions, such CURRENT_DATA, CURRENT_TIME, and CURRENT_TIMESTAMP, cannot be processed correctly. If multiple database subsystems are attached to the same accelerator, all these subsystems must use the same date and time settings. A violation of this rule might lead to unexpected query results.

User action:
The easiest solution is to avoid queries that contain SQL expressions such as CURRENT_DATA, CURRENT_TIME, or CURRENT_TIMESTAMP. If this is not possible, remove the accelerator from all subsystems with different time settings. Then align the date and time settings of the Db2 subsystems with those of the accelerator. Finally, re-add the accelerator.

AQT10074W **The stored procedure and the accelerator use different versions of the DRDA protocol (*releaseLevelProcedure* for the stored procedure and *releaseLevelAccelerator* for the accelerator).**

Explanation

releaseLevelProcedure
The version of the DRDA protocol that is supported by the stored procedure

releaseLevelAccelerator
The version of the DRDA protocol that is supported by the accelerator

Problem determination:
This warning occurs when the protocol versions of a stored procedure and an accelerator are not compatible.

User action:
Make sure that your accelerator is updated to the same or a higher PTF level than the stored procedures that you are using. When you upgrade your system, you must transfer and activate a new software version on the accelerator before you can activate the corresponding load libraries for the stored procedures. When you downgrade the accelerator by activating an older software version, you must also switch to a corresponding older version of the stored procedures.

AQT10075E **The *failingTest* test failed.**

Explanation

failingTest
The name of the test that failed. Possible names:

- *ping*
- *connect*
- *mtupath*
- *networkSpeed*
- *version*

Problem determination:
This error occurs when a test fails that was started by the SYSPROC.ACCEL_TEST_CONNECTION stored procedure.

User action:
See the output (value of the diagnostic_output parameter) of the SYSPROC.ACCEL_TEST_CONNECTION stored procedure for details.

AQT10076W **The *tableSchema.tableName* table on accelerator *acceleratorName* cannot be found in the catalog**

of the *unknownOnSide* or has been altered, such that the table definitions on the accelerator are not longer in synch with the definitions in the database.

Explanation

tableSchema
Schema that the out-of-sync table belongs to

tableName
Name of the table that is out of sync

acceleratorName
Name of the accelerator

unknownOnSide
Catalog that does not list a table with the specified name (ACCELERATOR or DATABASE).

Problem determination:
The Db2 catalog table SYSACCEL.SYSACCELERATEDTABLES and the information in the accelerator catalog are out of sync with regard to the specified table. This might have been caused by a restore operation or a modification of the original Db2 table.

User action

- If the warning was issued while the SYSPROC.ACCEL_REMOVE_TABLE stored procedure was running, do not take any action. The inconsistency has already been removed.
- If the warning was issued while the SYSPROC.ACCEL_GET_TABLE_INFO stored procedure was running, check if columns were added to or modified in the original table after it was defined on the accelerator. In this case, remove the table from the accelerator and define it again.
- In all other cases, check the SYSACCEL.SYSACCELERATEDTABLES table for recent modifications or restore operations to determine the cause of the inconsistency. If possible, restore the latest state of the table. If this is not possible, remove the accelerator table (using SYSPROC.ACCEL_REMOVE_TABLES) and redefine it (using SYSPROC.ACCEL_ADD_TABLES).

AQT10077W **The *tableSchema.tableName* table is unknown on the *acceleratorName* accelerator.**

Explanation

tableSchema
Schema name of the table that does not exist

tableName
Name of the table that does not exist

acceleratorName
Name of the accelerator

Problem determination:
A table cannot be removed because it cannot be found in the catalog of the accelerator. This table is ignored so that the processing of the other tables can continue.

User action:
No action necessary.

AQT10078E **The value *value* for the *columnName* column of type *columnDataType* in table *tableSchema.tableName* (partition ID: *partID*) is not supported by the product: *diagnosticText***

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table

columnName
Name of the affected table column

columnDataType
Data type of the column in which the unsupported value was found

value
Value that cannot be processed by the accelerator

partID
Logical ID of the partition in which the table resides

diagnosticText
Optional diagnostic information with details about the error.

Problem determination

A value in a Db2 table is not supported by the product. Displayed in the error message is a short extract of the full value. The first part (before the square brackets) shows the last 10 correct bytes. The square brackets indicate the point of error. The brackets contain the first 10 illegal or unsupported bytes, which makes it easier for you to locate the point of error and analyze its cause. Because the value is not supported, you cannot load a table that contains this value. The error occurs, for example, in the following cases:

- A Db2 string value contains invalid UTF-8 sequences in a CHAR column.
- A Db2 floating point value exceeds the exponent range for IEEE single precision in a REAL column.

User action

- Change the values in the original Db2 tables to close nearby values that will not influence the results too much.
- Alternatively, remove the table that cannot be loaded from the accelerator. You might receive a message similar to `04E-00E4007 abend LOC=DSNUT101.DSNUGSDA` for those Db2 UNLOAD utilities that have already unloaded data to the accelerator. You can ignore this message and delete the corresponding dump files.

AQT10079E **The *tableSchema*.*tableName* in the Db2 subsystem with the location name *locationName* has too many organizing keys. The table can have *maxNumOrganizingKeyColumns* organizing keys at most, but *numOrganizingKeyColumns* keys have been specified. The limit has been exceeded.**

Explanation

locationName
Location name identifying the Db2 subsystem to which the table belongs

tableSchema
Schema name of the table that uses the organizing keys

tableName
Name of the table that uses the organizing keys

numOrganizingKeyColumns
Number of organizing keys that were specified for the table

maxNumOrganizingKeyColumns
Allowed maximum number of organizing keys

Problem determination:
The table has too many organizing keys.

User action:
Specify fewer organizing keys so that the limit is not exceeded.

AQT10080E **The distribution key for table *tableSchema*.*tableName* in the database system with the location name *locationName* can have *maxNumDistributionKeyColumns* columns at most. However, *numDistributionKeyColumns* columns have been specified. The limit has been exceeded.**

Explanation

locationName
Location name identifying the Db2 subsystem to which the table belongs

tableSchema
Schema name of the table that uses the distribution key

tableName
Name of the table that uses the distribution key

numDistributionKeyColumns
Number of specified distribution key columns

maxNumDistributionKeyColumns
Allowed maximum number of distribution key columns

Problem determination:
The distribution key has too many columns.

User action:
Reduce the number of columns in the distribution key so that the limit is not exceeded.

AQT10081E **A plan with an ID of *planid* cannot be found on the accelerator.**

Explanation

planid
The plan ID that was specified for the stored procedure could not be found on the accelerator

Problem determination:
The specified plan ID is unknown on the accelerator.

User action:
Rerun the SYSPROC.ACCEL_GET_QUERIES stored procedure to obtain the most recent plan ID.

AQT10082E **The schema of the *tableSchema*.*tableName* table was changed after adding the table to this accelerator: *details*.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

details
Exception details

Problem determination:
This exception occurs when a column definition was changed or a column was added or removed from a table by an ALTER TABLE statement after adding the table to an accelerator.

User action

1. Remove the table from the accelerator.
2. Re-add the table to the accelerator.
3. Rerun the stored procedure that returned the error.

AQT10083W **The Netezza® debug configuration failed when executing the commands *nzCommands*. The error output for the commands was *nzOutput*.**

Explanation

nzCommands
Shell commands that were called on the Netezza host

nzOutput
Standard error (stderr) output of the shell commands

Problem determination:
This warning is issued when an accelerator trace profile could not set the specified debug level on the Netezza host.

User action

1. Use only the predefined trace profiles or profiles that were provided by IBM support.
2. If the problem also occurs in connection with predefined or IBM-supplied trace profiles, report the problem to IBM support.

AQT10084W **The *acceleratorName* accelerator was removed from the Db2 subsystem that sets the reference time for accelerators. From now on, the accelerator uses the time of the *newReferenceTimeSubsystemName* subsystem as the reference time. This might impose restrictions on the use of certain SQL statements.**

Explanation

newReferenceTimeSubsystemName
Name of the Db2 subsystem that now provides the reference time for the accelerators

acceleratorName
Name of the accelerator

Problem determination:
The system date and time of an accelerator must be in sync with the system date and time of the database subsystems to which it is attached. Otherwise queries that contain expressions like CURRENT_DATA, CURRENT_TIME, or CURRENT_TIMESTAMP cannot be processed correctly. If an accelerator is attached to multiple subsystems, the subsystem to which it was added first sets the reference time. Ideally, the system dates and times of all subsystems that use an accelerator are the same. If an accelerator is attached to subsystems with different date and time settings, and the accelerator is removed from the subsystem that sets the reference time, one of the other subsystems takes over, that is, sets the reference time for the accelerator. This might lead to different query results if your queries contain any of the SQL expressions mentioned before.

User action:
The easiest solution is to avoid queries that use SQL expressions such as CURRENT_DATA, CURRENT_TIME, or CURRENT_TIMESTAMP. If this is not possible, remove the accelerator from subsystems with different date and time settings.

AQT10086W **Values in the *columnName* column of the *tableSchema.tableName* table were cut off when the table was loaded on the accelerator. *numberOfTruncations* values were truncated.**

Explanation

tableSchema
Schema name of the table that was loaded

tableName
Name of the table that was loaded

columnName
Name of the column in which values were truncated

numberOfTruncations
Number of truncations

Problem determination:
The truncation is due to the fact that in Db2 for z/OS, VARCHAR values consist of up to 32704 characters, and VARGRAPHIC values of up to 16352 characters, whereas their NVARCHAR equivalent on the Netezza host allows a maximum of 16000 characters only. If values were truncated and the column is used in a query, the accelerator will not return the same query results as Db2 for z/OS.

AQT10088W **The *tableSchema.tableName* table was loaded or updated successfully on the accelerator, but the REFRESH_TIME column of the SYSACCEL.SYSACCELERATEDTABLES catalog table could not be updated. Diagnostic information: *diagnosticText***

Explanation

tableSchema
Schema name of the table that was loaded

tableName
Name of the table that was loaded

diagnosticText
Diagnostic information indicating why the REFRESH_TIME column could not be updated.

Problem determination:
The table was successfully loaded, but the timestamp for this event could not be updated in the SYSACCEL.SYSACCELERATEDTABLES catalog table in Db2. The next time you run the SYSPROC.ACCEL_LOAD_TABLES stored procedure on the same table, another attempt is made to update the timestamp.

AQT10089W **The following problem occurred during the migration of the accelerator catalog: *description*.**

Explanation

description
Description of the migration problem

Problem determination:
A problem was encountered during the migration (update) of the accelerator metadata. An auto-correction is in place for problems of this type.

User action:
If you are interested in the corrective actions that were taken by the accelerator, see the information in the problem description.

AQT10091W **Tried to add the *tableSchema*.*tableName* table to the *acceleratorName* accelerator, but the data type of the following columns is not supported: *columnList***

Explanation

acceleratorName
Name of the accelerator

tableSchema
Name of the table schema

tableName
Name of the table

columnList
Comma-separated list of all columns in the current table whose data type is not supported

Problem determination:
This warning occurs if you want to add a table to the accelerator, but the table contains one or more columns whose data type is not supported by IBM Db2 Analytics Accelerator for z/OS. To check the data type compatibility, IBM Db2 Analytics Accelerator for z/OS reads the entries in the SYSIBM.SYSCOLUMNS catalog table of Db2 for z/OS. For example, CHAR FOR BIT DATA and ROWID are not supported. See *Conditions that prevent query routing to an accelerator* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide* for information about supported data types.

AQT10096E **The stored procedure cannot obtain required information from the *acceleratorName* accelerator (reference accelerator). Diagnostics: *diagnostics***

Explanation

acceleratorName
Name of the accelerator

diagnostics
Forwarded error description

Problem determination:
Various causes might lead to this exception. See the diagnostic information for details. It seems that you are trying to save already archived partition data on an accelerator that is different from the accelerator that the data was moved to in the first place. However, to complete this operation, the stored procedure requires information from the target accelerator of the first move (reference accelerator). This information cannot be obtained.

User action:
Make sure that the reference accelerator is online.

AQT10100E **The value NULL was specified for the *parameterName* parameter of the stored procedure, but NULL is not a valid value for this parameter.**

Explanation

parameterName
Name of the parameter for which NULL was used as a value

Problem determination:
This error occurs when SQL NULL has been used as a parameter value although NULL is not allowed. In most error situations of this kind, NULL was used by default because the caller of the stored procedure omitted a value that was mandatory in the given context.

User action:
Specify a valid parameter value.

AQT10101E **The *value* value is not valid for the *parameterName* parameter.**

Explanation

parameterName
Name of the parameter whose value is incorrect

value
The incorrect value

Problem determination:
This error indicates that an incorrect input parameter value was passed to a stored procedure.

User action:
Correct the incorrect input value.

AQT10102E **The value *identifier* is incorrect for an identifier of the *type* type. The error condition is: *condition*. The identifier can take on a maximum length of *maxLength*.**

Explanation

condition
Describes the error. The following error conditions are possible:

TOO_LONG
The *identifier* is too long.

TOO_SHORT
The *identifier* is not long enough.

NOT_EXISTS
An object with the specified name (*identifier* value) does not exist in the Db2 subsystem.

ABSOLUTE_PATH_REQUIRED
The path that was specified as the value of the *identifier* is not an absolute path, but an absolute path is required.

type
Type of the incorrect identifier. The following types are possible:

DATABASE
The problem was caused by the Db2 subsystem name.

ACCELERATOR
The problem was caused by the accelerator name.

LOCATION
The problem was caused by the location name of the accelerator.

ACCELERATEDTABLE
The problem was caused by a table name used on the accelerator.

SCHEMA
The problem was caused by a schema name.

TABLE
The problem was caused by a Db2 table name.

COLUMN
The problem was caused by a column name.

TABLESPACE_QUALIFIER
The problem was caused by the qualifier of a table-space name.

TABLESPACE
The problem was caused by a table-space name.

PATH
The problem was caused by an zFS file path name.

identifier
The incorrect identifier value

maxLength
Allowed maximum length of the identifier

Problem determination:
This error occurs when an input parameter of a stored procedure contains an invalid identifier value. This happens, for example, if the accelerator name is too long. The incorrect value can be a database name, an accelerator name, a schema name, a location name, a table name, a column name, a table-space qualifier, a table-space name or a path in a z/OS UNIX file system (zFS).

User action:
Specify a correct identifier value.

AQT10103E **The value *value* is incorrect for the *name* environment variable, which is of the *type* type. Error condition: *condition*. Additional diagnostic information: *additionalDiagnostic***

Explanation

value
Value of the environment variable

name
Name of the environment variable

type
Type of the incorrect value. The following types can be used:

NUMBER
The expected value of the environment variable is a number.

PATH
The expected value is a file path or directory path in the z/OS UNIX file system (zFS).

condition
Describes the error. The following conditions are possible:

NOT_ALLOWED
The value is forbidden.

INVALID_NUMBER
The value is not a valid number.

ZERO_NOT_ALLOWED
The value must be greater than zero.

ABSOLUTE_PATH_REQUIRED
The specified zFS path is not an absolute path, but an absolute path is expected.

PATH_NOT_ACCESSIBLE
The specified zFS path cannot be found or cannot be read.

additionalDiagnostic
Additional diagnostic information

Problem determination:
This error occurs when an environment variable that is used by a stored procedure contains an invalid value, for example, a string where a number is expected.

User action:
Specify a correct value in the AQTENV data set.
If you use the ISPF editor, make sure to use the `NUM OFF` setting. Otherwise, automatically inserted line numbers are appended to the values of the environment variables.

AQT10105E **The *dataType* data type is used by the *columnName* column in the *schemaName.tableName* table. Columns with this data type cannot be used in distribution keys or as organizing keys.**

Explanation

schemaName
Name of the schema that the table with the column in question belongs to

tableName
Name of the table that the column belongs to

columnName
Name of the column for which an unsupported data type has been defined

dataType
Name of the unsupported data type

Problem determination:
This error occurs when a user has specified a column with an unsupported data type as an organizing key or as a distribution key column.

User action:
Select a different column as an organizing key or distribution key column or remove the key definition from the table specification.

AQT10109E **A table with the name *schemaName.tableName* was specified more than once.**

Explanation

schemaName
Name of the schema that the duplicate table belongs to

tableName
Name of the duplicate table

Problem determination

This error occurs when one and the same table occurs multiple times in an input parameter. Table names must be unique within a set of tables. This affects the following parameters:

- *`table_specification`* (SYSPROC.ACCEL_ADD_TABLES)
- *`table_load_specification`* (SYSPROC.ACCEL_LOAD_TABLES)
- *`table_set`* (SYSPROC.ACCEL_REMOVE_TABLES, SYSPROC.ACCEL_GET_TABLES_INFO, and SYSPROC.ACCEL_SET_TABLES_ACCELERATION)

User action:
Remove the duplicate table name from parameter value and rerun the stored procedure.

AQT10114E **The *tableSchema.tableName* table is of the type *tableType*, which is not a base table type. Only base tables are supported.**

Explanation

tableSchema
Name of the schema that the unsupported table belongs to

tableName
Name of the unsupported table

tableType
Type of the unsupported table

Problem determination:
This error occurs when a table that is referenced in an accelerated table set is not a base table. Only base tables (as opposed to other tables, such as views, auxiliary tables or XML tables) are supported.

User action:
Remove all tables that are not base tables from the table set an rerun the stored procedure.

AQT10115W **The support level of the stored procedure that was used to add the *schemaName*.*tableName* table (*tableSupportLevel*) is higher than the support level of the stored procdure currently processing the table. The stored procedure that is currently in use can only handle tables that were added by a stored procedure at support level *highestSupportLevelAllowed* or lower.**

Explanation

schemaName
Schema name of the table that was created with a newer version

tableName
Name of the table that was created with a newer version

tableSupportLevel
The support level of the table

highestSupportLevelAllowed
The highest support level that the stored procedure can handle

Problem determination:
This error occurs when a table was added to the accelerator with a newer version of the IBM Db2 Analytics Accelerator for z/OS stored procedures. The current version cannot handle these tables because it does not support the same range of data types or other features.

User action:
Check if an installation problem with the currently used stored procedure exists, for example, in the STEPLIB section of the WLM application environment. After correcting the problem, rerun the last operation with the most recent version of the stored procedure. If you want to continue using the currently installed (older) version of the stored procedure, remove the problematic table from the accelerator and re-add it by using the corresponding older version of SYSPROC.ACCEL_ADD_TABLES.

AQT10116E **The output location for the trace file is missing (empty string). Specify a valid file name or data-set name.**

Problem determination:
This error occurs when an empty string was specified as the value of the *outputLocation* XML attribute in the COMMAND input parameter.

User action:
Specify a file name in the *outputLocation* XML attribute of the COMMAND input parameter and rerun the stored procedure.

AQT10118E **The pairing code *givenPassword* is incorrect or has expired.**

Explanation

givenPassword
The specified pairing code

Problem determination:
This error occurs when the accelerator refuses a pairing code because it is incorrect or has expired.

User action:
Try to add the accelerator again with a newly generated (valid) pairing code.

AQT10120E **An accelerator with the name *acceleratorToCreate* could not be added to the Db2 subsystem because the name already exists.**

Explanation

acceleratorToCreate
Name of the accelerator

Problem determination:
This exception occurs when an accelerator cannot be added to a Db2 subsystem because the name of the accelerator already exists in the system tables of the subsystem.

User action:
Specify a different accelerator name.

AQT10121E **The PARTITIONS value for the *schemaName*.*tableName* table, which was specified as the value of the *parameterName* parameter for the SYSPROC.ACCEL_LOAD_TABLES stored procedure, contains an error. The error is at position *offset* of the value string *value*.**

Explanation

parameterName
Name of the parameter for which an incorrect PARTITIONS value was specified (table_load_specification)

schemaName
Name of the schema that the table belongs to

tableName
Name of the table for which an incorrect PARTITIONS value was specified

offset
The position of the error in the value string of the PARTITIONS element

value
The PARTITIONS value in which the error was found

Problem determination:
This error occurs when the PARTITIONS element of the input XML string for the SYSPROC.ACCEL_LOAD_TABLES stored procedure contains incorrect PARTITIONS values for a table. The XML input string is specified as the value of the table_load_specification parameter. The error message gives you information about the position of the error in the PARTITIONS element.

User action:
Correct the PARTITIONS value.

AQT10122E **The *parameterName* parameter of the stored procedure specifies invalid partitions *partitions* for the *schemaName*.*tableName* table.**

Explanation

parameterName
Name of the parameter in which the invalid partition numbers were found

schemaName
Name of the schema that the table belongs to

tableName
Name of the affected (partitioned) table

partitions
The invalid partition numbers that were found

Problem determination:
This exception occurs when the **`table_load_specification`** parameter of the SYSPROC.ACCEL_LOAD_TABLES stored procedure contains a logical partition number for a table partition that does not exist in Db2.

User action:
Remove the invalid partition numbers from the **`table_load_specification`** parameter and rerun the SYSPROC.ACCEL_LOAD_TABLES stored procedure.

AQT10123E **The *schemaName*.*tableName* table uses the *feature* feature, which is not supported by IBM Db2 Analytics Accelerator for z/OS.**

Explanation

schemaName
Name of the schema that the table with the unsupported feature belongs to

tableName
Name of the table with the unsupported feature

feature
Type of the feature that is not supported (descriptive string)

Problem determination:
This error occurs when the table that you want to add to an accelerator uses a Db2 feature that is not supported by IBM Db2 Analytics Accelerator for z/OS.

User action:
Remove the table with the unsupported feature from the table set and rerun the stored procedure.

AQT10124E **Your Db2 subsystem does not permit to write trace information to the standard output stream (STDOUT).**

Problem determination:
This error occurs if your Db2 subsystem does not permit to write trace data to the standard output stream (STDOUT).

User action:
Do not request the output to be directed to STDOUT. Modify the MESSAGE input parameter of the stored procedure accordingly.

AQT10126E **The *acceleratorName* accelerator is a virtual accelerator that cannot be used by this stored procedure.**

Explanation

acceleratorName
The specified accelerator

Problem determination:
This error occurs when a stored procedure calls a virtual accelerator. IBM Db2 Analytics Accelerator for z/OS stored procedures cannot run on or in connection with virtual accelerators. Virtual accelerators are for gathering EXPLAIN information only, that is, information that helps you decide whether it makes sense to accelerate a query.

User action:
Make sure that your stored procedures call regular accelerators only.

AQT10127E **The value of the *parameterName* parameter contains an invalid character at position *positionOfInvalidCharacter*. Diagnostic information: *diagnosticInfo*.**

Explanation

parameterName
Name of the parameter

positionOfInvalidCharacter
Position of the first invalid character in the input parameter value. Note that this is the character position, which is not necessarily identical to the byte position.

diagnosticInfo
The hexadecimal UTF-16 representation of the invalid character

Problem determination:
This error occurs when an input parameter value of a stored procedure contains a character that is not allowed by the XML 1.0 specification. The invalid character makes it impossible to return the wrong value in the XML message output parameter.

User action:
Correct the input value.

AQT10128E **The value *parameterValue* of the *parameterName* parameter contains a forbidden character at position *positionOfInvalidCharacter*.**

Explanation

parameterName
Name of the affected parameter

parameterValue
Parameter value that contains the forbidden characters

positionOfInvalidCharacter
Position of the first forbidden character in the parameter value (accelerator name)

Problem determination:
This error occurs when the accelerator name that is specified as an input value for the SYSPROC.ACCEL_ADD_ACCELERATOR stored procedure contains forbidden characters. Note: Only the first forbidden character is identified in the error message. There might be more.

User action:
Correct the parameter value. Allowed characters are A-Z, 0-9.

AQT10129E **The *componentName* component does not exist or is unknown. Hence events or errors related to this component cannot be traced.**

Explanation

componentName
Specified component name

Problem determination:
This error occurs when the MESSAGE input parameter contains an invalid XML component name.

User action:
Check and correct the component names in the MESSAGE input parameter.

AQT10130E **The accelerator could not be added to the Db2 subsystem because the *acceleratorConflict* accelerator uses the same IP address (*iPConflict*) and port (*portConflict*).**

Explanation

acceleratorConflict
The name of the accelerator that uses an identical IP address and port

iPConflict
The duplicate IP address

portConflict
The duplicate port

Problem determination:
All accelerators of a Db2 subsystem must have unique IP addresses and ports.

User action:
Check whether the existing accelerator with the same IP address and port is the one that you wanted to add. If not, assign a unique IP address and port to the new accelerator.

AQT10131E **The *tableSchema*.*tableName* table resides in the *tablespaceQualifier*.*tablespaceName* table space, which is unavailable or cannot be accessed for the requested operation. The status of this table space in the Db2 catalog table SYSIBM.SYSTABLESPACE (output of the Db2 command DISPLAY DATABASE SPACENAM) is *tablespaceStatus*.**

Explanation

tableSchema
Name of the schema that the table in the unavailable table space belongs to

tableName
Name of the table in the unavailable table space

tablespaceQualifier
: Name of the database containing the unavailable table space

tablespaceName
: Name of the unavailable table space

tablespaceStatus
: Status of the table space that is unavailable to the requested function according to the SYSIBM.SYSTABLESPACE catalog table or the output of the DISPLAY DATABASE SPACENAM command.

Problem determination:
This error occurs when a table is listed in a table set for accelerator processing, but the table-space status of this table does not permit the requested operation.

User action:
Remove the affected tables from the table set and rerun the stored procedure or make the table space available again, for example by running the REORG command to remove a status of AREO.

AQT10132E **An accelerator with the name *acceleratorToCreate* could not be added to the Db2 subsystem because a remote location with the name *locationToCreate* already exists.**

Explanation

acceleratorToCreate
: The specified accelerator name

locationToCreate
: The specified location name

Problem determination:
This exception occurs when an accelerator cannot be defined in the system tables of the Db2 subsystem because a remote location with the same name already exists.

User action:
Specify a different location name for the accelerator.

AQT10133E **The *acceleratedTableSchema.acceleratedTableName* table cannot be added twice to the *acceleratorName* accelerator. An entry already exists in the catalog table SYSACCEL.SYSACCELERATEDTABLES.**

Explanation

acceleratorName
: The specified accelerator

acceleratedTableSchema
: The specified schema

acceleratedTableName
: The specified table name

Problem determination:
This exception occurs when you try to add a table more than once to the same accelerator.

User action:
Consider dropping the table or modify the table set (table_specification parameter) for the SYSPROC.ACCEL_ADD_TABLES stored procedure.

AQT10134E **The *acceleratedTableSchema.acceleratedTableName* table does not exist on the *acceleratorName* accelerator.**

Explanation

acceleratedTableSchema
: Name of the table schema

acceleratedTableName
: Name of the table

acceleratorName
: Name of the accelerator

Problem determination:
This error occurs when a table is specified in an input table set, but not listed as a table for accelerated queries in the SYSACCEL.SYSACCELERATEDTABLES catalog table. A possible reason is that the table has just been defined by a concurrent run of the SYSPROC.ACCEL_ADD_TABLES stored procedure, which has not yet ended. Another possible reason is that the table is a reference table and its corresponding SYSACCEL.SYSACCELERATEDTABLES record is missing when invoking SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES.

User action

If the table is part of a set of tables that is just being added to the accelerator, wait until the table definition process has ended and then rerun the stored procedure that returned the error. If the table is a reference table in error, remove the table as follows:

1. Run SYSPROC.ACCEL_REMOVE_TABLES to remove the table from the accelerator.
2. Submit a DROP TABLE statement to remove its entry from the Db2 catalog.

In all other cases, take one of the following actions:

1. Remove the table from the input table set and rerun the stored procedure.

2. Add the table to the accelerator and rerun the stored procedure.

AQT10135E **The DSN_QUERYINFO_TABLE was not found in the *schemaName* schema.**

Explanation

schemaName
Name of the schema

Problem determination:
This error occurs when the DSN_QUERYINFO_TABLE, which is needed to obtain EXPLAIN information, does not belong to the specified schema.

User action:
Make sure that DSN_QUERY_INFO_TABLE exists in the specified schema (see the Db2 documentation on the data-definition language (DDL), the *IBM Db2 Analytics Accelerator for z/OS: User's Guide* or the IBM Documentation for information on how to create this table). Then rerun the stored procedure.

AQT10136E **A query matching the search criteria QUERYNO=*queryno* AND QBLOCKNO=*qblockno* AND EXPLAIN_TIME=*explaintime* was not found in *schemaName*.DSN_QUERYINFO_TABLE.**

Explanation

schemaName
Name of the specified schema. This must be the schema that DSN_QUERYINFO_TABLE belongs to.

queryno
Number that identifies a query. This number must exist in the QUERYNO column of DSN_QUERYINFO_TABLE.

qblockno
Number that identifies the query block of the query. This number must match the QBLOCKNO value of the query in DSN_QUERYINFO_TABLE, (in the same row as the corresponding QUERYNO value).

explaintime
Time needed to obtain EXPLAIN information about a query. This time must match the time value of the query in the EXPLAIN_TIME column of DSN_QUERYINFO_TABLE (in the same row as the corresponding QUERYNO value).

Problem determination:
This error occurs when a matching query could not be found in the DSN_QUERYINFO_TABLE.

User action:
Make sure that the values for queryno, qblockno, and explaintime identify an existing query (all values must the match the values in a single row of DSN_QUERYINFO_TABLE). Then rerun the stored procedure.

AQT10137E **The query with QUERYNO=*queryno* AND QBLOCKNO=*qblockno* AND EXPLAIN_TIME=*explaintime* in *schemaName*.DSN_QUERYINFO_TABLE is not suitable for acceleration because its REASON_CODE is *reasoncodeValue* and its TYPE is *typeValue*.**

Explanation

schemaName
Name of the schema that DSN_QUERYINFO_TABLE belongs to

queryno
Query number (QUERYNO value) in DSN_QUERYINFO_TABLE.

qblockno
QBLOCKNO value of the query in DSN_QUERYINFO_TABLE.

explaintime
EXPLAIN_TIME value of the query in DSN_QUERYINFO_TABLE.

typeValue
TYPE value of the query in DSN_QUERYINFO_TABLE. The expected value is A (with trailing blanks).

reasoncodeValue
REASON_CODE value of the query in the DSN_QUERYINFO_TABLE. The expected value is 0.

Problem determination:
This error occurs when a row identifying a query in DSN_QUERYINFO_TABLE does not relate to a query that is suitable for acceleration. Suitable queries have a REASON_CODE of 0 and a TYPE value of *A*.

User action:
Check the *queryno*, *qblock_no*, and *explain_time* values and make sure that these relate to a query that can be accelerated. Then rerun the stored procedure.

AQT10138E **The *identifier* value for identifier type *type* contains an illegal character at position *positionOfInvalidCharacter*.**

Explanation

type
Type of the incorrect name (that is, schema name, table name, and so on)

identifier
The illegal character in the name

positionOfInvalidCharacter
Position of the first illegal character in the name

Problem determination:
This error occurs when a schema name, table name, or column name contains a character that is not allowed on the accelerator. For example, backslashes or double quotation marks are not allowed.

User action:
Replace the illegal characters with characters that are allowed on the accelerator. To maintain compatibility with inhouse Db2 queries, you can insert a view that maps the changed names to the original names.

AQT10139E **The *tableSchema*.*tableName* table does not contain any column that is suitable for query acceleration.**

Explanation

tableSchema
Name of the schema name that the specified table belongs to

tableName
Name of the specified table

Problem determination:
The specified table cannot be used for accelerated queries because none of its columns fulfills the conditions. For more information, see *Conditions for query acceleration* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.

User action:
Do not add this table to an accelerator.

AQT10141E **The range-partitioned table *tableSchema*.*tableName* cannot be loaded because the partitioning ranges are not known.**

Explanation

tableSchema
Schema name of the table

tableName
Name of the table

Problem determination:
A range-partitioned table cannot be loaded because the boundaries or limits of the partitions are unknown. This happens if ranges have not been defined, because a partitioning index is missing.

User action:
1. Define partitioning ranges for the table by creating the partitioning index. 2. Re-try the SYSPROC.ACCEL_LOAD_TABLES operation.

AQT10142E **An accelerator with the name *acceleratorToCreate* could not be added to the Db2 subsystem because the SYSACCEL.SYSACCELERATEDTABLES catalog table contains entries for tables on an accelerator with the same name.**

Explanation

acceleratorToCreate
Name of the accelerator

Problem determination:
This exception occurs when table definitions exist in the the SYSACCEL.SYSACCELERATEDTABLES catalog table, although the accelerator that these tables belong to has already been deleted. This is most likely the result of an incomplete manual cleanup.

User action:
Remove the orphaned table entries from SYSACCEL.SYSACCELERATEDTABLES or specify a different accelerator name.

AQT10143E **An unsupported version *specifiedLevel* of the IBM Db2 Analytics Accelerator stored procedures was specified in the `compatibilityLevel` element of the message input parameter. The latest supported version is *currentLevel*.**

Explanation

specifiedLevel
Compatibility level that was specified in the message input parameter

currentLevel
Latest supported version

Problem determination:
This exception occurs when an invalid value was specified for the `compatibilityLevel` element of the message input parameter of a stored procedure. For backward compatibility, specify a version not older that the earliest and not newer than the latest supported version.

User action:
Specify a valid value or remove the `compatibilityLevel` element from the message input parameter.

AQT10147E **The *tableSchema*.*tableName* table resides in the *tablespaceQualifier*.*tablespaceName* table space, whose status is**

NOT LOGGED. You cannot enable incremental updates for such a table (the table cannot be enabled for replication).

Explanation

tableSchema
Name of the schema that the affected table belongs to

tableName
Name of the affected table

tablespaceQualifier
Name of the database containing the affected table

tablespaceName
Name of the table space of the affected table

Problem determination:
This error occurs when you want to enable incremental updates for a table, but logging is not enabled for activities that affect the table space of this table. Table changes can only be replicated if they can be written to the log of the table space.

User action:
Enable logging for the table space by using the ALTER TABLESPACE SQL statement.

AQT10148W **The specified partition list was ignored because the value *attributeValue* of the *attributeName* attribute does not permit the use of a partition list. The affected table is *attributeTableSchema.attributeTableName*.**

Explanation

attributeName
Name of the XML attribute

attributeValue
Value of the XML attribute

attributeTableSchema
Schema name of the affected table

attributeTableName
Name of the affected table

Problem determination:
If you specify `forceFullReload='true'` or `detectChanges='DATA'` for a table to be loaded, you cannot use a partition list because the partitions are determined by the accelerator.

User action:
Determine what most suits your needs: to selectively load certain table partitions, to capture and propagate the latest changes to a table, or to fully reload a table for whatever reason. Then adjust the XML input parameter accordingly. That is, change the attribute values, completely remove certain attributes, or remove the partition list from the XML input parameter. Finally, rerun the stored procedure.

AQT10149E **The *tableSchema.tableName* table has an incomplete definition according to its entry in the Db2 catalog table SYSIBM.SYSTABLES.**

Explanation

tableSchema
Name of the schema that the table in the unavailable table space belongs to

tableName
Name of the table in the unavailable table space

Problem determination:
This error occurs when a table is listed in a table set for accelerator processing, but the table definition is not complete. Only tables with complete definitions can be used.

User action

To solve this error, take one of the following actions:

- Remove all tables with incomplete definitions from the table set and rerun the stored procedure.
- Try to understand the reasons for the incomplete definitions by looking at the corresponding entries in the TABLESTATUS column of the SYSIBM.SYSTABLES catalog table. If you can eliminate the causes, remove the incompletely defined tables from the accelerator and re-define these. Finally, rerun the stored procedure.

AQT10151E **The *tableSchema.tableName* table cannot be defined on an accelerator because it uses multiple encoding schemes.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
The specified table contains columns that use encoding schemes other than the encoding scheme of the table space because they were created with an explicit CCSID clause. Queries that reference this table will always have to address these different encoding schemes in the SQL. Such queries do not qualify for acceleration.

AQT10200E **The *operation* operation failed. Error information: *sqlError*. The unsuccessful operation was initiated by the following statement: *sqlStatement*.**

Explanation

operation
A keyword that identifies the failed operation. The following keywords exist:

- CONNECT
- COMMIT
- ROLLBACK
- SAVEPOINT
- SELECT
- DELETE
- MERGE
- INSERT
- UPDATE
- DECLARE_TEMP_TABLE
- OPEN_CURSOR
- PREPARE
- DESCRIBE
- CLOSE_CURSOR
- FETCH
- EXECUTE_IMMEDIATE
- API
- CALL
- ASSOCIATE_LOCATOR
- ALLOCATE_CURSOR

sqlError
SQL error information that describes the error and lists the error tokens

sqlStatement
SQL statement or API call that caused the problem

Problem determination:
This error occurs when an unexpected error from an SQL statement or API call, such as DSNRLI in Db2 for z/OS, is encountered. Details about the error are part of the message, for example, the SQL code and the SQL message.

User action:
Look up the reported SQL code in the documentation of your database management system and try to correct the error.

AQT10201E **Information about the *acceleratorName* accelerator in the IBM Db2 Analytics Accelerator for z/OS catalog tables is inconsistent.**

Explanation

acceleratorName
Name of the affected accelerator

Problem determination:
This error occurs when inconsistent information about an accelerator is found in the IBM Db2 Analytics Accelerator for z/OS catalog tables in the Db2 subsystem. Example: An accelerator is listed in the SYSACCEL.SYSACCELERATORS table, but a corresponding entry cannot be found in the SYSIBM.LOCATIONS table. This error must not be confused with a situation in which the accelerator server cannot be reached.

User action

1. Remove all entries pertaining to the affected accelerator from the IBM Db2 Analytics Accelerator for z/OS catalog tables in your Db2 subsystem.
2. Add the accelerator again by using your administration client or by running the SYSPROC.ACCEL_ADD_ACCELERATOR stored procedure. Make sure to use the same name as before.

AQT10202S **The *acceleratorName* accelerator cannot be contacted over the network. Additional diagnostic data: *additionalDiagnostic***

Explanation

acceleratorName
Name of the affected accelerator

additionalDiagnostic
Additional diagnostic data

Problem determination

This error occurs when a connection to an accelerator could not be established. This might indicate that:

- A network problem exists between your Db2 subsystem and the accelerator.
- The IP addresses or port numbers that are stored in the Db2 subsystem are wrong.
- The accelerator is not running.

User action

1. Check the configuration of the network between your Db2 subsystem and the accelerator.

2. Verify that the entries in the system tables SYSACCEL.SYSACCELERATORS, SYSIBM.LOCATIONS, SYSIBM.IPNAMES, and SYSIBM.USERNAMES are correct.

AQT10205S **An internal processing error occurred in the stored procedure. Diagnostic information: *errorMessage***

Explanation

errorMessage
Additional diagnostic information about the error.

Problem determination:
This error occurs when IBM Db2 Analytics Accelerator for z/OS encounters an unknown internal error.

AQT10206E **The *operation* operation on the *fileName* file, data set, or pipe failed. Diagnostic information: *diagnosticMessage***

Explanation

fileName
Name of the file, data set, or pipe that causes the problem

operation
Operation that was attempted on the file, data set or pipe. The following operations are possible:

ALLOCATE
File, data set, or pipe allocation failure (z/OS only)

DEALLOCATE
File, data set, or pipe deallocation failure (z/OS only)

OPEN_EXISTING
Error while opening an existing file, data set, or pipe

OPEN_NEW
Error while opening a new file, data set, or pipe

OPEN
Error while opening a new or existing file, data set, or pipe

CLOSE
Error while closing a file, data set, or pipe

READ
Error while reading from a file, data set, or pipe

WRITE
Error while writing to a file, data set, or pipe

DELETE
Error while deleting a file, data set, or pipe

RENAME
Error while renaming a file, data set, or pipe

STATS
Error while collecting statistics of a file, data set, or pipe

SEEK
Error while looking for of a certain position in a file, data set, or pipe

SYNC
Error while synchronizing a file, data set, or pipe with a file on disk

UNKNOWN
Unknown error

diagnosticMessage
Additional diagnostic information.

Problem determination:
This error occurs when a file, data set, or pipe cannot be accessed. The cause of the error depends on the type of operation. For example: invalid file names or insufficient access rights.

User action:
Check the diagnostic information.

AQT10207E **The current status of the accelerator does not permit to perform the requested operation. Additional diagnostic information: *diagnosticMessage***

Explanation

diagnosticMessage
Additional diagnostic information

Problem determination:
This error occurs when the status of an accelerator does not permit the execution of the requested operation. For example, you might want to load a table set, but the status of the accelerator is *Maintenance*, which indicates that a software update is running at the same time.

User action:
Check the status of the accelerator by running the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure with the **`getInfo`** parameter. If possible, bring the status of the accelerator to *online* to make it possible to run the operation that was previously denied.

AQT10300E **A string could not be converted from the coded character set identifier (CCSID) *sourceCcsid* to CCSID *targetCcsid*. The z/OS Unicode Conversion Services (UCS) returned error code *returnCode* and reason code**

***reasonCode*. The error class is *error*.**

Explanation

error
Type of the error. The following types of errors exist:

sourceCcsid
CCSID in which the original string is encoded.

targetCcsid
CCSID to which the string should have been converted.

returnCode
Return code received from z/OS Unicode Conversion Services (UCS). If a return code is not available, 0 is returned.

reasonCode
Reason code received from UCS. If a return code is not available, 0 is returned. The availability of a reason code depends on the return code.

Problem determination:
This error occurs when the UCS could not convert a string from one CCSID or code page to another.

User action

1. Look up the return code and the reason code in the UCS documentation.
2. Verify a correct setup of UCS.

AQT10301E **An error occurred in the *interfaceName* service of your database management system (DBMS). The return code is *returnCode*. The reason code is *reasonCode*. Additional diagnostic information: '*additionalInfo*'**

Explanation

interfaceName
Name of the service that returned the error

returnCode
Return code received from the service

reasonCode
Reason code reported by the service

additionalInfo
Additional diagnostic information

Problem determination:
This error occurs when a problem is reported by a service of the database management system.

User action:
If the problem was reported by the DSNTIAR service or the DSNRLI service, you, as a user, cannot analyze and solve the problem. Contact IBM support in such a case. If the problem was reported by the DSNUTILU service, check the diagnostic information that was output by the DSNUTILU stored procedure and try to solve the problem.

AQT10302E **The SYSPROC.ADMIN_INFO_SYSPARM stored procedure could not retrieve the ZParms of the Db2 subsystem. Possible reasons: insufficient privileges of the user who invoked the stored procedure (MONITOR1 is required), IFI errors, or SQL errors in the SYSPROC.ADMIN_INFO_SYSPARM stored procedure. Additional diagnostic information from ADMIN_INFO_SYSPARM: '*additionalInfo***

Explanation

additionalInfo
Additional diagnostic information reported by ADMIN_INFO_SYSPARM that describe the IFI error or SQL error that is encountered.

Problem determination:
This error occurs when the ADMIN_INFO_SYSPARM stored procedure, which retrieves the ZParms of the current Db2 subsystem, fails to return valid results. Possible reasons are internal errors in ADMIN_INFO_SYSPARM (like failures to issue the respective IFI calls) or insufficient privileges of the caller.

User action:
Ask your system administrator to check the privileges of the caller or resolve the errors in the IFI call.

AQT10303E **The '*location*' data set cannot be allocated dynamically by the stored procedure. The allocation failed with return code *returnCode*, error code *errorCode* and reason code *reasonCode*.**

Explanation

location
Data set location (file name)

returnCode
Return code received from the dynalloc function

errorCode
Error code returned by the MVS dynamic allocation functions

reasonCode
Information code returned by the MVS dynamic allocation functions

Problem determination:
This exception occurs when a data set for a stored procedure could not be allocated dynamically on z/OS. In the context of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure, this usually means that there is not enough disk space for a data set big enough to hold the trace data to be saved.

User action:
Look up the reason code in the MVS dynalloc documentation and eliminate the cause of the allocation failure.

AQT10304W **An error occurred during the invalidation of the Db2 Dynamic Statement Cache after loading a table. The return code is *returnCode*. The following diagnostic information was returned by the DSNUTILU utility, which was used to invalidate the cache: *additionalInfo***

Explanation

returnCode
Return code received from DSNUTILU

additionalInfo
Additional diagnostic information from DSNUTILU

Problem determination:
This warning occurs when the loading of tables succeeded, but the invalidation of the dynamic statement cache failed for the processed tables.

User action:
Invalidate the dynamic statement cache manually. The statement that you have to use for this action is shown in the diagnostic information that was returned by the DSNUTILU utility.

AQT10305E **The execution of the system call *SystemCallName* fails with return code *returnCode* and reason code *reasonCode*.**

Explanation

SystemCallName
The name of the system call

returnCode
Return code from the system call

reasonCode
Reason code from the system call

Problem determination:
This failure occurs when IBM Db2 Analytics Accelerator for z/OS calls a z/OS system function and the call ends with an error.

User action

- Ask the z/OS system administrator whether restrictions exist for this system call.
- For information on how to proceed, see the z/OS documentation.

AQT10306E **The image copy *imageCopyDSN* was created by the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure for partition *partition* of the *tableSchema*.*tableName* table. According to information in the SYSIBM.SYSCOPY catalog table, this image copy is no longer the last image copy in the series of image copies for this partition. This incident suggests that the last image copy was not created by the High Performance Storage Saver, but manually by a user. Only storage saver image copies can be used for restoring a partition. The last image copy must therefore be an storage saver image copy.**

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table

tablespaceQualifier
Name of the database containing the table space

tablespaceName
Name of the table space

partition
Number of the physical partition

imageCopyDSN
Name of the image copy that is not the last copy according to SYSIBM.SYSCOPY.

Problem determination:
This exception occurs when a user manually creates an image copy of a partition during the time when this partition is being archived, or in the case of an archiving failure, after the creation of an image copy by the High Performance Storage Saver.

User action

1. Call the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure again. Should this step fail and return the same exception, continue with the following step.
2. If the previous step has failed, use the Db2 RECOVER utility to recover the *imageCopyDSN* image copy and call the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure again.

AQT10307E **The image copy *imageCopyDSN* was created by the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure for partition *partition* of the *tableSchema*.*tableName* table. According to information in SYSIBM.SYSTABLESPACESTATS catalog table, data in this partition was changed after creating the image copy. This is a forbidden operation because an archived partition must not be changed.**

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table

tablespaceQualifier
Name of the database containing the table space

tablespaceName
Name of the table space

partition
Number of the physical partition

imageCopyDSN
Name of the image copy that was created by the High Performance Storage Saver. Changes occurred after the creation of this image copy.

Problem determination:
This exception occurs when data of an archived partition was changed during the archiving process, or, in case of a failure of this process, after creating the image copy.

User action

1. Call the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure again. Should this step fail and return the same exception, continue with the following step.
2. If step “1” on page 156 has failed, complete the appropriate series of steps presented here:
   - If you want to keep the changes:
     a. Create a (manual) image copy of the partition.
     b. Remove all data from the partition.
   - If the data change was not intended or the data is not needed:
     a. Remove all data from the partition.
     b. Run the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure again.

AQT10308E **The caller (user ID) of the *storedProcedureSchemaDotName* stored procedure is not authorized to run the user-defined function *udfSchemaDotName* (missing EXECUTE authority). The value of the *storedProcedureParameterName* input parameter requests an execution of the subfunction *functionXMLElement*. The *udfSchemaDotName* function checks whether a user is authorized to run this subfunction.**

Explanation

udfSchemaDotName
Name of user-defined function that was run to check the authorization for the subfunction

functionXMLElement
Name of the subfunction that the caller of the stored procedure must not execute.

storedProcedureParameterName
Name of the parameter whose value indicates that the use of the indicated subfunction is not authorized.

storedProcedureSchemaDotName
Name of the stored procedure that was invoked

Problem determination:
Running the requested subfunction of the stored procedure requires EXECUTE authorization for a separate user-defined function in Db2 for z/OS.

User action:
Grant EXECUTE authority for the user-defined function to the caller of the stored procedure.

AQT10400E **An internal XML processing error of the type *xmlError* occurred. Information from the XML parser: '*errorMessage*'. Context information from the calling stored procedure: *errorContext*.**

Explanation

xmlError
Possible error types during XML processing

errorMessage
Additional information about the error from the XML parser

errorContext
Additional information about the error context in the stored procedure

Problem determination:
This error message is displayed when an internal XML processing error has occurred. Do not confuse this with the error that occurs when an invalid XML document cannot be parsed because it does not conform to the corresponding XML schema.

AQT10402E **The XML document cannot be validated and parsed correctly because it does not conform to the corresponding XML schema. Error *xercesError* was detected in line *lineNumber*, column *columnNumber* of the following XML document: *partialXmlDocument*.**

Explanation

xercesError
Error message returned by the XML parsing component

lineNumber
Number of the line containing the error

columnNumber
Number of the column containing the error

partialXmlDocument
The first 400® characters of the XML document containing the error

Problem determination:
This error occurs when an XML document cannot be parsed and validated because the document does not conform to the corresponding XML schema. The error message contains information about the position of the error.

User action:
If you use an XML document as input for a stored procedure, use the error information in the message to correct the error.

AQT10403E **The value (XML document) of the *parameterName* parameter is invalid because it contains an invalid attribute combination in the *elementName* XML element. The following attribute combination is not allowed: `'`*`attributeNameA`*`=\"`*`attributeValueA`*`\"'` and `'`*`attributeNameB`*`=\"`*`attributeValueB`*`\"'`.**

Explanation

attributeNameA
Name of the first attribute

attributeValueA
Value of the first attribute

attributeNameB
Name of the second attribute

attributeValueB
Value of the second attribute

elementName
Name of the element containing the invalid attribute combination

parameterName
Name of the parameter containing the invalid XML input document

Problem determination:
This error occurs when an XML document provided as input to a stored procedure cannot be validated because the document contains an XML element with an invalid combination of attributes or attribute values. The error message contains information about the XML element and its attributes.

User action:
Use the error information in the message to correct the error.

AQT10404E **An error occurred while indexes were rebuilt for the *tablespaceQualifier*.*tablespaceName* table space. One or more table-space partitions are in RECOVER PENDING state. This state prevents the completion of the REBUILD INDEX step of the restore operation. The following diagnostic information was returned by the REBUILD INDEX utility: '*additionalInfo*'**

Explanation

tablespaceQualifier
Name of the database containing the table space

tablespaceName
Name of the table space

additionalInfo
Additional diagnostic information returned by the REBUILD INDEX utility

Problem determination:
This error might occur when the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure has failed and has left partitions in RECOVER PENDING state. You might have rerun SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES to fix a previous problem, but maybe have not specified the same number of partitions. This has the effect that the omitted partitions remain in RECOVER PENDING state.

User action

1. Run the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure to list all partitions where the value of the archiveErrorStatus attribute is RestoreIncomplete.
2. Rerun the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure. Specify all the partitions on the list that was returned as a result of completing step 1.
3. If the REBUILD INDEX process fails again, check whether partitions were set to RECOVER PENDING state independently of IBM Db2 Analytics Accelerator operations. If so, take appropriate actions as indicated by the utility messages in the block of diagnostic information.

AQT10405I **The XML parsing configuration for stored procedures is invalid. The data set for the SAQTOSR DD contains invalid data.**

Problem determination:
This warning indicates that stored procedures cannot use the XML parser of z/OS because the configuration of this parser in the Workload Manager (WLM) procedure for stored procedures is invalid. Stored procedures will use an alternative XML parser in a situation like this, but this will result in a higher CPU consumption for XML processing.

User action:
Verify the SAQTOSR DD in the WLM procedure for stored procedures and make sure that the referenced data set corresponds to the same stored procedure version as the code that is referenced in the STEPLIB statement of the WLM procedure.

AQT10407E **The operation cannot be executed on the table *tableSchema.tableName* because the table is in a transition state on the accelerator.**

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table containing the partitions

Problem determination:
This warning indicates that the table on the this accelerator is not in sync with the same table on other accelerators. Therefore, certain operations are not allowed.

User action:
Verify if the transition state was set by a RESTORE operation by looking at the ARCHIVE column of the table. If the ARCHIVE value is 'X', partitions were restored on another accelerator. In this case, the table must be removed from the current accelerator because its data is obsolete. After removing the table, it can be added again and the operation can be executed.

AQT10407E **The operation cannot be executed on the table *tableSchema.tableName* because the table is in a transition state on the accelerator.**

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table containing the partitions

Problem determination:
This warning indicates that the table on the this accelerator is not in sync with the same table on other accelerators. Therefore, certain operations are not allowed.

User action:
Verify if the transition state was set by a RESTORE operation by looking at the ARCHIVE column of the table. If the ARCHIVE value is 'X', partitions were restored on another accelerator. In this case, the table must be removed from the current accelerator because its data is obsolete. After removing the table, it can be added again and the operation can be executed.

AQT10411E **The *acceleratedTableSchema.acceleratedTableName* table cannot be removed from the *acceleratorName* accelerator because it is an accelerator-only table. You can only drop such tables by using a Db2 DROP TABLE statement.**

Explanation

acceleratorName
Name of accelerator on which the affected table resides

acceleratedTableSchema
Name of the table schema

acceleratedTableName
Name of the affected table

Problem determination:
The SYSPROC.ACCEL_REMOVE_TABLES stored procedure does not remove accelerator-only tables.

User action:
Remove the table by using a Db2 DROP TABLE statement rather than the SYSPROC.ACCEL_REMOVE_TABLES stored procedure or the **Remove** function in your administration client.

AQT10412E **The *acceleratedTableSchema.acceleratedTableName* table cannot not be used as input to the stored procedure *procedureName* because it is an accelerator-only table.**

Explanation

procedureName
The name of the stored procedure that does not allow an operation on an accelerator-only table

acceleratedTableSchema
The specified schema

acceleratedTableName
The specified table name

Problem determination:
Accelerator-only-tables are not allowed as input to the stored procedure.

User action:
Remove accelerator-only table from the input tables set of the stored procedure.

AQT10413E **The Db2 real-time statistics could not be externalized. Diagnostic information: *diagnosticInfo*.**

Explanation

diagnosticInfo
Diagnostic information about the cause of the issue.

Problem determination:
The real-time statistics could not be externalized, that is, could not be written from the system memory to the real-time statistics table. This means that the newly gathered statistics do not contribute to the metrics that support processing decisions. If processing continues, the decisions will be based on obsolete and therefore inaccurate data.

User action

1. Analyse the diagnostic information in the message text. Ensure that the user (ID) who runs the stored procedure has the right to invoke the Db2 command `-ACCESS DATABASE (ACCESS DB)`.
2. Check whether the value of the AQT_RTS_EXTERNALIZATION_AUTH_FAILURE environment variable is appropriate.

AQT10414W **The Db2 real-time statistics could not be externalized because the user is not authorized to invoke the Db2 command `-ACCESS DATABASE (ACCESS DB)`. Processing continues because the AQT_RTS_EXTERNALIZATION_AUTH_FAILURE environment variable is set to WARNING.**

Problem determination:
The real-time statistics could not be externalized, that is, could not be written from the system memory to the real-time statistics table. This means that the newly gathered statistics do not contribute to the metrics that support processing decisions. Processing continues because the AQT_RTS_EXTERNALIZATION_AUTH_FAILURE environment variable is set to the value WARNING, but processing decisions will be based on obsolete and therefore inaccurate data. This might cause unwanted side-effects (see the description of AQT_RTS_EXTERNALIZATION_AUTH_FAILURE in the *Stored Procedures Reference*).

User action:
No action is required.

AQT10415W **Db2 column statistics are not available for the *acceleratedTableSchema.acceleratedTableName* table. The absence of appropriate statistics will prevent an automatic creation of organizing keys.**

Explanation

acceleratedTableSchema
Schema name of the affected table

acceleratedTableName
Name of the affected table

Problem determination:

As long as column statistics are missing, organizing keys cannot be automatically created. As a result, incremental updates might be applied inefficiently.

User action:
You can ignore this message unless you want to create organizing keys automatically during the initial load of the table and have enabled the corresponding function on the IBM Db2 Analytics Accelerator Console. In this case, run the Db2 RUNSTATS utility to collect the missing statistics.

AQT10416I **Accelerator *acceleratorName* has started a load with *loadParallelismGoal* parallel threads. The load has the unique identifier *loadTaskId*.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

loadParallelismGoal
Number of parallel threads

Problem determination:
This message indicates the beginning of a load process on the specified accelerator. If the environment variable AQT_DO_SYSLOGGING is set, this message is written to the system log (SYSLOG) of the connected z/OS system.

User action:
No action is required.

AQT10417I **The *acceleratorName* accelerator has completed load task *loadTaskId*.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

Problem determination:
This message indicates the completion of a load process on the accelerator. If the environment variable AQT_DO_SYSLOGGING is set, this message is written to the system log (SYSLOG) of the connected z/OS system. The message does not indicate whether the load failed or succeeded.

User action:
No action is required.

AQT10418W **Load task *loadTaskId* on accelerator *acceleratorName* is executed with fewer parallel threads than configured. *loadParallelismGoal* threads have been configured, but only *loadParallelismCurrent* threads are running.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

loadParallelismGoal
Configured number of parallel threads

loadParallelismCurrent
Actual number of parallel threads

Problem determination:
The load job is carried out with fewer parallel threads than intended. Too many concurrent load jobs, including those that serve other connected Db2 subsystems, might be the reason.

User action:
The load continues, but the processing time will increase. The number of threads increases automatically when free resources become available again. To manually increase the number of threads for this particular job, identify and cancel other concurrently running load jobs on the accelerator.

AQT10419W **The number of parallel threads for load task *loadTaskId* on accelerator *acceleratorName* has increased again. *loadParallelismGoal* threads have been configured. *loadParallelismCurrent* threads are running.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

loadParallelismGoal
Configured number of threads

loadParallelismCurrent
Actual number of threads

Problem determination:

The number of parallel threads executing this load job has been increased again because the total number of concurrent load jobs went down. An automatic increase occurs only if the number of threads was reduced before.

User action:
No action is required.

AQT10420E **Load task *loadTaskId* on accelerator *acceleratorName* could not load the *tableSchema.tableName* table. The table was not loaded because: *notLoadedCause*.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

tableSchema
Schema name of the affected table

tableName
Name of the affected table

notLoadedCause
Reason why the table could not be loaded

Problem determination:
A particular table could not be loaded. This does not necessarily mean that none of the specified tables could be loaded. This message is only displayed if lock mode TABLE, PARTITIONS, ROW, or NONE is used.

User action

1. Find the faulty table.
2. Remove the cause of the failure.
3. Start a new load for this table only.

AQT10421E **Load task *loadTaskId* on accelerator *acceleratorName* could not load partition *partition* of the *tableSchema.tableName* table. The partition was not loaded because: *notLoadedCause*.**

Explanation

acceleratorName
Name of the accelerator

loadTaskId
Unique identifier of the load task on the accelerator

tableSchema
Schema name of the affected table

tableName
Name of the affected table

partition
Number (identifier) of the affected physical partition

notLoadedCause
Reason why the partition could not be loaded

Problem determination:
A particular partition of a table could not be loaded. This does not necessarily mean that none of the specified tables or partitions could be loaded. This message is only displayed if lock mode PARTITIONS, ROW, or NONE is used.

User action

1. Find the faulty partition.
2. Remove the cause of the failure.
3. Start a new load for this partition only.

AQT10422E **The table *tableSchema.tableName* cannot be restored to Db2 because one or more partitions of this table have already been restored from another accelerator.**

Explanation

tableSchema
Name of the schema that the table belongs to

tableName
Name of the table containing the partitions to be restored

Problem determination:
This warning indicates that the restore operation cannot be executed because archived partitions can only be restored once to Db2.

User action

To ensure a consistent state of the table on both accelerators, complete these steps:

1. Remove the table from the current accelerator.
2. Re-define the table on the same accelerator.
3. Load the table.
4. Use the High Performance Storage Saver to archive the same partitions as on the other accelerator. That is, exclude partitions that have been restored from or not been archived on the other accelerator.

AQT10429E **The network connection from the z/OS IP address *sourceIP* to the accelerator IP address *targetIP***

does not allow the transfer of data messages with a large payload (portion of the user data in a network packet).

Explanation

sourceIP
IP address of the z/OS LPAR that communicates with the accelerator.

targetIP
IP address of the accelerator

Problem determination:
This message indicates that the MTU size or the MTU path discovery not configured properly for the network path between the z/OS LPAR and the accelerator. Large TCP/IP packets might get lost on their way. The loss goes unnoticed. This might in turn lead to communication problems and deadlock situations. As a possible consequence, stored procedure calls do not return anything to the caller.

User action:
With the help of your network administrator, validate the configuration of all network components from the z/OS LPAR to the accelerator including switches, routers, and gateways in-between. Make sure that the MTU size is set properly, that the switches do not drop network packets, and that MTU path discovery is enabled on the switches. If your switches do not support this feature, make sure that all components use the same MTU size.

AQT10441E **An error occurred during the creation of reference tables for federated access. *errorInfo* .**

Explanation

errorInfo
Errors for each failed table

Problem determination:
One or more reference tables for federated access could not be created.

User action:
For details, check the message parameter in the output of the SYSPROC.ACCEL_CREATE_REFERENCE_TABLES stored procedure.

AQT10442E **An error occurred during the removal of reference tables. *errorInfo* .**

Explanation

errorInfo
Errors for each failed table

Problem determination:
One or more reference tables for federated access could not be removed.

User action:
For details, check the message parameter in the output of the SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES stored procedure.

AQT10444E **The REMOTELOCATION column does not exist in the SYSACCEL.SYSACCELERATEDTABLES table.**

Problem determination:
A required column is missing from the SYSACCEL.SYSACCELERATEDTABLES table.

User action:
Make sure you have applied Db2 fix (PTF) UI48595 (for Db2 11) or UI54293 (for Db2 12) and have run the Db2-supplied job DSNTIJAS.

AQT10445E **This version of the accelerator software does not support federated table access.**

Problem determination:
The current software version does not contain the federation functions.

User action:
Update to a of version of the IBM Db2 Analytics Accelerator software that supports federated access.

AQT10446E **Federated access does not work for the type of tables that *tableSchema.tableName* belongs to.**

Explanation

tableSchema
Schema name of the unsupported table

tableName
Name of the unsupported table

Problem determination:
The table type is unsuitable for federated access.

User action:
Remove all tables of this type from the input table set and rerun the stored procedure.

AQT10447E **You do not have the privilege or authorization to grant access to table *tableSchema.tableName* .**

Explanation

tableSchema
Schema name of the unsupported table

tableName
Name of the unsupported table

Problem determination:
You do not have the privilege or authorization to run this stored procedure.

User action:
Obtain the required privilege and rerun the stored procedure.

AQT10448E An accelerator-shadow table with the name *tableSchema*.*tableName* cannot be found on the accelerator.

Explanation

tableSchema
Schema name of the table that cannot be found

tableName
Name of the table that cannot be found

Problem determination:
The stored procedures that grant or revoke federated access, or create reference tables for federated access all refer to accelerator-shadow tables on a particular accelerator. These accelerator-shadow tables must exist or otherwise, the stored procedures cannot run successfully. In the present case, however, one of these tables is missing from the specified accelerator or cannot be found.

User action:
Check to see if the table exists on the accelerator or remove the table from the table set and rerun the stored procedure.

AQT10449E Access to the *tableSchema*.*tableName* table on the *acceleratorName* accelerator by a referencing Db2 subsystem *remoteLocationName* has been revoked or never been granted.

Explanation

tableSchema
Schema name of the table that cannot be accessed

tableName
Name of the table that cannot be accessed

acceleratorName
Name of the accelerator

remoteLocationName
Location name of the referencing Db2 subsystem

Problem determination:
Access to the specified table has been revoked or not been granted.

User action:
Remove the table from the input table set and rerun the stored procedure.

AQT10450E The reference table *tableSchema*.*tableName* already exists.

Explanation

tableSchema
Schema name of the already existing reference table

tableName
Name of the already existing reference table

Problem determination:
The reference table has already been created.

User action

1. Check the schema name and the table name. Correct these if necessary or remove the table from the input table set.
2. Rerun the stored procedure.

AQT10451E The reference table *tableSchema*.*tableName* could not be created because an error occurred.

Explanation

tableSchema
Schema name of the reference table that cannot be created

tableName
Name of the reference table that cannot be created

Problem determination:
The reference table was not created.

User action:
Check the parameters of the stored procedure. You might want to switch on tracing to diagnose the error. Rerun the stored procedure when you have eliminated the error.

AQT10452E A reference table with the name *tableSchema*.*tableName* cannot be found.

Explanation

tableSchema
Schema name of the reference table that cannot be found

tableName
Name of the reference table that cannot be found

Problem determination:
The reference table cannot be found.

User action:
Check the parameters of the stored procedure. You might want to switch on tracing to diagnose the error. Rerun the stored procedure when you have eliminated the error.

AQT10453E **A database with the name *databaseName* does not exist.**

Explanation

databaseName
Name of the database

Problem determination:
The specified database does not exist on the accelerator.

User action

1. Correct the database name or leave out the `userSpecifiedDatabase` parameter.
2. Rerun the stored procedure.

AQT10454E **The reference table *tableSchema.tableName* already exists on the accelerator.**

Explanation

tableSchema
Schema name of the already existing reference table

tableName
Name of the already existing reference table

Problem determination:
The reference table cannot be created because it already exists.

User action:
Remove the table name from the input table set and rerun the stored procedure.

AQT10455E **Access to the *tableSchema.tableName* table has not been granted.**

Explanation

tableSchema
Schema name of the table that cannot be accessed

tableName
Name of the table that cannot be accessed

Problem determination:
You cannot create the reference table because the owner has not (yet) given you access to the underlying accelerator-shadow table.

User action

1. Ask the table owner to give you access to the table or run the SYSPROC.ACCEL_GRANT_TABLES_REFERENCE on the owning Db2 subsystem if you have sufficient privileges.
2. Then rerun the stored procedure to create the reference table.

AQT10456E **The *tableSchema.tableName* table has not been loaded.**

Explanation

tableSchema
Schema name of the empty table

tableName
Name of the empty table

Problem determination:
The table contains no data.

User action:
Load the table on the accelerator with data and rerun the stored procedure.

AQT10457E **You don't have the privilege or authorization to revoke table access.**

Problem determination:
You're not allowed to remove access to the tables.

User action:
Obtain the required privilege or authorization and rerun the stored procedure.

AQT10458E **An error occurred while running a stored procedure for federated access support.**

Problem determination

An error occurred in connection with one of the following stored procedures:

- SYSPROC.ACCEL_GRANT_TABLES_REFERENCE
- SYSPROC.ACCEL_CREATE_REFERENCE_TABLES
- SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES
- SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE

User action:
Diagnose the error, eliminate its cause, and rerun the stored procedure.

AQT10459E **The current version of Db2 for z/OS does not support federated access.**

Problem determination:
You cannot use federated access in connection with your version of Db2 for z/OS.

User action:
To be able to use federated access, you must migrate to Db2 11 for z/OS or an even higher version.

AQT10460E **You cannot run the SYSPROC.ACCEL_ALTER_TABLES stored procedure on the *tableSchema*.*tableName* table because federated access has been enabled for this table.**

Explanation

tableSchema
Schema name of the table that cannot be altered

tableName
Name of the table that cannot be altered

Problem determination:
You cannot change the distribution key or organizing keys of a table as long as federated access is possible for this table.

User action:
Remove the table from the input table set and rerun the stored procedure.

AQT10464E **The table *newTableSchema*.*newTableName* cannot be added because its database ID and object ID (*clientTableID*) are used by the table *existingTableSchema*.*existingTableName*, which already exists on the accelerator.**

Explanation

newTableSchema
Schema name of the table to be added

newTableName
Name of the table to be added

clientTableID
Client table ID (dbid/obid) of the table to be added

existingTableSchema
Schema name of the already existing table

existingTableName
Name of the already existing table

Problem determination:
The table cannot be added because its identifier is used by an already existing table.

User action:
Remove the existing table from the accelerator and rerun the stored procedure.

AQT10500W **An error prevented the stored procedure from returning all the result sets. However, the result that contains the output messages (status information regarding the stored-procedure run, associated with the MESSAGES_CURSOR) might be available none the less. If not, then you can find this information in the MESSAGE output parameter. If trace files could not be transferred to the result set associated with the SP_TRACE_CURSOR, they might have been left on the z/OS system, in the following location: *traceFiles***

Explanation

traceFiles
Location of trace files that were left on the z/OS system

Problem determination:
This warning indicates that your stored-procedure-run did not return all the result sets that are normally returned by the stored procedure.

User action

1. Let applications that call the stored procedure dynamically process only the available result sets.
2. Inspect the MESSAGE output parameter, or, if available, the MESSAGES_CURSOR result set to find the cause of the problem.
3. If trace files have been left on the z/OS system, the location is indicated in this message. Check these messages if they are available.

AQT10501E **The *procSchema*.*procName* stored procedure is incorrectly defined in DB2. The *parameterName* parameter of type *parameterType* was not found.**

Explanation

procSchema
Schema name of the stored procedure

procName
Name of the stored procedure

parameterName
Name of the expected parameter in the CREATE PROCDURE DDL statement.

parameterType
Type of the expected parameter in the CREATE PROCDURE DDL statement.

Problem determination:
An expected stored-procedure parameter was not found.

User action

- Make sure that the DDL statement for the stored procedures are correct. (These are usually set by the installer program AQTTRIN, except for remote wrapper stored procedures)
- Define at least the message output parameter as a CLOB with a size of 64K.
- Regarding remote wrapper procedures, contact the DDL provider for instructions.
- For procedures supplied with the product, verify that these were installed correctly. If so, contact IBM support.

AQT10502W **Wrong value *`unicodeSortOrderValue`* for UNICODE_SORT_ORDER. The value `BinaryUTF8` is taken instead.**

Explanation

`unicodeSortOrderValue`
Ignored UNICODE_SORT_ORDER value

Problem determination:
This warning indicates that the stored procedure should not be called with a UNICODE_SORT_ORDER value other than `BinaryUTF8`.

User action:
To avoid this warning, specify `BinaryUTF8` as the value of UNICODE_SORT_ORDER.

AQT10503E **The accelerator could not be contacted. Diagnostic information with details about the connectivity problem: *errorMessage***

Explanation

errorMessage
Additional diagnostic information about the error

Problem determination:
This exception occurs if the version of an accelerator cannot be determined because the accelerator cannot be contacted.

User action:
The diagnostic information in the message text gives you more details about the error. It might have occurred because the input to the stored procedure was incorrect. Check if the network address and the port number are correct.

AQT10504W **The *tableSchema.tableName* table can not be removed because it is still enabled for query acceleration. Disable query acceleration before you remove the table.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
You cannot remove a table if it is still enabled for query acceleration unless you run the SYSPROC.ACCEL_REMOVE_TABLES stored procedure with the force option. In doing so, however, you will end currently running queries prematurely.

User action

Either:

1. Disable query acceleration for the table.
2. Rerun the SYSPROC.ACCEL_REMOVE_TABLES stored procedure.

Or:

Run the SYSPROC.ACCEL_REMOVE_TABLES stored procedure with the force option.

AQT10505W **The *tableSchema.tableName* table cannot be removed because the data of one or more of its partitions has been archived on an accelerator. Restore the data to these partitions before you remove the table.**

Explanation

tableSchema
Schema name of the irremovable table

tableName
Name of the irremovable table

Problem determination:
You cannot remove a table if data from one or more of its partitions has been archived on an accelerator. Removing such a table without restoring the data is possible only if you run the

SYSPROC.ACCEL_REMOVE_TABLES stored procedure with the force option. In doing so, however, you forfeit the chance of an easy recovery.

User action

Recommended:

1. Make sure that all archiving operations from this table have been completed.
2. Restore the archived partition data by using the appropriate function in your administration client or by running the SYSPROC.RESTORE_ARCHIVE_TABLES stored procedure.
3. Remove the table from the accelerator.

Not recommended: If you are sure that you want to remove the table from the accelerator without restoring the data, you can run the SYSPROC.ACCEL_REMOVE_TABLES stored procedure with the force option. Note that if you need to restore the data after that, you must complete the following steps:

1. Manually reset the state of the affected Db2 table space to read/write.
2. Manually recover the archived data from the Db2 image copies that were created at the time when the data was archived.

AQT10506S **The data set referenced by the statement DD:AQTDEF6 cannot be opened or is empty.**

Problem determination:
The data set referenced by the statement DD:AQTDEF6 cannot be found or the data set is empty.

User action:
Make sure that the data set exists, that it is readable, and that it contains valid XML code that conforms to the corresponding XML schema.

AQT10507W **At least one column of the *tableSchema*.*tableName* table has been altered, added, or removed in Db2 for z/OS after defining the table on the *acceleratorName* accelerator.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

acceleratorName
Name of the accelerator

Problem determination:
This exception occurs when a column definition was changed or a column was added or removed from a table by an ALTER TABLE statement after adding the table to an accelerator.

User action

1. Remove the table from the accelerator.
2. Re-add the table to the accelerator.

AQT10508E **An unsupported stored procedure or sub-function of a procedure has been called by IBM Db2 Analytics Accelerator for z/OS version *version*: *unsupportedFunction***

Explanation

unsupportedFunction
Name or description of the unsupported procedure or function.

version
Version of the stored procedure

Problem determination:
This exception occurs when an unsupported stored procedure or sub-function has been called.

User action:
The function is not supported and cannot be used with version of the stored procedure.

AQT10509E **In the input parameter (XML string) of the SYSPROC.ACCEL_TEST_CONNECTION stored procedure, the version element, which is a sub-element of the diagnosticCommand element, contains an invalid combination of attributes. That is, the 'accelerator' attribute is used together with the 'ip' or 'port' attribute. For details, see: *errorMessage*.**

Explanation

errorMessage
Additional diagnostic information about the error.

Problem determination:
This error occurs if you run the SYSPROC.ACCEL_TEST_CONNECTION stored procedure with the DIAGNOSTIC_SPECIFICATION parameter and if the value of this parameter (an XML string) contains an invalid combination of attributes.

User action:

Use the 'accelerator' attribute only, or both, the 'ip' and 'port' attributes.

AQT10510E **The threads could not be started within the timeout period of *timeout* seconds.**

Explanation

timeout
Timeout period in seconds

Problem determination:
This error occurs if the SYSPROC.ACCEL_LOAD_TABLES or the SYSPROC.ACCEL_TEST_CONNECTION stored procedure cannot start parallel threads within the given timeout period. A timely start might be prevented by a high system workload with concurrent operations.

User action:
Rerun the procedure operation when the system utilization is lower.

AQT10511W **The *packageFileName* package was deployed (copied) without an integrity check because you neither specified a hash value for comparison, nor an algorithm or method for the hash verification.**

Explanation

packageFileName
Name of the unchecked package file

Problem determination:
The integrity of the package could not be checked after deploying the file, such that a future activation step might fail.

User action:
No action is required at this point. If the activation of the package fails, remove the package and deploy it again.

AQT10512E **The accelerator listening on port *usedPort* at IP address *usedAddress* is running on the *platform* platform. The stored procedure SYSPROC.ACCEL_ADD_ACCELERATOR2 did not find the license module *member*, which entitles you to use that platform.**

Explanation

usedAddress
Network address used to contact the accelerator

usedPort
Port number used to contact the accelerator

platform
Hardware platform the accelerator is running on

member
License module (member of SAQTMOD) containing the hardware-specific license

Problem determination:
This error occurs if a hardware-specific license module is missing from the SAQTMOD data set of your installation. The license module is needed to operate an accelerator on that particular hardware.

User action:
Check your installation on z/OS. You must install the entitlement FMID for the platform of the accelerator.

AQT10513I **The external load utility function *callbackFunctionName* ended with return code *returnCode* and the following message: *message*. Context: *diagnosticInformation*.**

Explanation

callbackFunctionName
Name of the external load utility function

diagnosticInformation
Additional information about the context

returnCode
Return code issued by the external load utility function

message
Message issued by the external load utility function

Problem determination:
An external load utility function called by SYSPROC.ACCEL_LOAD_TABLES issued a return code and a message. This message might contain important information or instructions.

User action:
Follow the instructions in the message (if any).

AQT10514W **The external load utility function *callbackFunctionName* ended with return code *returnCode* and the following warning: *message*. Context: *diagnosticInformation*.**

Explanation

callbackFunctionName
Name of the external load utility function

diagnosticInformation
Additional information about the context

returnCode
Return code issued by the external load utility function

message
Message issued by the external load utility function

Problem determination:
An external load utility function called by SYSPROC.ACCEL_LOAD_TABLES issued a return code and a warning.

User action:
Heed the warning, as it might hint at unwanted circumstances or side-effects.

AQT10515E **The external load utility function *callbackFunctionName* ended with return code *returnCode* and the following error message: *message*. Context: *diagnosticInformation*.**

Explanation

callbackFunctionName
Name of the external load utility function

diagnosticInformation
Additional information about the context

returnCode
Return code issued by the external load utility function

message
Message issued by the external load utility function

Problem determination:
An external load utility function called by SYSPROC.ACCEL_LOAD_TABLES issued a return code and an error message.

User action

1. Use the information in the message to eliminate the error.
2. Rerun the SYSPROC.ACCEL_LOAD_TABLES stored procedure.

AQT10517I ***component=status***

Explanation

component
Internal name of the software component to be updated

status
Status of the software update for the component

Problem determination:
When you run the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure with the **`getActivationLog`** sub-command during a software update, the procedure tries to retrieve the update status for each software component. The status is reported in a separate message for each component.

User action:
Monitor the update progress until the process has been completed for each component. If a software update could not be completed, contact IBM support.

AQT10518W **The stored procedure is unable to connect to the *acceleratorName* accelerator. For that reason, the activation log was retrieved from the software update component.**

Explanation

acceleratorName
Name of the accelerator

Problem determination:
You ran SYSPROC.ACCEL_CONTROL_ACCELERATOR with the `<getActivationLog/>` command to retrieve the software activation log from the specified accelerator. However, the accelerator stopped during a software update and cannot respond at the moment. If that happens, the stored procedure retrieves the activation log from the software update component instead.

User action:
If you get this message during a planned software update, the stored procedure behaves as expected. Otherwise, check why the accelerator is offline. If you cannot find the cause and solve the problem, contact IBM support.

AQT19000I **The AQTTRIN program was started for Db2 subsystem *subsystemId* Db2 for z/OS version *db2Version*, mode (*systemMode*)**

Explanation

subsystemId
Name of the Db2 subsystem

db2Version
Version of Db2

systemMode
Current Db2 mode. Possible values are: CMn Conversion mode, where n represents the Db2 version that you migrate from. NFM New-function mode

Problem determination:

This message is returned when the AQTTRIN program starts. It gives you the Db2 version and the processing mode of the Db2 subsystem that the program is connected to.

User action:
No action is required.

AQT19001I **The AQTTRIN program reports an internal error at location *locationName* because an unexpected object of the type *objectType* was encountered. The name of this object is *objectName*.**

Explanation

locationName
Diagnostic information that identifies the function that was involved in the error.

objectType
Type of the object

objectName
The name of the object, which can be a specific name or a qualified name.

Problem determination:
The program ended because an unexpected object was encountered.

User action:
Check the maintenance level of the AQTRIN program. Update the program to the newest level if necessary. If the problem persists, contact IBM support.

AQT19002I **The AQTTRIN program received the unexpected SQL code *sqlCode* from the SQL statement *sqlRequest*.**

Explanation

sqlCode
SQL code (a numeric value that hints at the cause of the SQL error)

sqlRequest
All or part of the SQL statement that was issued. Long statements are truncated.

Problem determination:
The AQTTRIN program received an unexpected SQL code while it was processing an SQL statement. Processing ended for that reason.

User action:
Examine the standard SQLCA diagnostic messages and output in the SYSPRINT output log at the point of failure to determine the cause of the error. Correct the error and rerun the program.

AQT19003I **AQTTRIN has encountered an error in the following control statement: dd = *ddName*, error = *errorText*, line = *lineNumber*, statement = *errorStatement***

Explanation

ddName
The name of the data definition (DDL) statement.

errorText
Additional information about the error. Possible values: - CONTROL STATEMENT HAS AN INVALID FORMAT - ROUTINE NAME HAS AN INVALID FORMAT - CONTROL STATEMENT EXCEEDS 720 BYTES - NO VALUE FOUND FOR THE FIRST|SECOND|THIRD PARAMETER OR ONE OR MORE CONTROL STATEMENTS WAS NOT SEPARATED BY A BLANK LINE - ONLY ONE PARAMETER FOUND - ACTUAL PARAMETERS: x, EXPECTED PARAMETERS: y - UNEXPECTED PARAMETER parameter-name FOUND - CONTROL STATEMENT FOR routine-name NOT SUPPORTED - CONTROL STATEMENT FOR routine-name ALREADY PROCESSED - CONTROL STATEMENT FOR routine-name NOT FOUND

lineNumber
The location (line number) of the error.

errorStatement
The control statement that caused the error.

Problem determination:
AQTTRIN, the configuration program for routines that are supplied by IBM Db2 Anayltics Accelerator, encountered an error while it was processing a configuration control statement.

User action:
See the faulty control statement at the indicated line in the input, which was allocated by the specified data definition statement. Analyze and correct the problem. Then resubmit the AQTxxxxx job to resume processing.

AQT19004I **The AQTTRIN program has successfully created the *objectType objectName* object.**

Explanation

objectType
Type of the object

objectName
Name of the object, which is a specific name or a qualified name: object-name is a specific name SPECIFIC NAME name object-name is a qualified name qualifier.name

Problem determination:

The AQTTRIN program has successfully created the SQL object.

User action:
No action is required.

AQT19005I **The AQTTRIN program has skipped *processingType* processing of the *objectType*, *objectName* object for the following reason: *reasonText***

Explanation

processingType
Attempted SQL operation

objectType
Type of the object

objectName
Name of the object, which is a specific name or a qualified name: object-name is a specific name SPECIFIC NAME name object-name is a qualified name qualifier.name

reasonText
This operation requires Db2 to be run in new-function mode.

Problem determination:
The AQTTRIN program could not complete SQL processing for the indicated object.

User action:
If you receive reason code 10, run the job again after Db2 has entered the new-function mode. Otherwise, no action is required.

AQT19006I **The authorization IDs *authorizationIds* were granted *accessType* access on the *objectType objectName* object.**

Explanation

accessType
Authorization or access that was granted

objectType
Type of the object

objectName
Name of the object

authorizationIds
The authorization IDs to which access was granted.

Problem determination:
The AQTTRIN program has granted access on one or more objects to one or more authorization IDs as indicated in the message.

User action:
Verify that the granted access is appropriate for each of the authorization IDs. If the access is not appropriate, modify the GRANTTO parameter of the configuration control statement for each stored procedure or user-defined function that requires a different access list. Then resubmit the AQTTIJSP job.

AQT19007I **The AQTTRIN program has dropped (deleted) the *objectType*, *objectName* object.**

Explanation

objectType
Type of the object

objectName
Name of the object, which is a specific name or a qualified name: object-name is a specific name SPECIFIC NAME name object-name is a qualified name qualifier.name

Problem determination:
The indicated object has been dropped (deleted). The AQTTRIN program drops existing objects only if the program runs in REINSTALL mode or REINITIALIZE mode.

User action:
No action is required.

AQT19008I **The AQTTRIN program has set the Workload Manager (WLM) environment for the *routineType objectName* object to *environmentName*.**

Explanation

routineType
The type of routine: user-defined function or stored procedure.

objectName
Name of the object, which is a specific name or a qualified name: object-name is a specific name SPECIFIC NAME name object-name is a qualified name qualifier.name

environmentName
Name of the Workload Manager (WLM) environment.

Problem determination:
A WLM environment has been assinged to a stored procedure or user-defined function as shown.

User action:
Verify that the assigned WLM environment is appropriate for the stored procedure or function. If the WLM environment is inappropriate, modify the WLMENV parameter of the configuration control statement for each stored procedure or user-defined

function that requires to be run in a different environment. Then resubmit the AQTTIJSP job.

AQT19009I **The AQTTRIN request to bind a package for collection ID *collectionId.packageName* succeeded with return code *returnCode*. The owner of the package is *packageOwner*.**

Explanation

returnCode
Return code of the command

collectionId
Collection ID of the bound package

packageName
Name of the bound package

packageOwner
Name (user ID) of the package owner

Problem determination:
A bind job was completed for the indicated package.

User action:
No action is required.

AQT19010I **AQTTRIN *requestType* plan request for *packageName* failed with return code *returnCode*.**

Explanation

requestType
The type of request: BIND PACKAGE, BIND PLAN, FREE PACKAGE, or FREE PLAN.

packageName
Name of the package that could not be bound or freed

returnCode
Numeric value that indicates the reason for the failure.

Problem determination:
The requested operation was rejected by Db2. See the diagnostic messages that were issued before this message.

User action:
Check the diagnostic messages from Db2. Correct the problem. Then resubmit the AQTTRIN request.

AQT19011I **One or more of the submitted input parameters are invalid. `Reason code =` *`reasonCode`*`,` `text =` *`reasonText`***

Explanation

reasonCode
The reason code describes the cause of the error:

Reason code = 1
The number of parameters is incorrect. The text of the reason code suggests that fewer or more parameters were submitted than expected, where:

number-passed
The number of submitted parameters

number-expected
The number of expected parameters

Reason code = 2
A required parameter is missing. Text of the reason code:

```
REQUIRED PARAMETER parameter-name WAS NOT
PASSED
```

where `parameter-name` is the name of the expected parameter, which was not provided.

Reason code = 3
An unknown parameter was submitted. Text of the reason code:

```
UNRECOGNIZED PARAMETER parameter-name WAS
PASSED
```

where `parameter-name` is the name of the submitted parameter that was not expected.

Reason code = 4
One of the parameter values is invalid. Text of the reason code:

```
THE VALUE FOR PARAMETER parameter-name IS
NOT VALID
```

where `parameter-name` is the name of the parameter with the invalid value.

Reason code = 5
The routine specified in the `DD:SYSIN` statement is not a valid Db2-supplied routine. Text of the reason code:

```
schema.routine-name IS NOT A VALID Db2-
SUPPLIED ROUTINE
```

where `schema.routine-name` is the qualified name of the invalid routine.

Reason code = 6
The same parameter was specified two or more times. Text of the reason code:

```
'PARAMETER parameter-name WAS PASSED MORE
THAN ONCE.'
```

Reason code = 7
The length or size of the submitted statement exceeds the allowed maximum number of bytes. Text of the reason code:

```
'STATEMENT EXCEEDS number-of-bytes BYTES'
```

where `number-of-bytes` denotes the byte limit.

reasonText
Text that comes with the reason code

Problem determination:
The program encountered an error as it attempted to process the input parameters in the PARM parameter or the `SYSIN DD` statement.

User action:
Correct the problem and run the program again.

AQT19013I **The AQTTRIN program for the Db2 versions *supportedVersion1* and *supportedVersion2* cannot be run on Db2 version *db2Version*.**

Explanation

supportedVersion1
The first supported version of Db2.

supportedVersion2
The second supported version of Db2.

db2Version
The Db2 version.

Problem determination:
The maintenance levels of the AQTTRIN program and Db2 do not match. This happens, for example, if, during the migration to a new Db2 version, the program was run on a member of a Db2 data sharing group whose maintenance level is lower than that of the program. To run members of a data sharing group at different maintenance levels is called coexistence. It is a common practice.

User action:
If you have a Db2 data sharing group with coexisting members, run the program on a member that has been migrated to the latest version. If this is not possible, make sure that the load library that is specified for program execution in the JOBLIB or STEPLIB statement is at the same maintenance level as Db2.

AQT19014I **The AQTTRIN program reported a *fileOperation* file error for the DD:*ddName* statement. Diagnostic text: *diagnosticText***

Explanation

fileOperation
The file operation during which the error occurred, that is, OPEN, READ, WRITE, or CLOSE

ddName
The name of the data definition (DDL) statement

diagnosticText
Additional diagnostic information about the error, as provided by the IBM Language Environment®.

Problem determination:
The program detected an error in the data set that was allocated to the data definition statement.

User action:
Correct the error as suggested in the diagnostic information.

AQT19015I **The preview option was specified for the AQTTRIN program. Therefore, the *installType* action was not carried out. An installation JCL job was generated and stored in the *storedLocation* member.**

Explanation

installType
Type of the installation job

storedLocation
Contains the name of the data set and the name of the JCL member in which the generated installation job was stored.

Problem determination:
The Db2-supplied routines and supporting objects were not installed, configured, or bound to the system because the preview option was specified. To generate a JCL that contains the actual SQL and BIND statements that eventually install the Db2-supplied routines, you must specify a DD statement with the name JCLOUT in the JCL. If you want to know which objects you must include for processing, see the detailed report in SYSPRINT.

User action:
See the report in SYSPRINT. Modify the AQTTIJSP installation job so that it includes the required objects. Then rerun the job without the preview option or submit the generated JCL.

AQT19025I **The preview option of the AQTTRIN program was specified. Therefore, the *installType* action was not carried out.**

Explanation

installType
Type of installation

Problem determination:
The Db2-supplied routines and supporting objects were not installed, configured, or bound to the system because the preview option was specified. You find details about this job in SYSPRINT.

User action:
See the report in SYSPRINT. Modify the AQTTIJSP installation job so that it includes the required objects. Then rerun the job without the preview option.

AQT19033I The AQTTRIN installation program ended with return code *returnCode*.

Explanation

returnCode
The most severe return code encountered at any time during processing.

Problem determination:
Batch processing ended abnormally.

User action:
If the return code is 4 or higher, look for other messages that might explain the cause of the warning or failure. Proceed as instructed by the message.

AQT19052I The authorization ID *authorizationId*, is not a Db2 installation system administrator ID. The authorization of the submitted ID is insufficient.

Explanation

authorizationId
Specified authorization ID

Problem determination:
The authorization ID, which was specified in the AUTHID parameter for the program, is not a Db2 installation system administrator ID. However, the authorization of such an ID is required if stored procedures or user-defined functions must be marked as system-defined procedures or functions. This is the case here.

User action:
Change the value of the AUTHID parameter so that it provides a Db2 installation system administrator ID. Then run the AQTTIJSP job again.

AQT19053I The AQTTRIN program was unable to process the IFI read request for IFCID 106. Return code: *returnCode*, reason code: *reasonCode*, bytes that were not moved to the instrumentation facility communications area (IFCA): *bytesRemaining*

Explanation

returnCode
Return code from IFI

reasonCode
Reason code from IFI

bytesRemaining
Number of bytes that did not fit in the return area and therefore still remains in the buffer

Problem determination:
The AQTTRIN program tried a synchronous read of IFCID trace record 106, but received an unexpected return code from the Db2 Instrumentation Facility Interface (IFI).

User action:
To find the cause of the error, check the return code and the reason code for information about the IFCARC1 and IFCARC2 values in the instrumentation facility communications area (IFCA). Correct the error. Then rerun the AQTTIJSP job.

AQT19054I One or more objects of the AQTTRIN program must be updated. The update might require the rebinding of packages that reference these objects.

Problem determination:
Run in preview mode, the AQTTRIN program found one or more objects that must be updated. However, the program cannot determine whether packages that refer to these objects will become invalid as a result of the update. If this is the case, you must rebind these packages.

User action:
If the ABIND subsystem parameter is set to NO and you want to use the generated JCL job to install or update the Db2-supplied routines, resubmit the AQTTIJSP installation job with the MODE(INSTALL-PREVIEW) option. This identifies invalid packages that must be rebound and generates corresponding BIND statements. Finally, run these BIND statements. If the ABIND subsystem parameter is set to YES, or if you run the AQTTIJSP job without the preview option, no action is required because in either case, Db2 for z/OS rebinds invalidated packages automatically.

AQT19400I The job was completed successfully (SQL code 0).

Problem determination:
You might see warnings (SQLWARN indicators) although the job could be completed successfully.

User action:
You can ignore the warnings. No action is required.

AQT19909I The AQTTRIN bind plan request succeeded with return code *returnCode* for the *packageName* package with owner *packageOwner*.

Explanation

returnCode
Return code of the command

packageName
Name of the bound package

packageOwner
Owner (ID) of the bound package

Problem determination:
The bind request has been carried out successfully.

User action:
No action is required.

AQT19910I **The AQTTRIN *requestType* request for the *collectionId*.*packageName* package failed with return code *returnCode*.**

Explanation

requestType
Type of the request: BIND PACKAGE, BIND PLAN, FREE PACKAGE, or FREE PLAN

collectionId
Collection ID of the package that could not be bound or freed

packageName
Name of the package that could not be bound or freed.

returnCode
A numeric value that indicates the reason for the failure

Problem determination:
The request was rejected by Db2. Diagnostic messages have been issued in addition to this message.

User action:
Check the diagnostic Db2 messages to identify the problem. Correct the problem and resubmit the request.

AQT20001W **A plan with an ID of *planID* cannot be found for SQL statement *statementID* with statement text *statement* on the *acceleratorName* accelerator.**

Explanation

statementID
The accelerator ID of the SQL statement for which a corresponding plan file cannot be found

planID
The Netezza ID of the plan file that does not exist

statement
The original SQL statement

acceleratorName
Name of the accelerator

Problem determination:
This message indicates that a query was processed by an accelerator, but the corresponding plan information can no longer be retrieved. The plan might have been deleted from the accelerator. Query plans are kept for a limited time only. Newer queries take precedence over older queries, so that the oldest plans are deleted after some time.

User action:
Rerun the query to collect fresh plan information.

AQT20003W **ATTENTION: Potential risk of data loss. A unique constraint, such as a primary key or unique index, is not defined for the *tableSchema*.*tableName* table. If multiple rows are inserted into a Db2 table, all or a subset of these might end up as a single row in the corresponding accelerator-shadow table. Similarly, a deletion of one row from the original table, might result in the deletion of multiple rows from the accelerator-shadow table. This leads to inconsistent tables on either side, meaning that incorrect query results are returned by the accelerator. Therefore, do not enable incremental updates for this table unless you know for sure that it does not contain duplicate rows. Another problem with tables of this type is that queries and incremental updates might take much longer.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
Without a unique constraint, all columns of a table are used to identify a table row. There might be multiple rows with exactly the same column values. If that is the case, inserting the same row twice results in the insertion of only one row into the accelerator-shadow table. A deletion of one row from a Db2 for z/OS table, on the other hand, will cause the deletion of all rows with the same

values from the corresponding accelerator-shadow table. This is because the duplicate rows cannot be differentiated. The consequence are inconsistent tables, which means that queries on the accelerator do not run on the same data as in Db2. This leads to incorrect results. The absence of a unique constraint also means that the identification of table rows takes longer because the entire table must be scanned. For large tables, this might result in a poor query and update performance.

User action

To add a unique constraint to the table, follow these steps:

1. Disable incremental updates for the table.
2. Remove the accelerator-shadow table.
3. In Db2, define the unique constraint.
4. Redefine the accelerator-shadow table.
5. Enable incremental updates for the redefined table.
6. Load the redefined table.

AQT20004W **The *tableSchema*.*tableName* table is enabled for replication, but has no unique constraint that matches one or more of its organizing keys. This lack might cause incremental updates to be applied very slowly to the table. The unique constraints are: *uniqueConstraints*. The organizing keys are: *organizingKeys*.**

Explanation

tableSchema
Schema name of the table that is being modified and still in use by some other task

tableName
Name of the table that is being modified and still in use by some other task

uniqueConstraints
Unique constraints on the table

organizingKeys
Organizing keys on the table

Problem determination:
A large number of row updates or deletions cannot be applied efficiently because the absence of an organizing key that matches one or more of the table's unique constraints requires a scan of the entire table. The identification of the rows to be updated thus takes longer.

User action:
1) Disable replication (incremental updates) for the affected table. 2) Define the orgainizing keys for the table in such a way that one or more of the keys match a unique constraint. 3) Re-enable replication for the table.

AQT20006E **At least one of the tables that are to be loaded with lock mode *lockMode* is enabled for incremental updates. To prevent inconsistencies between the source tables in Db2 for z/OS and the tables on the accelerator, tables that are enabled for incremental updates must be loaded with lock mode TABLESET. Lock mode ROW is also allowed but imposes special restrictions.**

Explanation

lockMode
Lock mode used for table loads

Problem determination:
A capture point for incremental updates is set at the end of a load operation. Updates to the source tables that occurred before the setting of the capture point are not propagated as incremental updates, which causes Db2 source tables and accelerator tables to be out of sync. To avoid a situation like this, you must use a lock mode that prevents updates of the source tables until the capture point is set during a load. If you cannot tolerate the Db2 lock scope implied by lock mode TABLESET, you can also use lock mode ROW. However, this requires incremental updates to be stopped during the load and additional measures to guarantee data consistency.

User action:
Load the tables with lock mode TABLESET.

AQT20007E **The *acceleratorName* is in the '*currentState*' state, which does not allow a *attemptedOperation* operation.**

Explanation

acceleratorName
Name of the affected accelerator

currentState
Current replication state

attemptedOperation
Attempted operation that failed due to the replication state

Problem determination:
The current replication state does not allow the attempted operation. For example, the state might

be in the process of being changed, or incremental updates are not enabled for the Db2 subsystem.

User action:
If incremental updates are disabled for the affected Db2 subsystem, enable these from the IBM Db2 Analytics Accelerator Configuration Console. If incremental updates are currently being started or stopped, wait for the operation to complete, then retry running the operation.

AQT20009E **Operation *operation* failed: *errorMessage***

Explanation

operation
ID of the incremental update operation that reported the error.

errorMessage
An error message generated by the incremental-update automation component.

Problem determination:
Mostly, this error is connected with incremental update operations. A common cause is a structural change (DDL change) in one of the affected tables since the last successful operation. For example, a column or a constraint might have been added to a table in Db2. In such a case, incremental update process fail or queries can no longer be routed to an accelerator.

User action

1. Retry running the requested operation.
2. If the problem persists and incremental updates are involved, look at the last replication event message in the list to diagnose its cause. Click the Get Events link in your administration client to display the list of replication event messages. If incremental updates can be ruled out, try to find the cause of the error by just looking at the diagnostic information in this error message or the error log in your administration client.
3. Identify the tables that have caused a problem.
4. Stop all incremental update processes that these tables are part of.
5. Remove the tables from all accelerators.
6. Redefine the tables.
7. Enable incremental updates for the redefined tables.
8. Load the redefined tables.

If the problem is not connected with incremental update processes, complete just steps 1, “2” on page 177, “3” on page 177, “5” on page 177, “6” on page 177, and “8” on page 177.

If you cannot identify the tables or cannot solve the problem by following the previous steps, contact IBM support.

AQT20010E **Replication can not be started on the *acceleratorName* accelerator until at least one table is enabled for replication and loaded.**

Explanation

acceleratorName
Name of the accelerator

Problem determination:
You can only start replication if at least one table is is enabled for replication and loaded, that is, the #Replication Since# column in the GUI must show a timestamp value.

User action:
Enable at least one table for replication and load it, then start replication.

AQT20011W **The following columns of the *tableSchema*.*tableName* table contain TIME values that were converted from 24:00:00 to 23:59:59 or TIMESTAMP values whose time part was converted from 24:00:00.000000 to 23:59:59.999999: *columnNames*. Hence query results returned by the accelerator might differ from those that Db2 for z/OS returns.**

Explanation

tableSchema
Schema name of the table that contains TIME or TIMESTAMP columns with converted values

tableName
Name of the table that contains TIME or TIMESTAMP columns with converted values

columnNames
Names of the TIME or TIMESTAMP columns

Problem determination:
In earlier product versions before version 4 PTF-2, a Db2 TIME value 24:00:00 or a TIMESTAMP value whose time portion is 24:00:00.000000 cannot be loaded on the accelerator. By default, such values cause a conversion error, but the accelerator can be configured to convert these values or value portions to 23:59:59 or 23:59:59.999999. This warning informs you that such a conversion took place while the table was being loaded.

User action

- Consider upgrading the product to version 4 PTF-2 or later.
- You can also modify the original values in Db2 for z/OS to avoid this warning and guarantee result consistency, but this is no longer recommended.

AQT20012W **Incremental updates have been enabled for the *tableSchema*.*tableName* table. To guarantee the consistency of the original Db2 table and its counterpart on the accelerator, this table has been reset to InitialLoadPending state, which requires a reload of the entire table. Without this reload, you cannot run accelerated queries or start incremental update processes for this table.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
If incremental updates are enabled for a loaded table, there might be pending changes that occurred after the last load. The incremental update process does not propagate such changes to the corresponding accelerator tables because changes are not registered by the process before the capture point is set. The capture point, however, is set when the incremental update process starts. So there might be changes that go unnoticed. To avoid such a condition, loaded tables are reset to InitialLoadPending state when they are enabled for replication, which requires a fresh reload of the table data before the incremental update process begins. This ensures that no table changes go unnoticed.

User action:
Reload the table.

AQT20013E **Task *canceledTaskID* of type *taskType* has been canceled by task *cancelingTaskID* of type *cancelingTaskType* for the following reason: *cancelReason*.**

Explanation

canceledTaskID
Unique identifier of the canceled task

taskType
Type of the canceled task

cancelReason
Reason for the task cancelation

cancelingTaskID
Identifier of the task that caused the cancelation

cancelingTaskType
Type of the task that caused the cancelation

Problem determination:
This error occurs when a user cancels an accelerator task from your administration client or by running the SYSACCEL.ACCEL_CONTROL_ACCELERATOR stored procedure. This error also occurs when a task is canceled by another, conflicting task. For example, removing a table from the accelerator leads to the cancellation of any other operation on the table happening at the same time.

User action:
If you want to prevent other users from canceling a task, adjust the execution privileges for the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

AQT20014I **The following data was transferred to the *tableSchema*.*tableName* table: Scope: *scope*, number of rows: *numRows*, amount of data: *dataSizeInMB* MB, time: *timeInSeconds* seconds.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

scope
Scope of the data that was transferred (full table or selected partitions)

numRows
Number of rows that were transferred

dataSizeInMB
Amount of data that was transferred

timeInSeconds
Time that was needed to transfer the data

Problem determination:
This message, which is issued for each transaction of the same type, reports information about the table data that was transferred by the SYSPROC.ACCEL_LOAD_TABLES stored procedure.

AQT20015I **No data was transferred for table *tableSchema*.*tableName*.**

Explanation

tableSchema
Schema name of the table that was skipped

tableName
Name of the table that was skipped

Problem determination:
You want to load or move a table, but no data is transferred to the accelerator. This can happen if the detectChanges='DATA' attribute is specified for the SYSPROC.ACCEL_LOAD_TABLES stored procedure, but table changes cannot be detected or if the data of the specified partitions has been moved to an accelerator by the High Performance Storage Saver.

User action:
No action required.

AQT20016W **An error occurred during the re-enablement of incremental updates for the Db2 subsystem with the location name *locationName*. You must enable incremental updates for a Db2 subsystem from the IBM Db2 Analytics Accelerator Configuration Console before you can update tables incrementally. The technical description of the error is: *errorXML***

Explanation

locationName
Location name identifying the Db2 subsystem to be enabled for incremental updates (replication)

errorXML
A technical description of the error that is generated by the replication automation component.

Problem determination:
When incremental updates are enabled for a Db2 subsystem and all tables have been removed from the the connected accelerator, incremental updates are disabled and the replication state changes to 'Configured'. As soon as new tables are defined on the accelerator, an attempt is made to re-enable incremental updates by using the previous configuration. This operation can fail, for example, if the capture engine of IBM InfoSphere Change Data Capture for z/OS is not started or if the password of the Db2 user who configured incremental updates for the subsystem has changed.

User action:
Re-enable incremental updates for the Db2 subsystem from the IBM Db2 Analytics Accelerator Configuration Console. No action is required if you do not want to resume the incremental update process.

AQT20018E **The replication service cannot confirm that all changes were applied to the *accelerator* accelerator. Reason: *reasonID*.**

Explanation

accelerator
Name of the affected accelerator.

reasonID
Reason why the replication service was not able to confirm

Problem determination:
The replication service cannot confirm that all data changes were applied to the accelerator. Possible reasons: 1: The user-specified timeout period has elapsed. 2: An internal error has occurred.

User action:
1) Check which of the indicated reasons applies. 2) Increase the timeout value or contact IBM support.

AQT20020E **The Db2 credentials for incremental updates are invalid. Update the Db2 credentials of user *userName* in Db2 subsystem *locationName*.**

Explanation

locationName
Location name identifying the Db2 subsystem

userName
Db2 user name that is used for incremental update processes

Problem determination:
The Db2 user name or the password of the user ID that is used by IBM Db2 Analytics Accelerator for z/OS for incremental updates is invalid or has expired.

User action

1. From the IBM Db2 Analytics Accelerator Console, update the Db2 credentials of the user that is mentioned in the error message.
2. Restart the incremental update process.

AQT20021E **A table could be loaded successfully, but a subsequent incremental update operation on this table failed. Message: *errorMessage***

Explanation

errorMessage
An error message generated by the incremental-update automation component.

Problem determination:
An incremental update operation failed after a successful load operation, probably because incremental updates were not enabled for the table. However, the table remains in LOADED state, which means that it can be used for accelerated queries.

User action:
Try to enable incremental updates for the table.

AQT20022E **Incremental updates could not be enabled for the *acceleratorName* accelerator because the following script is missing: *scriptName***

Explanation

acceleratorName
Name of the accelerator

scriptName
Name of the missing script

AQT20023E **Incremental updates could not be enabled for the *acceleratorName* accelerator because the following script returned an error: *scriptName***

Explanation

acceleratorName
Name of the accelerator

scriptName
Name of the failed script

AQT20024E **The data of partition *partition* of the *tableSchema.tableName* table cannot be archived because a restore operation is in progress for the same partition or did not run to completion. A restore operation must be completed successfully before the partition data can be archived again.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

partition
Number of the physical partition that cannot be archived

Problem determination:
You cannot archive partition data with the High Performance Storage Saver if the same data is being restored at the same time, or if a restore process for the same data has failed before it could be completed. You can run the SYSPROC.ACCEL_GET_TABLES_DETAILS stored procedure to find all archiving or restore operations that did not run to completion, and must therefore be fixed.

User action

1. Rerun the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure for the indicated partition to complete the restore operation.
2. If this fails, run SYSPROC.ACCEL_GET_TABLES_DETAILS procedure to list all failed archiving and restore operations. Try to analyze and fix the error.
3. If this does not help, remove the indicated partition from the XML input list of the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure or do not select it for storage saver processing in your archiving client.

AQT20025E **The data of partition *partition* of the *tableSchema.tableName* table cannot be restored because an archiving operation is in progress for the same partition or has failed before it could be completed. it has started, but not completed, archive processing. An archiving operation must be completed successfully before the partition data can be restored.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

partition
Number of the physical partition that cannot be restored

Problem determination:
You cannot restore partition data if the same data is being archived at the same time, or if an archiving process has failed before it could be completed. You can run the SYSPROC.ACCEL_GET_TABLES_DETAILS

stored procedure to find all archiving or restore operations that did not run to completion, and must therefore be fixed.

User action

1. Rerun the SYSPROC.ACCEL_RESTORE_MOVE_TABLES stored procedure or the corresponding function in your administration client for the indicated partition to complete the archiving operation.
2. If this fails, run SYSPROC.ACCEL_GET_TABLES_DETAILS procedure to list all failed archiving and restore operations. Try to analyze and fix the error.
3. If this does not help, remove the indicated partion from the XML input list of the SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedure or do not select it for storage saver processing in your administration client.

AQT20026E **The *operation* operation cannot be carried out because the version of the stored procedure is incompatible with the current version of the accelerator. To run this operation, install version *minimumRequiredVersion* or later of the stored procedures. The version of the stored procedures must never be higher than the software version of the accelerator; the software version of the currently active accelerator is *currentAcceleratorVersion*.**

Explanation

operation
Operation that should have been completed by the stored procedure

minimumRequiredVersion
Minimum stored-procedure version that is required for compatibility with the accelerator.

currentAcceleratorVersion
Software version of the currently active accelerator

Problem determination:
Your combination of stored procedure and accelerator software versions does not support the requested operation. To archive partition data on a version 4.1.x accelerator, the stored procedures must also be upgraded to version 4.1.x.

User action:
Upgrade your stored procedures to the same program-temporary-fix (PTF) level as the accelerator.

AQT20028W **One or more archived partitions of the *tableSchema.tableName* table ending at *limitKeys* can no longer be found in Db2 for z/OS and will therefore be deleted from the accelerator. You might want to delete the following image copies of these partitions, which cannot be used anymore: *backupImageNames***

Explanation

tableSchema
Schema name of the affected tables

tableName
Name of the affected table

limitKeys
Limit keys identifying the partitions

backupImageNames
Obsolete image copies

Problem determination:
The data of one or more partitions has been archived on an accelerator before the partitions were deleted or modified in Db2 for z/OS (for example by an ALTER TABLE ROTATE PARTITION statement), so that the limit keys used for moving the data no longer identify the original Db2 data. In such a case, the archived partitions are deleted from the accelerator when the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure or the corresponding function in your administration client is called the next time for the same table. The image copies that were created when the data was archived cannot be restored anymore because the table definition has changed. You can therefore delete the corresponding data sets.

User action:
Consider deleting the specified image copies.

AQT20032W **A migration of the accelerator from version *currentVersion* (current version) to version *targetVersion* (target version) is not permitted. Reason: *description***

Explanation

targetVersion
Target version

currentVersion
Current version

description
Reason why a migration from the current version to the target version is not permitted

Problem determination:

This warning occurs when a check of IBM Db2 Analytics Accelerator catalog indicates that a migration to the target version would lead to compatibility problems. For example, a backward migration from version 4.x.x to version 3.x.x is only allowed if all archived partitions have been archived or restored completely.

User action:
The description might suggest corrective actions after which you can successfully complete the operation.

AQT20034E **The query contains *numParameterMarkers* parameter markers or special registers and therefore cannot be explained.**

Explanation

numParameterMarkers
The number of parameter markers or special registers in the query to be explained

Problem determination:
You cannot extract Db2 Explain information about a query from an accelerator if the query contains parameter markers or special registers.

User action:
Revise the query and replace the parameter markers or special registers with literals, for example dummy values.

AQT20044E **You are trying to load *numReplicatedTablesToLoad* tables that are enabled for incremental updates, but the limit is *limit*.**

Explanation

numReplicatedTablesToLoad
Number of tables to be loaded

limit
Maximum number of tables that can be loaded

Problem determination:
There is a limit on the number of tables enabled for incremental updates that can be loaded in a single load operation. This limit was exceeded, so the load operation does not begin.

User action:
Specify fewer incremental-update-enabled tables for the load operation.

AQT20045W **The table *schemaName.tableName* cannot be added to the database with the location name *locationName* because the database has been removed from the catalog.**

Explanation

locationName
Location name of the database system that is removed

schemaName
Schema name of a table to be added to the removed database

tableName
Name of a table to be added to the removed database

Problem determination:
You can only define new tables using the SYSPROC.ACCEL_ADD_TABLES stored procedure or create new accelerator-only tables if the pairing between Db2 for z/OS is intact and has not been removed or is about to be removed.

User action:
Ensure that a pairing between Db2 for z/OS and the accelerator exists. Then repeat the operation.

AQT20046W **Tables originating from the Db2 subsystem with the location name *systemLocationName* were still on the accelerator when the pairing between Db2 and the accelerator was removed. These *numTables* tables were also deleted from the accelerator.**

Explanation

systemLocationName
Location name of the Db2 subsystem that the tables originated from

numTables
Number of tables that still existed on the accelerator when it was removed

Problem determination:
This warning occurs when the command or stored procedure to remove an accelerator is executed although tables still exist on the accelerator. This might happen if a table is defined just at the time when the accelerator is being removed. Note that all such tables are removed.

User action:
No action is required. You can ignore the error messages that are issued by related processes running in parallel.

AQT20047E **The detectChanges=DATA attribute must not be specified for loading the *schemaName.tableName* table because an external loader is employed.**

Explanation

schemaName
Schema name of the table to be loaded

tableName
Name of the table to be loaded

Problem determination:
The load operation involves an external load utility and the data source might not be Db2. The Db2 run-time statistics cannot be used to determine reliably whether a partition or table has changed and must therefore be reloaded on the accelerator.

User action:
Do not specify detectChanges=#DATA#.

AQT20048E **The accelerator-only table *schemaName.tableName* can only be loaded with the append-only option.**

Explanation

schemaName
Schema name of the accelerator-only table to be loaded

tableName
Name of the accelerator-only table to be loaded

Problem determination:
Load operations on accelerator-only tables can only append new data. Existing data cannot be replaced.

User action:
Set the attribute loadAppend=#true# in the XML load specification.

AQT20049E **Task *maintenanceTaskID* of type *maintenanceTaskType* has switched the system into maintenance mode, which prevents the current task *taskID* of type *taskType* from doing the same.**

Explanation

taskID
ID of the task that requests the maintenance mode

taskType
Type of the task requests the maintenance mode

maintenanceTaskID
ID of the task that currently works in maintenance mode

maintenanceTaskType
Type of the task that currently works in maintenance mode

Problem determination:
Another task has already reserved the maintenance mode for the system. This mode cannot be given to more than one task at the same time.

User action

1. Wait until the other maintenance task has finished.
2. Retry the operation.

AQT20050E **Task *taskID* of type *taskType* cannot switch the system into maintenance mode within *timeOut* ms because the following other tasks are still running: *blockingTasks*.**

Explanation

taskID
ID of the task that requests the maintenance mode

taskType
Type of the task that requests the maintenance mode

timeOut
Timespan required to reach the desired state (in ms)

blockingTasks
List of the tasks that block the switch into maintenance mode

Problem determination:
The system cannot be switched into maintenance mode within the given timespan because at least one other task exists that claims the maintenance mode for itself.

User action:
1. Wait until the other tasks have finished, or cancel those tasks. 2. Retry the operation.

AQT20051E **The *columnName* column of the *tableSchema.tableName* table was dropped from the Db2 subsystem after the table was defined on the accelerator.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

columnName
Name of the affected table column

Problem determination:
This exception occurs when a column dropped from a table after the table was defined on an accelerator.

User action:
1. Remove the table from the accelerator. 2. Redefine the table on the accelerator. 3. Rerun the stored procedure that returned the error.

AQT20052E **The *tableSchema.tableName* table cannot be defined on the accelerator because the row length exceeds the limit of *rowLengthLimit* bytes.**

Explanation

tableSchema
Schema name of the table that could not be created in the database on the accelerator

tableName
Name of the table that could not be created in the database on the accelerator

rowLengthLimit
Maximum row length in bytes permitted by the database on the accelerator

Problem determination:
The rows of the table that you want to define are longer than the allowed maximum (in bytes).

User action:
Change the columns widths of the table in Db2 for z/OS so that the resulting row length fits the allowed maximum.

AQT20053E **The *tableSchema.tableName* table on the '*acceleratorName*' accelerator uses the unsupported CCSID *ccsid*.**

Explanation

acceleratorName
Name of the accelerator

tableSchema
Schema name of the table that uses an unsupported CCSID

tableName
Name of the table that uses an unsupported CCSID

ccsid
The unsupported CCSID

Problem determination:
The specified single-byte coded character set identifier (CCSID) for the table is not supported by IBM Db2 Analytics Accelerator for z/OS. Therefore, you cannot define the table on the accelerator.

User action:
Use a supported CCSID.

AQT20054W **The load process for the *tableSchema.tableName* table detected invalid bytes sequences in the load data. This affected *numberDiscardedRows* rows, which have been discarded during the process.**

Explanation

tableSchema
Schema name of the table that contains the invalid UTF-8 characters

tableName
Name of the table that contains the invalid UTF-8 character

numberDiscardedRows
Number of discarded rows

Problem determination:
The load process for the specified table detected invalid bytes sequences (like invalid UTF-8 characters) in the load data. The load process was completed, but the rows containing the invalid characters were discarded. Be aware that the table data on the accelerator now differs from the data in the corresponding Db2 table. Therefore, the results of a query can vary, depending on where it is executed (accelerator or Db2 for z/OS).

User action:
1. Replace the invalid characters in the original Db2 table with valid ones. 2. Reload the corresponding accelerator table. 3. Review processes or applications programs that might have inserted the invalid bytes sequences. IBM can help you identify the numbers of the discarded rows in the accelerator trace information.

AQT20055W **The numbers of distinct values in each column of the *schemaName.tableName* table, which are required for the selection of an organizing key, could not be determined.**

Explanation

schemaName
Schema name of the table that was supposed to provide the organizing key

tableName
Name of the table that was supposed to provide the organizing key

Problem determination:
The number of distinct values in a table column is the decisive factor for its suitability as an organizing key. It was not possible to determine this number for each table column. An organizing key was therefore not created.

User action:
1. Run the Db2 utility RUNSTATS to generate up-to-date statistics for the estimated number of distinct values in each column, also known as COLCARDF. 2. Disable incremental updates for the table and then re-enable these. 3. Retry the load operation.

AQT20056E The call-home function could not be enabled because an internal error occurred.

User action:
Contact IBM support.

AQT20057E The call-home function could not be disabled because an internal error occurred.

User action:
Contact IBM support.

AQT20060W Entries were excluded from the query history because the QUERY_LIST output parameter of the SYSPROC.ACCEL_GET_QUERIES stored procedure cannot contain more than *maxMessageSize* bytes of data. This maximum was exceeded. Therefore, only *retrievedNumSQLStatementEntries* entries out of *requestedNumSQLStatementHistoryEntries* have been included in the retrieved query history.

Explanation

maxMessageSize
Maximum allowed size for the XML document of the SQL statement history (in bytes)

requestedNumSQLStatementHistoryEntries
Requested (total) number of SQL statement history entries

retrievedNumSQLStatementEntries
Number of actually retrieved entries of the SQL statement history

Problem determination:
The maximum size of the output list was exceeded, and therefore, some entries had to be excluded from the query history. Entries that show up at the top are preserved, and later entries that exceeds the limit are included.

User action:
Use a more restrictive filter so that the number of output entries fits the allowed maximum size. Change the filter by altering the values in the 'query_selection' parameter of the stored procedure, or the settings of the View and Show drop-down lists in the query history of your administration client.

AQT20061E The load specification specifies the XML attributes `detectChanges=`*`detectChanges`* and `forceFullReload=false` for the incremental-update-enabled table *schemaName.tableName*. This is not permitted.

Explanation

schemaName
Schema name of the table

tableName
Name of the table

detectChanges
Explicit or default setting for the `detectChanges` XML attribute

Problem determination:
The XML load specification for incremental-update-enabled tables must set the attributes for change detection or a full-table reload to guarantee that updates cannot be lost during the load on those partitions that are not being loaded.

User action:
Repeat the load operation with `detectChanges="DATA"` or `forceFullReload=true` in the XML load specification for the table.

AQT20062E A procedure package with the name *packageName* already exists. Its version is *existingVersion*. Another procedure package with the same name (and version *newVersion*) cannot be added.

Explanation

packageName
Name of the procedure package

existingVersion
Version of the already existing package

newVersion
Version of the package to be added

Problem determination:
A procedure package with the given name already exists on the accelerator. Therefore, you cannot add a package with the same name.

User action

1. Remove the existing package.

2. Add the new package.

AQT20063E A procedure package with the name *packageName* does not exists.

Explanation

packageName
Name of the procedure package

Problem determination:
A procedure package with the given name does not exists on the accelerator. Therefore, you cannot refer to it.

User action:
Correct the package name or install the package.

AQT20064E A procedure package with the name *packageName* has already been enabled for the Db2 subsystem with the location name *locationName*. It cannot be enabled again without disabling it before.

Explanation

packageName
Name of the procedure package

locationName
Location name of the Db2 subsystem

Problem determination:
A procedure package with the given name has already been enabled for the Db2 subsystem. Therefore, you cannot enable it again.

User action:
Change the name of the procedure package or disable it before you enable it again.

AQT20067E The *clientProcedureName* stored procedure cannot be executed because a package that contains this stored procedure was not installed on the accelerator.

Explanation

clientProcedureName
Name of the client stored procedure to be executed

Problem determination:
Each stored procedure to be executed on the accelerator must be installed from a procedure package. The given stored procedure was not installed in this way.

User action:
1. Install the procedure package that contains this stored procedure on the accelerator. 2. Try again to execute the stored procedure.

AQT20068E The *clientProcedureName* stored procedure cannot be executed because the *packageName* procedure package was not enabled for a Db2 subsystem with the name *locationName*.

Explanation

clientProcedureName
Name of the client stored procedure to be executed

packageName
Name of the procedure package

locationName
Location name of the Db2 subsystem

Problem determination:
A stored procedure can only process tables that come from the Db2 subsystem for which the stored procedure has been enablend. This is not the case for the given stored procedure and the package that it was delivered in.

User action:
1. Enable the procedure package for the above-mentioned Db2 subsystem. 2. Try again to execute the stored procedure.

AQT20069E The *clientProcedureName* stored procedure from the *packageName* package failed on the accelerator. Error message: *errorMessage*.

Explanation

clientProcedureName
Name of the client stored procedure to be executed

packageName
Name of the procedure package

errorMessage
Error message

Problem determination:
A problem was encountered during the execution of the stored procedure. The error message contains details about this problem.

User action:
1. Eliminate the cause of the error. Contact the vendor of the procedure package if necessary. 2. Try again to execute the stored procedure.

AQT20070E The execution of the *clientProcedureName* stored

procedure from the *packageName* package failed because the metadata in the XML document could not be validated and processed. The following validation error was reported during the execution: *errorMessage* in metadata of document *xmlDocument*.

Explanation

clientProcedureName
Name of the client stored procedure to be executed

packageName
Name of the procedure package

errorMessage
Error message that was issued after validating the XML document that was generated by the DESCRIBE procedure

xmlDocument
XML document that was generated by the DESCRIBE procedure

Problem determination:
The stored procedure could not be executed because an error ocurrred as the accelerator tried to validate the metadata that it received from the procedure package.

User action:
Contact the vendor of the procedure package.

AQT20071E **Running the stored procedure with the name *clientProcedureName* from procedure package *packageName* created the accelerator-only table *schemaName*.*tableName*. The *columnName* column of that table uses the unsupported data type *dataType*.**

Explanation

clientProcedureName
Name of the client stored procedure to be executed

packageName
Name of the procedure package

schemaName
Schema name of the accelerator-born table

tableName
Name of the accelerator-born table

columnName
Name of the column in the accelerator-born table

dataType
Data type of the column

Problem determination:
The execution of the stored procedure on an accelerator led to the creation of a new table, which contains a column with an unsupported data type. Therefore, all actions that were taken by the stored procedure are undone.

User action:
Contact the vendor of the procedure package.

AQT20072E **A load of the *tableSchema*.*tableName* table requires Netezza Performance Server (NPS®) version *requiredNPSVersion*.**

Explanation

tableSchema
Schema name of the table that contains invalid UTF-8 characters

tableName
Name of the table that contains invalid UTF-8 characters

requiredNPSVersion
Required version of the Netezza Performance Server (NPS) software

Problem determination:
A table contains UTF-8 characters that were not supported until recent. To be able to process these characters, you need the indicated NPS version or an even higher version.

User action:
Upgrade your NPS software on the accelerator.

AQT20081E **Operation *operation*, which was supposed to transfer *numBytes* bytes on TCP/IP socket *socket*, failed because the connection to the remote machine at IP address *remoteIPAddress* and port *remotePort* has been closed unexpectedly. Reason: *errorNumber*.**

Explanation

socket
Socket number

operation
Operation that failed as a result of the closed connection

numBytes
Number of bytes to be processed by the operation

remoteIPAddress
IP address of the remote machine

remotePort
Port number of the remote machine

errorNumber
Error number

Problem determination:
The TCP/IP operation failed because the remote machine closed the connection unexpectedly. The cause is probably a connection problem or an internal protocol error.

User action:
Verify the proper functioning of the network connection. If the network connection is OK, contact IBM support.

AQT20082E **Operation *operation*, which was supposed to transfer *numBytes* bytes on TCP/IP socket *socket*, failed because the connection to the remote machine at IP address *remoteIPAddress* and port *remotePort* has been closed by a regular shutdown. *numAlreadyProcessedBytes* bytes could be processed until the shutdown.**

Explanation

socket
Socket number

operation
Operation that failed as a result of the closed connection

numBytes
Number of bytes to be processed by the operation

remoteIPAddress
IP address of the remote machine

remotePort
Port number of the remote machine

numAlreadyProcessedBytes
Number of bytes that could be sent and received

Problem determination:
The TCP/IP operation failed because the connection was shut down. The cause is probably a connection problem or an internal protocol error.

User action:
Verify the proper functioning of the network connection. If the network connection is OK, contact IBM support.

AQT20083E **Operation *operation*, which was supposed to transfer *numBytes* bytes on TCP/IP socket *socket* to the remote machine at IP address *remoteIPAddress* and port *remotePort* could not be completed within the time allotted (*timeoutInSeconds* seconds). *numAlreadyProcessedBytes* bytes could be processed until the connection broke off.**

Explanation

socket
Socket number

operation
Operation that failed as a result of the closed connection

numBytes
Number of bytes to be processed by the operation

remoteIPAddress
IP address of the remote machine

remotePort
Port number of the remote machine

numAlreadyProcessedBytes
Number of bytes that could be sent and received

timeoutInSeconds
Timeout period that was exceeded

Problem determination:
The TCP/IP operation could not be completed within the time allotted. The cause is probably a connection problem or an internal protocol error.

User action:
Verify the proper functioning of the network connection. If the network connection is OK, contact IBM support.

AQT20084E **Operation *operation*, which was supposed to transfer *numBytes* bytes on TCP/IP socket *socket* to the remote machine at IP address *remoteIPAddress* and port *remotePort*, was interrupted. *numAlreadyProcessedBytes* bytes could be processed until the interruption.**

Explanation

socket
Socket number

operation
Operation that was interrupted

numBytes
Number of bytes to be processed by the operation

remoteIPAddress
IP address of the remote machine

remotePort
Port number of the remote machine

numAlreadyProcessedBytes
Number of bytes that could be sent and received

Problem determination:
The TCP/IP operation was interrupted, for example because the task was canceled.

AQT20085E **Socket *socket* could not be connected to the remote machine at IP address *remoteIPAddress* and port *remotePort*. Reason: *error*.**

Explanation

socket
Socket number

remoteIPAddress
IP address of the remote machine

remotePort
Port number of the remote machine

error
Error message indicating the reason for the failure; (error number mapped to a string)

Problem determination:
A TCP/IP connection could not be established. The error message explains why.

User action:
Correct the problem and retry the operation.

AQT20086E **The communication protocol flow was broken by the *peerType* because an error happened at the end of the network communication.**

Explanation

peerType
Cause of the communication failure

Problem determination:
This internal error message is accompanied by other messages indicating the cause of the error

User action:
Check the accompanying messages to find the cause of the error and resolve the problem.

AQT20087E **The communication protocol flow is broken because the unexpected network message *receivedNetworkMessage* was received although message *expectedNetworkMessage* was expected.**

Explanation

expectedNetworkMessage
Expected network message

receivedNetworkMessage
Received network message

Problem determination:
This internal error points to a protocol violation between the communicating machines.

User action:
Contact IBM support.

AQT20088W **An internal warning occurred on the '*acceleratorName*' accelerator: *diagnosticMessage***

Explanation

acceleratorName
Name of the accelerator

diagnosticMessage
Diagnostic information returned by the accelerator

Problem determination:

AQT20089E **A distributed commit between the stored procedure and the server failed for the following reason: *diagnosticMessage***

Explanation

diagnosticMessage
Reason why the distributed commit failed

Problem determination:
A distributed commit between the stored procedure and the server failed for the indicated reason.

User action:
Contact IBM support.

AQT20090E **Enter a location name. The location name must not be an empty string.**

Problem determination:
This error occurs if you do not specify a location name (the location name is an empty string).

User action:
Try to add the accelerator again. This time, provide a location name.

AQT20091W **The Db2 subsystem or data sharing group with the location name *removedTimeReferenceSystemLoca***

***tionName*, which acted as the time reference system for the accelerator, has been removed. The new time reference system is the Db2 subsystem or data sharing group with the location name *newReferenceTimeLocationName*.**

Explanation

newReferenceTimeLocationName
Location name of the new time reference system

removedTimeReferenceSystemLocationName
Location name of the previous time reference system

Problem determination:
This warning occurs when the time reference system changes.

User action:
No action necessary

AQT20092E **The load request failed because the following error occurred: *reasonText***

Explanation

reasonText
Root exception message

Problem determination:
This error occurs if a load request fails and no tables were loaded with data.

User action:
Run the load job again. If the problem persists, contact IBM support.

AQT20093E **A table named *tableSchema*.*tableName* cannot be found in the Db2 subsystem or data sharing group with the location name *locationName*.**

Explanation

locationName
Location name of the Db2 subsystem or data sharing group that is supposed to contain the table

tableSchema
Schema name of the table that cannot be found

tableName
Name of the table that cannot be found

AQT20094E **The option *invalidContentOption* was specified in a request to save trace data, but that option is unknown.**

Explanation

invalidContentOption
Content flag that is not compatible with others in the list

Problem determination:
The request to save trace data cannot be processed because an unknown content option (*invalidContentOption*) was used.

User action:
Remove the invalid option *invalidContentOption* and repeat the operation.

AQT20095E **You specified the option *invalidContentOption* in a request to save trace data, but that option is not compatible with other options in *contentOptionList*.**

Explanation

invalidContentOption
Incompatible content option flag

contentOptionList
List of all content options used

Problem determination:
Restrictions apply when you specify a combination of content options in a request to save trace data. One of these restrictions was violated when you specified option *invalidContentOption* together with other options in the list (*contentOptionList*).

User action:
Remove the option *invalidContentOption*. Use this option individually in a separate request to save trace data, so that it cannot conflict with the other options.

AQT20096E **EXPLAIN information about this query could not be obtained. Diagnostic information: *exceptionText***

Explanation

exceptionText
Detailed exception text returned by the EXPLAIN service

Problem determination:
The inability to retrieve EXPLAIN information can have a number of reasons. The database system can be offline, the SQL statement might be invalid, or a table, view or column referenced by the query no longer exists on the accelerator.

User action

1. Verify that the accelerator is online.

2. Verify that all tables referenced by the query still exist on the accelerator.
3. Review the diagnostic information in this message to diagnose the problem.

AQT20097E **The component responsible for managing incremental updates did not finish the execution of a command within the timeout period.**

Problem determination:
This internal error most likely indicates a temporary overload situation on the accelerator.

User action:
Check the states of your incrementally updated tables. If tables are in unexpected states, take action to change the table states, so that the operation can be executed successfully.

AQT20098E **The request-issuing component did not receive a reply within the timeout period.**

Problem determination:
This internal error most likely indicates a temporary overload situation on the accelerator.

User action:
Check the overall system status. Take compensating action if a system component is in an unexpected state.

AQT20099E **An interaction between two components failed due to the following reason: *diagnosticMessage***

Explanation

diagnosticMessage
Reason why the interaction failed

Problem determination:
An interaction between two components failed for reasons indicated in the diagnostic text.

User action:
Contact IBM support.

AQT20100E **The name of an already existing file (*fileName*) was specified in the `SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage` command. The file already exists in the target directory on the accelerator.**

Explanation

fileName
Name of the already existing file

Problem determination:
The name of a file to be transferred with the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage`** command must be unique and the file must not yet exist in the transfer directory on the accelerator.

User action:
You might cancel the operation because you need not deploy the same file twice. If you want to redeploy the same file, rename it in the source directory or remove the existing file from the accelerator by using the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 removeDeployedPackages`** command before you transfer it to the accelerator.

AQT20101E **The file integrity could not be verified for the *fileName* file. The file was transferred using the `SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage` command. The hash value *callerHash* that was provided does not match the hash value computed on the accelerator. Checksum output of the accelerator: *computedHash***

Explanation

fileName
Name of the file that failed the checksum integrity test

callerHash
The provided checksum value

computedHash
The checksum computed on the accelerator

Problem determination:
When a file is transferred to an accelerator by the SYSPROC.ACCEL_SOFTWARE_UPDATE2 stored procedure, a checksum is computed to verify the integrity of the file. This checksum does not match the checksum provided by you or the calling application.

User action:
Retry the file transfer. If the error persists, check the validity of the file in the source source directory. The checksum file might contain an invalid checksum. For example, verify that a binary option was used when the file was downloaded to the source location. Also verify that a valid `checksum*.txt` file was placed in the source directory.

AQT20102W **The file *fileName* could not be removed because it was not found.**

Explanation

fileName
File name of the package to be removed

Problem determination:
You tried to remove a package file, but that file was not found. Either the specified file name is incorrect, or the file has already been removed by another command that was issued around the same time.

User action

Check the file name. If the file name was correct, but the file does no longer exist, ignore this warning.

Tip: To retrieve the list of files that can be removed, run this command:**`CALL SYSPROC.ACCEL_UPDATE_SOFTWARE2(listDeployedPackages)`**

AQT20103W **The file *fileName* could not be extracted. Diagnostic information: *diagnostics***

Explanation

fileName
Name of the package to be deployed

diagnostics
Diagnostic information

Problem determination:
You tried to transfer a package file, but the file could not be extracted. The file has been saved on the accelerator, but can not be installed by the software update function.

User action:
Make sure that you have used a valid software update file. Remove the transferred file from the accelerator and retry, or open a support case to help you diagnose the problem, so that you can install the update manually. Tip: To remove the file from the accelerator, run this command: **`CALL SYSPROC.ACCEL_UPDATE_SOFTWARE2(removeDeployedPackages)`**

AQT20104E **A unique constraint, such as a primary key or unique index, is not defined for the *tableSchema.tableName* table. During an insertion, data consolidation might merge multiple rows into just one row, or the deletion of one row might cause the deletion of multiple rows from the accelerator table. Incorrect query results will be the consequence. Another problem with tables of this type is that queries and incremental updates take much longer. That's why it is not allowed to enable this table for incremental updates.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
Without a unique constraint, all columns of a table are used to identify a single table row. There might be multiple rows with exactly the same column values. In this case, inserting the same row multiple times results in the insertion of just one row at the target. In addition, the deletion of one row from a Db2 for z/OS table might entail the deletion of all rows with the same values from the corresponding accelerator-shadow table because duplicate rows cannot be differentiated. This leaves you with inconsistent tables, which inevitably leads to incorrect query results from the accelerator. The absence of a unique constraint also means that the identification of table rows takes longer because the entire table must be scanned. For large tables, this might result in a poor query and update performance. That's why it is not allowed to enable this table for incremental updates.

User action

To add a unique constraint to the table, follow these steps:

1. Remove the table from the accelerator.
2. In Db2, define a unique constraint.
3. Redefine the table on the accelerator.
4. Enable incremental updates for the redefined table.
5. Load the redefined table.

AQT20105E **A stored procedure or sub-function of a procedure has been called that is not supported for this type of accelerator: *unsupportedFunction***

Explanation

unsupportedFunction
Name or description of the unsupported procedure or function

Problem determination:
The product supports different deployment modes and platforms with a uniform application programming interface (API). Some functions, however, are only supported in specific deployment modes or on specific platforms. This exception occurs if you run a function that is not supported in the current deployment mode or on the current platform of this accelerator.

User action:

The function is not supported and cannot be used for this accelerator. Check the documentation for the supported functions in this deployment mode or on this platform.

AQT20106E **An invalid file name was specified in the `SYSPROC.ACCEL_SOFTWARE_UPDATE2 activateDeployedPackage` command. File Name: *fileName*.**

Explanation

fileName
Name of the already existing file

Problem determination:
The SYSPROC.ACCEL_SOFTWARE_UPDATE2 stored procedure can transfer any file to an accelerator, but only files that have been identified as valid software updates can be installed. The file name that you specified was not identified as a valid software update package.

User action:
Use the **`SYSPROC.ACCEL_SOFTWARE_UDPATE2 listDeployedPackages`** command to display a list of software packages that can be activated. If you have not yet transferred a valid software update package to the accelerator, use the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 deployPackage`** command. Some packages must be installed manually. For these, refer to the instructions that came with the package or contact IBM support.

AQT20107E **The component responsible for software updates did not finish the execution of the command *command* within the timeout period of *timeout* milliseconds.**

Explanation

command
Unfinished software update command

timeout
Maximum wait time in seconds

Problem determination:
The **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 activateDeployedPackage`** was invoked to install new software on the accelerator. The operation did not finish within the timeout period. This internal error most likely indicates a temporary overload situation on the accelerator.

User action:
This error indicates a temporary problem. Retry the operation. If the problem persists, contact IBM support.

AQT20108E **The `SYSPROC.ACCEL_SOFTWARE_UPDATE2 activateDeployedPackage` command failed for product version *version*. The accelerator is not ready for the installation. The prerequisite check failed with the following messages: *message* Diagnostic information 1: *stdout* Diagnostic information 2: *stderr***

Explanation

version
The software version that was specified in the **`activateDeployedPackage`** command.

message
Additional description of the error

stdout
Diagnostic information

stderr
Diagnostic information

Problem determination:
The **`SYSPROC.ACCEL_SOFTWARE_UPDATE2 activateDeployedPackage`** command was invoked to install new software on the accelerator. Before the software is installed, the status of the accelerator and the necessary installation prerequisites are checked. This check failed.

User action:
If the error information indicates a temporary problem, retry the **`SYSPROC.ACCEL_SOFTWARE_UPDATE2`** command. If the problem persists, contact IBM support.

AQT20109E **The `SYSPROC.ACCEL_CONTROL_ACCELERATOR getActivationLog` command failed. An installation log was not found. Diagnostic message: *message***

Explanation

message
Additional description of the error

Problem determination:
The **`SYSPROC.ACCEL_CONTROL_ACCELERATOR getActivationLog`** command was invoked to retrieve the latest information from the installation log, but an installation log was not found.

User action:
If you have never installed a software update before, this error is expected. If you have already installed or

tried to install software packages, but still run into this error, contact IBM support.

AQT20121E **The load request failed because the following error occurred: *reasonText***

Explanation

reasonText
Root exception message

Problem determination:
This error occurs if a load request fails, and no tables were loaded with data.

User action

1. Run the load job again.
2. If the problem persists, contact IBM support.

AQT20127E **The request-issuing component did not receive a reply within the timeout period.**

Problem determination:
This internal error most likely indicates a temporary overload situation on the accelerator.

User action:
Check the overall system status. Take compensating action if a system component is in an unexpected state.

AQT20133E **The DATA CAPTURE CHANGES option is not defined for the *tableSchema*.*tableName* table.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
If DATA CAPTURE CHANGES is not specified for the table, no detailed log data is written. Detailed log data is required if you want to update a table incrementally. You can set the DATA CAPTURE CHANGES option when you create a table in Db2 for z/OS. It is also possible to add this option later by submitting an ALTER TABLE statement. The DATA CAPTURE CHANGES option is mandatory if you want to enable incremental updates for a table.

User action

To change the DATA CAPTURE logging option, follow these steps:

1. Remove the table from the accelerator.
2. In Db2, submit an ALTER TABLE statement to set the DATA CAPTURE CHANGES option for the table. Alternatively, drop the table and recreate it with DATA CAPTURE CHANGES enabled.
3. Redefine the table on the accelerator.
4. Enable incremental updates for the redefined table.
5. Load the redefined table.

AQT20134E **The `<waitForReplication/>` element is not a valid option in your particular setup because the *accelerator* accelerator is not configured for incremental updates with IBM InfoSphere Change Data Capture for z/OS.**

Explanation

accelerator
Name of the affected accelerator

Problem determination:
The `<waitForReplication/>` element only works for incremental updates in connection with IBM InfoSphere Change Data Capture for z/OS. Your accelerator is not configured in this way. It might be configured for integrated synchronization, which does not allow the `<waitForReplication/>` element as a command parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

User action

- If your system is configured to use integrated synchronization for incremental updates, consider WAITFORDATA processing. WAITFORDATA processing is the superior option when compared with `<waitForReplication/>`. Also note that the use of `<waitForReplication/>` is deprecated.
- If incremental updates are not configured at all, you are free to choose between IBM InfoSphere Change Data Capture for z/OS and integrated synchronization. In a setup with IBM InfoSphere Change Data Capture for z/OS, the `<waitForReplication/>` element will be accepted. However, the setup is quite complex, and since WAITFORDATA processing exists, you do not need `<waitForReplication/>`. Therefore, consider a setup that uses integrated synchronization.

AQT20135E **The software can only be installed on the accelerator if incremental updates have been stopped for all connected Db2 subsystems.**

Problem determination:

You tried to install new software on the accelerator by running the SYSPROC.ACCEL_SOFTWARE_UPDATE2 stored procedure in connection with the **`<activateDeployedPackage/>`** command. The operation failed because it is checked before the installation whether incremental updates have been stopped for all connected Db2 subsystems. This was not the case.

User action

1. Use either of these approaches:
   - In your administration client:
     a. Check your database connection profiles to find the Db2 subsystems that have been paired with the accelerator you want to update.
     b. Open the **Accelerator** view of each of these profiles and make sure that the **Replication status** is *Stopped*. If not, click the **Stop** link.
   - Run the SYSPROC.ACCEL_CONTROL_ACCELERATOR with the **`<stopReplication/>`** command for each connected Db2 subsystem.
2. Retry the software update.

AQT20136E **Invalid parameter specification. Contact IBM support. '*configParameter*' (*errorDescription*).**

Explanation

configParameter
Configuration parameter that caused the error

errorDescription
Description of the error

Problem determination:
This error occurs if the configuration file of the accelerator server contains an invalid setting.

User action:
Contact IBM support.

AQT20137I **The collection of statistics on the *tableType* data of the table *tableSchema.tableName* was completed successfully in *timeInSeconds* seconds.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

tableType
Flag that indicates whether the table data was archived. Possible values are "archived" or "non-archived".

timeInSeconds
Time that was needed for the statistics collection

Problem determination:
The SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure returns this message for each successful statistics collection on a table in the input table set (`table_statistics_specification`).

AQT20138E **Collecting table statistics for the *tableType* data of table *tableSchema.tableName* failed due to: *reasonText*.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

tableType
Identification whether this is the "archived" or the "non-archived" data

reasonText
Reason describing the failure

Problem determination:
Collecting table statistics failed.

User action:
If the error information indicates a temporary problem, retry the SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure. If the problem persists, contact IBM support.

AQT20139E **A statistics collection on this accelerator is not permitted because IBM InfoSphere Change Data Capture for z/OS (CDC) is the selected replication technology. The only supported replication technology is IBM Integrated Synchronization.**

Explanation

acceleratorName
Name of the accelerator

Problem determination:
Table statistics cannot be collected by the SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure if CDC is the configured replication technology.

User action:
Use IBM Integrated Synchronization as the replication technology.

AQT20140E **Statistics cannot be collected on the *tableSchema*.*tableName* table because it is a reference table for federated access.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
It is not possible to run the SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure on this type of table.

User action:
Collect the statistics directly on the accelerator-shadow table that the reference table points to.

AQT20141W **Statistics cannot be collected on the *tableSchema*.*tableName* table because the table is neither loaded, nor does it contain archive data.**

Explanation

tableSchema
Schema name of the affected table

tableName
Name of the affected table

Problem determination:
The table does not contain any data that the SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS stored procedure could collect statistics on.

User action:
Load the table before you collect table statistics.

AQT20144E **Optimization profile configuration operations cannot be completed because the optimization profile catalog table does not exist.**

Problem determination:
The optimization profile catalog table is required if you want to use optimization profiles.

User action:
Contact IBM support.

Chapter 5. Troubleshooting

In the following sections, you find descriptions of known IBM Db2 Analytics Accelerator for z/OS problems. The author and the development team have tried to provide a solution wherever possible. However, a solution might be unavailable because the cause of a problem cannot be clearly identified. This is mostly the case if multiple causes can lead to the same symptom. It can also be that a solution has not yet been found. In such cases, contact IBM support.

Important: During a customer session with IBM support, system parameters might be changed. For the parameter changes to take effect, a restart of an accelerator is often required. In such situations, the accelerator will be unavailable for operational tasks until it has come back online.

NOT Programming Interface information

SYSPROC.ACCEL_ARCHIVE_TABLES returns AQT10301E with DSNU016I

The SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure returns error AQT10301E with an embedded message (code DSNU016I).

Symptoms

The full message is similar to the following example:

```
AQT10301E - An error was encountered in the SYSPROC.DSNUTILU service of
your database management system (DBMS). The return code is 319488.
The reason code is 0. Additional diagnostic information:
'1DSNU000I    229 10:38:10.78
DSNUGUTC - OUTPUT START FOR UTILITY, UTILID = AQT000000010016 DSNU1045I
   229 10:38:10.81
DSNUGTIS - PROCESSING SYSIN AS UNICODE UTF-8 0DSNU050I
   229 10:38:10.81
DSNUGUTC -  TEMPLATE CD DSN NTCH0REG.LW0A.TPCH30.HPSSLI30.P0016 DSNU1035I
   229 10:38:10.81
DSNUJTDR - TEMPLATE STATEMENT PROCESSED SUCCESSFULLY 0DSNU050I
   229 10:38:10.81
DSNUGUTC -  COPY TABLESPACE "TPCH30"."HPSSLI30" DSNUM 16 SHRLEVEL
   REFERENCE   COPYDDN CD DSNU1038I   229 10:38:10.92
DSNUGDYN - DATASET ALLOCATED.  TEMPLATE=CD   DDNAME=SYS00001
DSN=NTCH0REG.LW0A.TPCH30.HPSSLI30.P0016 -DSNU016I
   229 10:38:13.03
DSNUGSAT - UTILITY BATCH MEMORY EXECUTION ABENDED, REASON=X'0B37'-DSNU016I
   229 10:38:14.10
DSNUGBAC - UTILITY BATCH MEMORY EXECUTION ABENDED, REASON=X'00E40347''
```

Causes

If the COPY utility is invoked by the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure and the procedure ends with this error, the configuration of the z/OS feature **DFSMSdfp** might be incorrect.

Resolving the problem

1. Start the **DFSMSdfp** feature in z/OS. For more information, follow the link under **Related information**.
2. Check whether the prefix for image copies (value of the AQT_ARCHIVECOPY_HLQ environment variable) is mapped to the correct data class. For more information, refer to *Moving partition or table data with the High-Performance Storage Saver* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*.
3. Check whether the space on the corresponding device is sufficient.

Related information

IBM DB2 Analytics Accelerator for z/OS: User's Guide

No CEEDUMPs

You cannot find a CEEDUMP although a signal was caught while you were running an IBM Db2 Analytics Accelerator for z/OS stored procedure.

Symptoms

A signal was caught while running an IBM Db2 Analytics Accelerator for z/OS stored procedure, but a CEEDUMP has apparently not been produced.

Causes

Missing configuration settings.

Resolving the problem

To analyze problems that cause the stored procedure to end abnormally or stop with a signal, a CEEDUMP is required. The JCL startup procedure for the Workload Manager (WLM) application environment of the IBM Db2 Analytics Accelerator for z/OS stored procedure allows you to specify a location for a CEEDUMP.

Important: Make sure that the JCL startup procedure for the Workload Manager (WLM) application environment of the IBM Db2 Analytics Accelerator for z/OS stored procedure contains a `DD` statement for `CEEDUMP`.

SQL code -471 referring to a function in the DSNAQT schema

You receive a message with `SQLCODE= -471` and reason code `00E79002` when trying to run the internal function DSNAQT.ACCEL_READFILE3.

Symptoms

You see an error message that is similar to the following:

```
DSNT408I SQLCODE = -471, ERROR: INVOCATION OF FUNCTION OR PROCEDURE
DSNAQT.ACCEL_LIST_SOFTWARE FAILED DUE TO REASON 00E79002
DSNT418I SQLSTATE = 55023 SQLSTATE RETURN CODE
DSNT415I SQLERRP = DSNX9GPL SQL PROCEDURE DETECTING ERROR
DSNT416I SQLERRD = -40 0 0 -1 0 0 SQL DIAGNOSTIC INFORMATION
DSNT416I SQLERRD = X'FFFFFFD8' X'00000000' X'00000000' X'FFFFFFFF'
X'00000000' X'00000000' SQL DIAGNOSTIC INFORMATION
```

Causes

In addition to externally published stored procedures in the SYSPROC schema, IBM Db2 Analytics Accelerator for z/OS employs user-defined functions internally. These functions are included in the DSNAQT schema and serve the following purposes:

- Reading temporary trace files from IBM Db2 Analytics Accelerator for z/OS stored procedures (function DSNAQT.ACCEL_READFILE3)
- Checking available stored-procedure interface versions (function DSNAQT.ACCEL_GETVERSION)

If such a function was stopped for some reason (for example by a database administrator), the end-user who has triggered the execution of the internal function sees a Db2 error `-471 00E79002`.

Resolving the problem

1. From your Db2 subsystem, check the function status by using the following command:

   ```
   -DIS FUNCTION SPECIFIC (DSNAQT.*)
   ```

2. If the function is not in the state STARTED, start it with the following command:

   ```
   -STA FUNCTION SPECIFIC (DSNAQT.*)
   ```

The trace information that was supposed to be delivered to the caller of the stored procedure in a Db2 result set has been kept in its temporary location (`/tmp` by default). Thus you can still transfer this information after restarting the stopped function.

Finding the trace file after a stored procedure failure

Read how you can find the incomplete trace file after a stored procedure failure. Generally, if a stored procedure ends unexpectedly, the trace file or trace information is not returned to your administration client.

Symptoms

Trace information that would help you analyze the failure of the stored procedure is missing.

Causes

The interrupted stored procedure run has not reached the stage where the trace information or trace file is returned to your administration client.

Environment

Trace information for each stored procedure call is stored in the `/tmp` folder of the z/OS UNIX file system (zFS) file unless a different location has been specified. When the stored procedure run is completed successfully, the trace file is transferred to your administration client, from where it can be accessed by the user who has started the function that resulted in the stored procedure call. After the transfer, the trace file in the `/tmp` folder is deleted.

The naming schema for stored procedure trace files is `aqt-trace-<date>-<time>-<sp_name>-`XXXXXX, where:

```
aqt-trace-20171113-1028-ACCEL_TEST_CONNECTION-ghbHxQ
```

<date>
: The date on which the trace file was written, in the format `YYYYMMDD`.

<time>
: The time at which the trace file was written, in the format `hhmm`.

<sp-name>
: The name of the stored procedure that the trace information is about.

XXXXXX
: A unique identifier consisting of numbers and characters

Example:

```
aqt-trace-20171113-1028-ACCEL_TEST_CONNECTION-ghbHxQ
```

Note: In the `/tmp` folder, you find all incomplete trace files. These can be the result of erroneous and currently active stored procedure runs.

Resolving the problem

1. Make a note of the time stamp that is shown when IBM Db2 Analytics Accelerator reports the failure of the stored procedure, for example:

   ```
   20170216-1457
   ```

2. Connect to the zFS of z/OS by using a command-line FTP client.
3. After the logon, change to the `/tmp` folder:

   ```
   cd /tmp
   ```

4. Enter `ls -l` to show all files in this folder.
5. Look for a stored procedure trace file with a time stamp similar to the one you took down, for example:

   ```
   aqt-trace-20171113-1028-ACCEL_TEST_CONNECTION-ghbHxQ
   ```

6. Enter `bin` to switch to binary mode.
7. Enter the following command:

   ```
   get <trace file name>
   ```

   where `<trace file name>` stands for the name of the trace file.

   Note: To receive all stored-procedure trace-files, you can enter the following command:

   ```
   mget aqt-trace-*
   ```

8. Enter `quit` to exit the FTP program.

The downloaded trace file is in UTF-8 encoding. You can view it in a text editor.

End of NOT Programming Interface information

Appendix A. Environment variables

The job control language (JCL) for the configuration of the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures contains a data definition `(DD) "AQTENV"`. This data definition includes a data set in which environment variables are defined. These variables control the behavior of several stored procedures.

Important:

- When editing the AQTENV data set with an ISPF editor, make sure not to use the NUM ON option. Otherwise the line numbers of the columns from 72 to 80 become part of the variable values. Also, do not insert blanks before or after the equals sign and make sure that you do not have trailing blanks at the end of the line.

  The AQTENV data set is made available to the stored procedures by the RUN OPTION `'ENVAR("_CEE_ENVFILE_S=DD:AQTENV")'`, which is set in the CREATE PROCEDURE statement for each procedure. If a line in the AQTENV data set matches the pattern *`NAME=VALUE`*, the environment variable *`NAME`* is set to *`VALUE`*. For more information, follow the **Related information** link at the end.
- Make sure that the permissions for the AQTENV data set include read access for all users who execute stored procedures. Otherwise, the environment variable settings do not take effect and corresponding error messages are written to the system log.
- After modifying settings in the AQTENV data set, refresh the WLM environment so that the changes can take effect.
- When upgrading to a new release, always use the AQTENV sample that is provided with the new version as a template for your own AQTENV data set. This ensures that you do not set obsolete environment variables and specify all mandatory new variables that were introduced with the new release.

You can set or override environment variables temporarily on a per-call basis. Settings will be valid only for the duration of a stored-procedure call. For more information, follow the link to the description of the *message* parameter under **Related reference** at the end of this topic.

AQT_ARCHIVE_COPY1

Specifies the name of the first image-copy data-set to be created when the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure archives partition data from Db2 for z/OS on an accelerator. For example:

```
AQT_ARCHIVE_COPY1 = &USERID..&DB..&TS..P&PART..&UNIQ.
```

where

&USERID.

ID of the user who runs SYSPROC.ACCEL_ARCHIVE_TABLES

&DB.

Name of the database that a partition resides in

&TS.

Name of the table space that the partition resides in

&PART.

Identifier of the (physical) partition. The letter P in the example is a text or string constant used as a prefix. This is required because &PART. resolves to a numeric value like 00001, and this not valid for qualifiers in a data-set name.

&UNIQ.

Causes the creation of a unique identifier

The template specification in the example could, for instance, result in the following image copy name:

```
BCKE.V4L1.BCKERTSE.CKRANGE3.P00001.D72R4KHN
```

- All template variables that are documented for the Db2 COPY utility can be used, with the exception of &SEQ (&SQ), &LIST (&LI), and &DSNUM.
- The chosen variables must ensure the uniqueness of image-copy data-set names. It is therefore recommended that you use at least the &PART. and &UNIQ. template variables.
- Templates must resolve to valid z/OS data set names.
- The template data-set names that you use must have been mapped to suitable data classes in the DFSMS.

Important: The AQT_ARCHIVECOPY_HLQ environment variable that was used with earlier versions of the High Performance Storage Saver is deprecated. Remove it from the AQTENV data set.

AQT_ARCHIVE_COPY2

Specifies the name of the second image-copy data-set to be created when the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure archives partition data from Db2 for z/OS on an accelerator. For more details, see AQT_ARCHIVE_COPY1.

AQT_ARCHIVE_RECOVERYCOPY1

Specifies the name of the third image-copy data-set to be created when the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure archives partition data from Db2 for z/OS on an accelerator. For more details, see AQT_ARCHIVE_COPY1.

AQT_ARCHIVE_RECOVERYCOPY2

Specifies the name of the fourth image-copy data-set to be created when the SYSPROC.ACCEL_ARCHIVE_TABLES stored procedure archives partition data from Db2 for z/OS on an accelerator. For more details, see AQT_ARCHIVE_COPY1.

AQT_HOST_PACKAGE_DIRECTORY

Update packages can be downloaded to and transferred from a directory in the UNIX file system (z/FS) of your z/OS data server or from a workstation. If a z/FS directory is used, you must set AQT_HOST_PACKAGE_DIRECTORY to an absolute path to point to the directory containing the update packages because your administration client and the SYSPROC.ACCEL_UPDATE2 stored procedure look for update packages in this directory. Most update packages must be downloaded from IBM Shopz or from IBM Fix Central. In the following steps, they are transferred to the accelerator, from where they are deployed and activated. There might be packages that you cannot install by yourself, but require the help of IBM support. For these types of updates, you must open a support case first. See *Opening a support case* in the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide* for more information.

AQT_DO_SYSLOGGING

Causes messages about the start and the completion of load operations to be written to the SYSLOG of the z/OS LPAR in which the SYSPROC.ACCEL_LOAD_TABLES stored procedure runs. Messages contain the name of the accelerator, the task number of the load jobs, and information about the reduction or increase of parallel processing threads (up to the limit set by AQT_MAX_UNLOAD_IN_PARALLEL). The variable is set by default (`AQT_DO_SYSLOGGING=SET`). To suppress these messages, comment out this line.

AQT_KEEPIDLE_INTERVAL

Specifies an interval that determines for how long a TCP/IP connection can be idle before keep-alive-packages are sent to the connected accelerator. The stored procedures build TCP/IP connections to the accelerator. To avoid connection failures caused by implicit connection closures, routers, or VPN (virtual private network) tunnels, TCP sends keep-alive-packages or probes after a period of time during which the network was idle. This is necessary to keep the connections up during long running load operations.

This setting affects all stored procedures and overwrites the KEEPALIVEOPTIONS settings for the TCP/IP stack that is specified in the Worload Manager (WLM) environment of the IBM Db2 Analytics Accelerator stored procedures. By default, the interval is set to 60 seconds. If the setting should turn out to be the source of problems, change the value or disable the mechanism entirely by commenting out the corresponding line in the AQTENV data set.

Note that socket connections between the stored procedures and the accelerator are also configured to send keep-alive-packages, and that the NO_DELAY and NON_BLOCKING options are set for the sending of these packages.

AQT_MAX_RETRIES_DSNUTILU

Determines how many times the SYSPROC.ACCEL_LOAD_TABLES stored procedure tries to call DSNUTILU if the attempt failed in the first place with SQL code `-471` and reason code `00E79002` (timeout error). The default value is 2.

AQT_MAX_RETRIES_FOR_LOCKS

This variable affects the lock mode **ROW**, which you can specify for table load processes. It is the number of times that the stored procedure tries to set a lock on a table. The default value is 2.

AQT_MAX_UNLOAD_IN_PARALLEL

The maximum number of parallel DSNUTILU invocations used by the SYSPROC.ACCEL_LOAD_TABLES, SYSPROC.ACCEL_ARCHIVE_TABLES, and SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES stored procedures when loading data from a partitioned Db2 table. Increasing the value leads to a better performance, provided that enough processors are available to handle additional parallel processes. Note also that increasing the value of AQT_MAX_UNLOAD_IN_PARALLEL to more than 8 might not increase the throughput any further. The default value is 4.

Important: If you increase the value of AQT_MAX_UNLOAD_IN_PARALLEL, you must, in most cases, *decrease* the NUMTCB value of the WLM application environment for the IBM Db2 Analytics Accelerator for z/OS stored procedures.

NUMTCB determines the number of stored procedures that can run in parallel in the same WLM address space. It is not the maximum number of threads each stored procedure can create internally. During the initial setup, the NUMTCB value is set to 15.

Each WLM address space has 2^{31} or 2 GB of memory. This memory is shared by all the stored procedures running in that WLM address space. So a higher NUMTCB value means that less memory is available for each individual stored procedure. SYSPROC.ACCEL_LOAD_TABLES starts around AQT_MAX_UNLOAD_IN_PARALLEL multiplied by 3 threads when it is running. So if AQT_MAX_UNLOAD_IN_PARALLEL is set to 20 and NUMTCB is set to 15, each thread has only 2 MB of memory (2 GB : 15 : 20 : 3 ≆ 2 MB). Therefore, if you increase AQT_MAX_UNLOAD_IN_PARALLEL, you must lower NUMTCB.

AQT_QUOTEFIX1026

Converts Turkish characters that would otherwise cause errors during table load operations. This variable is set by default (`AQT_QUOTEFIX1026=SET`).

AQT_RTS_EXTERNALIZATION_AUTH_FAILURE

Determines the behavior if the Db2 command `-ACCESS DATABASE` (`ACCESS DB`) is called by a stored procedure, but cannot be invoked because the user who runs the stored procedure lacks the proper authorization. Possible values:

ERROR (default)

Processing stops and an error message is displayed.

`WARNING`

Processing continues, but a warning is displayed at the end of the process. There will be just one generic warning.

`IGNORE`

Processing continues without a warning or error message.

AQT_SECONDS_BEFORE_RETRY_DSNUTILU

Specifies the interval in seconds between DSNUTILU calls (retries). By default, this interval is set to 60 seconds.

AQT_SECONDS_BEFORE_RETRY_LOCKS

This variable affects the lock mode **ROW**, which you can specify for table load processes. It is the interval of seconds that passes between the attempts to set a lock on a table. The default is 10 seconds.

AQT_SKIP_LOCK_LOCKMODE_ROW

Affects the behavior of SYSPROC.ACCEL_LOAD_TABLES if this stored procedure is run in conjunction with lock mode `ROW`. When enabled, the environment suppresses additional locks on the Db2 for z/OS tables or partitions to be unloaded. These locks prevent data mismatches between the source table and the target tables under certain conditions, but have negative side effects that might outweigh this advantage (slower load performance, timeout situations, interference with other applications).

AQT_SKIP_UNLOAD_EMPTY_PARTS

Suppresses invocations of the Db2 UNLOAD utility (DSNUTILU) when SYSPROC.ACCEL_LOAD_TABLES was called to load an accelerator-shadow table or partition, but the Db2 source table or partition is empty. This improves the performance of the load process. The detection of empty tables relies on Db2 real-time statistics.

Important: If the `<externalTool>` element is used as part of the *`table_load_specification`* parameter for SYSPROC.ACCEL_LOAD_TABLES, the Db2 UNLOAD utility is always called, irrespective of the setting of this variable.

AQT_SKIP_VALIDATE_XML

Skips the validation of XML code when it is returned in result sets or output parameter strings. The omission of this step leads to a better performance, especially when large result sets are returned. To disable this setting (turn XML validation on), comment out the line in the AQTENV data set.

AQT_SORTDEVT

Specifies the device type to use when sorting temporary data sets dynamically. This setting is used by the sort program when anIBM Db2 Analytics Accelerator stored procedure sends a request to a Db2 Utility that includes a sort job like this.

AQT_UTILITY_TMP

Specifies a template for the generation of unique temporary data-set names to be used by Db2 utilities. For example, several IBM Db2 Analytics Accelerator stored procedures invoke Db2 utilities, which in turn create temporary data sets with the user ID of the stored-procedure caller as the high-level qualifier. If you do not want this to happen, you can set AQT_UTILITY_TMP to generate a different name. The syntax is:

```
AQT_UTILITY_TMP = &USERID..AQT.&UNIQ.
```

where

&USERID.

ID of the user who runs the stored procedure

&UNIQ.

Causes the creation of a unique identifier

The specification needs to resolve to valid and unique data-set names. To ensure uniqueness, always make the &UNIQ. variable part of the template value. The string AQT is a constant. For a description of the DB2 template variables, see the entry for AQT_ARCHIVE_COPY1 further up in this list.

TZ

Sets the correct timezone for stored procedures running in the WLM address space that has been configured for IBM Db2 Analytics Accelerator. The setting affects the formatting of timestamps in trace files.

Related reference

The message parameter

The `message` parameter is used by all IBM Db2 Analytics Accelerator stored procedures. It serves as an input parameter and as an output parameter.

Related information

TCP_KeepAlive socket option

z/OS V2R1.0 XL C/C++ Programming Guide (SC14-7315-13)

Specifying how long utilities wait for resources

Appendix B. Required access rights

The following table lists the access rights that are required for calling IBM Db2 Analytics Accelerator stored procedures successfully (the user who installs or binds the stored procedures requires different access rights; for more information, see the *IBM Db2 Analytics Accelerator for z/OS: Installation Guide*).

Important: Each stored procedures requires write access to the `/tmp` directory in the z/OS UNIX file system (zFS) on z/OS if tracing is turned on for the procedure.

Table 14. Access rights by stored procedure

Stored procedure	Required access rights
• SYSPROC.ACCEL_ADD_ACCELERATOR2	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_ADD_TABLES	• EXECUTE on the stored procedure • MONITOR1 (needed for calling ADMIN_INFO_SYSPARM from the procedure) • Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
SYSPROC.ACCEL_ALTER_TABLES	• EXECUTE on the stored procedure • MONITOR1 (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)). • Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**
SYSPROC.ACCEL_ARCHIVE_TABLES	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • Authorization to read the original DB2 base tables of the accelerator-shadow tables that are listed in the *table_archive_specification* parameter • Authorization to run the UNLOAD utility on all affected DB2 tables. • Authorization to run the **-STOP DATABASE** command on the Db2 for z/OS table spaces that contain the table partitions to be archived • Authorization to run the **-DISPLAY DATABASE** command on all Db2 for z/OS databases that contain table partitions to be archived • Authorization to run the **-START DATABASE** command on the Db2 for z/OS table spaces that contain the table partitions to be archived • MONITOR1 authorization • Authorization to run the COPY utility on all affected Db2 for z/OS tables • Authorization to run the LOAD utility on all affected Db2 for z/OS tables • Authorization to run the RUNSTATS utility on all affected tables • Authorization to run the REPAIR utility on all affected Db2 for z/OS tables • Read/write and execute access to the /tmp directory for the user who calls the stored procedure (UNIX System Services pipes are created in this directory) • Privilege to connect to Db2 for z/OS via RRSAF. **Important:** If you want to grant the required DB2 privileges to secondary authorization IDs, for example groups, make sure that the DB2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
SYSPROC.ACCEL_ARCHIVE_TABLES (continued)	• Privilege to create data sets with the qualifier *<USERID>*.AQT, where *<USERID>* is the user ID of the person invoking the stored procedure • ACCESS(ALTER) in RACF for COPY data sets • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_COLLECT_TABLE_STATISTICS	• EXECUTE on the stored procedure • MONITOR1 (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)). • Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_CONFIG_OPTIMIZATION_PROFILE	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**
• SYSPROC.ACCEL_CONTROL_ACCELERATOR	• EXECUTE on the stored procedure • MONITOR1 authorization (needed so that DSNWLIR can be used to run Db2 commands via the Db2 Instrumentation Facility Interface (IFI)). • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • If an output file or data-set location is specified: Write access in RACF to the output data set for trace data or write access to the specified location in the z/OS UNIX file system. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called. **By element:** In addition to the general authorizations, the use of COMMAND parameter input elements requires the right to execute a particular user-defined function. This allows you to grant execution rights individually for each element. **<cancelTasks>** EXECUTE on DSNAQT.ACCEL_CONTROL_CANCELTASKS **<clearTraceData>** EXECUTE on DSNAQT.ACCEL_CONTROL_CLEARTRACEDATA **<getAcceleratorInfo>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETACCELERATORINFO **<getAcceleratorTasks>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETACCELERATORTASKS **<getActivationLog>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETACTIVATIONLOG **<getReplicationEvents>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETREPLICATIONEVENTS

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
- SYSPROC.ACCEL_CONTROL_ACCELERATOR (continued)	**<getTraceConfig>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETTRACECONFIG **<getTraceData>** EXECUTE on DSNAQT.ACCEL_CONTROL_GETTRACEDATA **<setTraceConfig>** EXECUTE on DSNAQT.ACCEL_CONTROL_SETTRACECONFIG **<startReplication>** EXECUTE on DSNAQT.ACCEL_CONTROL_STARTREPLICATION **<stopReplication>** EXECUTE on DSNAQT.ACCEL_CONTROL_STOPREPLICATION **<waitForReplication>** EXECUTE on DSNAQT.ACCEL_CONTROL_ WAITFORREPLICATION
SYSPROC.ACCEL_CREATE_REFERENCE_TABLES	- EXECUTE on the stored procedure - EXECUTE on the SYSACCEL.* packages - DBCTRL authority on the database explicitly specified by the stored procedure option `userSpecifiedDatabase`, or, if `userSpecifiedDatabase` is not specified, DBCTRL authority on database DSNDB04. You can also grant a higher authority that is not database-specific, such as SYSADM, system DBADM, and so on. - MONITOR1 authorization - Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure - RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. - SELECT, INSERT, and UPDATE on SYSACCEL.SYSACCELERATEDTABLES - EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**
• SYSPROC.ACCEL_GET_QUERIES • SYSPROC.ACCEL_GET_QUERIES2	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_GET_QUERY_DETAILS • SYSPROC.ACCEL_GET_QUERY_DETAILS2	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
SYSPROC.ACCEL_GET_QUERY_EXPLAIN	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**
• SYSPROC.ACCEL_GET_TABLES_DETAILS	• EXECUTE on the stored procedure • Authorization to run the DISPLAY DATABASE command for the databases that hold the tables listed in the *table_specification* parameter • Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics) • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_GET_TABLES_INFO	• EXECUTE on the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights

SYSPROC.ACCEL_GRANT_TABLES_REFERENCE	• EXECUTE on the stored procedure
	• EXECUTE on the SYSACCEL.* packages
	• INSERT on SYSACCEL.SYSACCELERATEDTABLESAUTH
	• MONITOR1 authorization
	• Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure
	• RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment.
	• EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
	• One of the following privileges:
	– SELECT privilege with GRANT option on every table in the *`table_set`* that is used with the stored procedure.
	– Access to the predefined RACF profile DB2ACCEL.FEDERATION. To this end, it is required that:
	1. The stored procedure runs in a WLM environment with an APF-authorized address space. That is, the load module for the stored procedure must reside in an APF-authorized library.
	2. A RACF profile DB2ACCEL.FEDERATION exists in the RACF resource class for Db2 (DSNR).
	Example:
	`-- Setup profile DB2ACCEL.FEDERATION`
	`with group FEDADMG`
	`SETR CLASSACT(DSNR)`
	`RDEFINE DSNR (DB2ACCEL.FEDERATION)`
	`OWNER(DB2OWNER) UACC(NONE)`
	`PERMIT DB2ACCEL.FEDERATION CLASS(DSNR)`
	`ID(FEDADMG) ACCESS(READ)`
	
	`SETR RACLIST(DSNR) REFRESH`
	The `ID` can be a group (here: `FEDADMG`), a single user, or a list of users. A sample job including the profile definition is provided in the SAQTSAMP.AQTSFEDR data set member.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
• SYSPROC.ACCEL_LOAD_TABLES	• EXECUTE on the stored procedure • MONITOR1 authorization • Authorization to read the original Db2 base tables of the accelerator-shadow tables that are listed in the *table_load_specification* parameter • DISPLAY authorization (for calling -DIS GROUP) • Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics) • Authorization to execute the RUNSTATS utility on the databases that contain the tables to be loaded, such as the STATS authorization. • Authorization to run ADMIN_INFO_SYSPARM so that ZPARMs can be retrieved. • Read/write and execute access to the /tmp directory for the user who calls the stored procedure (UNIX System Services pipes are created in this directory) • Privilege to connect to Db2 for z/OS via RRSAF. **Important:** If you want to grant the required Db2 privileges to secondary authorization IDs, for example groups, make sure that the Db2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
• SYSPROC.ACCEL_REMOVE_ACCELERATOR	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
SYSPROC.ACCEL_REMOVE_REFERENCE_TABLES	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • Ownership of the accelerator-only tables in the table set. Alternatively, you can use DBADM, SYSADM, SYSCTRL, or system DBADM authority. • If tables reside in implicitly defined databases, authority to remove tables from these databases. • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • SELECT and DELETE on SYSACCEL.SYSACCELERATEDTABLES • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
• SYSPROC.ACCEL_REMOVE_TABLES	• EXECUTE on the stored procedure • DELETE on SYSACCEL.SYSACCELERATEDTABLESAUTH • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • Authorization to run the ACCESS DB command on the databases that the tables reside in (needed to refresh Db2 real-time statistics). • Authorization to run the -DISPLAY DATABASE command for all databases that contain table partitions to be restored • Authorization to run the **`-START DATABASE`** command on all table spaces that the tables belong to (for the utility lock that is used during the restore operation) • Authorization to run the **`-STOP DATABASE`** command on all table spaces that the tables belong to • Authorization to run the REPAIR utility on all tables (to reset the `DB2 RO_ACCEL` state) • Authorization to run the RECOVER utility on all tables • Authorization to run the CHECK DATA utility on all tables • Authorization to run the REBUILD INDEX utility on all indexes of the restored tables • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure (temporary files are created there) • Privilege to connect to Db2 for z/OS via RRSAF. **Important:** If you want to grant the required DB2 privileges to secondary authorization IDs, for example groups, make sure that the DB2 SIGNON exit is correctly configured for the use of such IDs. The default SIGNON exit in Db2 for z/OS does not accept secondary authorization IDs. You must therefore use a sample SIGNON exit for this purpose, that is, DSN3SSGN or an equivalent. For more information, see Using secondary IDs for sign-on requests.

Table 14. Access rights by stored procedure (continued)	
Stored procedure	**Required access rights**
SYSPROC.ACCEL_RESTORE_ARCHIVE_TABLES (continued)	• Privilege to create data sets with the qualifier `<USERID>.AQT`, where `<USERID>` is the user ID of the person invoking the stored procedure • Read access in RACF to COPY data sets that serve as the basis for the restore operation • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • DELETE on SYSACCEL.SYSACCELERATEDTABLESAUTH • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**

SYSPROC.ACCEL_REVOKE_TABLES_REFERENCE (continued)	• In addition, the caller of the stored procedure needs *one* of the following privileges:
	– SECADM authorization
	– ACCESSCTRL authorization
	Either of these authorizations is unnecessary if *one* of the following conditions is met:
	– Matching user IDs: For each table in your *table_set*, look up the ID in the GRANTOR column of SYSACCEL.SYSACCELERATEDTABLESAUTH. Compare these IDs with the ID of the user who runs the stored procedure. The GRANTOR IDs of all tables in the *table_set* must match.
	– Access to the predefined RACF profile DB2ACCEL.FEDERATION. To this end, it is required that:
	1. The stored procedure runs in a WLM environment with an APF-authorized address space. That is, the load module for the stored procedure must reside in an APF-authorized library.
	2. A RACF profile DB2ACCEL.FEDERATION exists in the RACF resource class for Db2 (DSNR).
	Example:
	`-- Setup profile DB2ACCEL.FEDERATION`
	`with group FEDADMG`
	`SETR CLASSACT(DSNR)`
	`RDEFINE DSNR (DB2ACCEL.FEDERATION)`
	`OWNER(DB2OWNER) UACC(NONE)`
	`PERMIT DB2ACCEL.FEDERATION CLASS(DSNR)`
	`ID(FEDADMG) ACCESS(READ)`
	
	`SETR RACLIST(DSNR) REFRESH`
	The `ID` can be a group (here: `FEDADMG`), a single user, or a list of users. A sample job including the profile definition is provided in the SAQTSAMP.AQTSFEDR data set member.

Table 14. Access rights by stored procedure (continued)

Stored procedure	Required access rights
• SYSPROC.ACCEL_SET_TABLES_ACCELERATION	• EXECUTE on the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
SYSPROC.ACCEL_SET_TABLES_REPLICATION	• EXECUTE on the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
• SYSPROC.ACCEL_TEST_CONNECTION	• EXECUTE on the stored procedure • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • RACF ACCESS(READ) on the data set that contains the AQTDEF6 file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Table 14. Access rights by stored procedure (continued)

Stored procedure	**Required access rights**
• SYSPROC.ACCEL_UPDATE_CREDENTIALS	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.
SYSPROC.ACCEL_UPDATE_SOFTWARE2	• EXECUTE on the stored procedure • EXECUTE on the SYSACCEL.* packages • MONITOR1 authorization • Read/write and execute access to the `/tmp` directory for the user who calls the stored procedure • For the functions **`listZPackageDirectory`** and **`deployPackage`**: Read access to the directory that the AQT_HOST_PACKAGE_DIRECTORY environment variable points to. • RACF ACCESS(READ) on the data set that contains the AQTENV file in the started task procedure of the Workload Manager (WLM) environment. • Sub-commands of the `update_command` option might require further authorizations. • EXECUTE on the DSNADM.DSNADMIZ package to allow access to system parameters when the SYSPROC.ADMIN_INFO_SYSPARM stored procedure is called.

Appendix C. Members of SAQTSAMP

The SAQTSAMP data set contains various samples for installing, configuring or running the product's stored procedures. The following table lists all these members and provides brief descriptions of their functions.

NOT Programming Interface information

Table 15. Members of the SAQTSAMP data set

Member name	Function
AQTDEF6	Default trace specification and other settings for the **message** input parameter. The default input is used when stored procedures are called with a NULL value or empty string in the **message** input parameter. The sample uses a verbosity level of INFO for all stored procedure calls and specifies that the traces are to be kept in a specific directory if the procedure ends with an error. To use this member, you must add an appropriate DD statement (DD AQTDEF6) to the JCL that starts the Workload Manager (WLM) environment for IBM Db2 Analytics Accelerator stored procedures.
AQTENV	Default environment variable settings for the stored procedures.
AQTLCABA	Sample C program that implements the rerouting and load-callback functions for the high-availability load sample.
AQTLMAIN	Sample C program that implements the main functionality for the high-availability load sample.
AQTLSHMC	Sample C program that implements inter-process communication (IPC) functionality for the high-availability load sample.
AQTLSHMH	Header files for the inter-process communication (IPC) functionality for the high-availability load sample.
AQTSCALL	Sample application program in the C programming language with embedded SQL CALL statements for calling IBM Db2 Analytics Accelerator stored procedures.
AQTSCI01	Db2 command line processor script that calls the following stored procedures in the order indicated: 1. SYSPROC.ACCEL_REMOVE_TABLES (returns only the version of the stored procedure) 2. SYSPROC.ADMIN_INFO_SYSPARM 3. SYSPROC.ADMIN_COMMAND_Db2(-DIS ACCEL) 4. SYSPROC.ADMIN_COMMAND_Db2(-DIS GROUP) 5. SYSPROC.DSNUTILU(UNLOAD)

Table 15. Members of the SAQTSAMP data set (continued)

Member name	Function
AQTSCI02	Db2 command line processor script that calls the following stored procedures and Db2 commands in the order indicated: 1. SYSPROC.ACCEL_ADD_ACCELERATOR for the initial setup Running this stored procedure requires a valid pairing code. You must therefore edit AQTSCI02 so that a valid pairing code is provided when SYSPROC.ACCEL_ADD_ACCELERATOR is invoked. **Important:** In the sample script, the call of this stored procedure has been commented out. To activate the call, you must uncomment the corresponding line. 2. SYSPROC.ACCEL_TEST_CONNECTION 3. SYSPROC.ACCEL_ADD_TABLES 4. SYSPROC.ACCEL_GET_TABLES_INFO 5. SYSPROC.ACCEL_LOAD_TABLES To enforce a reload of the accelerator-shadow tables and enable these for query acceleration: 1. SYSPROC.ACCEL_LOAD_TABLES 2. SYSPROC.ACCEL_SET_TABLES_ACCELERATION(ON) To execute the query: 1. START -ACCEL 2. Query execution 3. SYSPROC.ACCEL_GET_QUERIES 4. STOP -ACCEL To check various other functions: 1. SYSPROC.ACCEL_REMOVE_TALBES 2. SYSPROC.ACCEL_CONTROL_ACCELERATOR (including the various subfunctions) 3. SYSPROC.ACCEL_REMOVE_ACCELERATOR **Important:** In the sample script, the call of this stored procedure has been commented out. To activate the call, you must uncomment the corresponding line. **Note:** This script requires customization.
AQTSFEDR	Sample JCL to create a RACF profile for federated access. The profile provides users with appropriate RACF permissions to run the following stored procedures for federated access: • ACCEL_GRANT_TABLES_REFERENCE • ACCEL_REVOKE_TABLES_REFERENCE
AQTSJI00	JCL that collects information about required IBM Db2 Analytics Accelerator databases and tables.
AQTSJI01	JCL that calls the Db2 command line processor for verifying IBM Db2 Analytics Accelerator stored procedures without an accelerator.

Table 15. Members of the SAQTSAMP data set (continued)

Member name	Function
AQTSJI02	JCL that calls all verification steps one-by-one. For a successful completion, a running and connected accelerator is required.
AQTSJI03	JCL that compiles, links, and invokes the AQTSCALL sample program, which calls IBM Db2 Analytics Accelerator stored procedures. For a successful completion of the job, a running accelerator is required.
AQTSJI04	JCL for a manual recovery of a partition that was archived with the High Performance Storage Saver. With the help of this JCL, you can recover an archived partition if the accelerator that holds the data is unavailable. No changes will be made on the accelerator. The JCL uses the image copies in Db2 for z/OS for the recovery. **Important:** When the accelerator returns to the *online* state at a later point-in-time, the data might not be in sync anymore with the data in Db2 for z/OS. This might result in different query results. To solve this problem, restore all archived partitions to Db2 for z/OS, and delete the corresponding accelerator-archive table when the accelerator is back online.
AQTSJI05	JCL that compiles, links, and starts the AQTSCALL sample program with code samples for federated access. These code samples call stored procedures to complete the following tasks: • Grant a referencing Db2 subsystem access to an accelerator-shadow table originating from the owning (remote) Db2 subsystem, or to an accelerator-only table. • Create a referencing accelerator-only table • Remove the referencing accelerator-only table • Revoke access to the accelerator-shadow table or accelerator-only table. A successful completion of the job requires a running accelerator. For more information, see *Federated access* in the *IBM Db2 Analytics Accelerator for z/OS: User's Guide*. **Note:** In IBM Db2 Analytics Accelerator for z/OS Version 5.1, the AQTSJI05 member serves a different function. It contains a JCL for the management of installation packages in the z/OS UNIX file system (zFS). This version of AQTSJI05 is not supported by IBM Db2 Analytics Accelerator Version 7.x. It is therefore important that you use an SAQTSAMP sample data set that was shipped with product version 7.x. Do not customize and use an outdated version of that data set.
AQTSJL01	JCL to compile, link, and bind the high-availability load sample programs.
AQTSJL02	JCL to compile, link, and bind the high-availability load sample programs.
AQTSJL03	JCL that starts the high-availability load sample program.

Table 15. Members of the SAQTSAMP data set (continued)

Member name	Function
AQTSSCHK	UNIX System Services shell script that checks the output of Db2 command line processor scripts containing IBM Db2 Analytics Accelerator stored procedure calls. If BPXBATCH is used to call this script, as in the AQTSJI02 member, the return code of this job step is determined by the **MESSAGE** output parameters of the stored procedures. Divide this return code by 256 to determine the severity of an error: **0** All stored procedures that were called returned information messages **4** Warning (return code 1024) **8** Error (return code 2048) **12** Severe (return code 3072)
AQTSSCPY	UNIX System Services shell script that copies SAQTSAMP members to the `/tmp/ivp` directory in the z/OS UNIX file system. **Note:** Db2 command line processor scripts cannot be run from a data set.
AQTSSLDG	Sample JCL that creates certificates for IBM Db2 for z/OS Data Gate. It generates a pair of keys and a certificate for Db2 for z/OS and stores it in a key ring for inbound access. The sample JCL also generates a pair of keys and a certificate for IBM Db2 for z/OS Data Gate on IBM Cloud Pak for Data for outbound access, and exports the certificate as a PKCS#12 file. In addition, appropriate RACF permissions are given to the user who runs IBM Db2 for z/OS Data Gate.
AQTSSLSA	Sample JCL that creates a CA certificate for IBM Integrated Synchronization, stores the certificate in a key ring and exports it, so that it can be imported into a Java key store. In addition, appropriate RACF permissions are given to the user who runs IBM Integrated Synchronization.
AQTSXADD	XML definition of a simple table specification (`<tableSpecifications>` element). This XML code can be used as input for the SYSPROC.ACCEL_ADD_TABLES stored procedure.
AQTSXALT	XML definition of a table specification as input for the SYSPROC.ACCEL_ALTER_TABLES stored procedure.
AQTSXCN0	XML code for the deletion of collected trace data. The XML code is used as the value of the **COMMAND** parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.
AQTSXCN1	XML code for the retrieval of accelerator status information. The XML code is used as the value of the **COMMAND** parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.
AQTSXCN2	XML code for the configuration of accelerator tracing. The XML code is used as the value of the **COMMAND** parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.
AQTSXCN3	XML code for the collection of trace data. The XML code is used as the value of the **COMMAND** parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.

Table 15. Members of the SAQTSAMP data set (continued)

Member name	**Function**
AQTSXCN4	XML code that lists the active tasks on the accelerator. The XML code is used as the value of the **COMMAND** parameter of the SYSPROC.ACCEL_CONTROL_ACCELERATOR stored procedure.
AQTSXIM0	XML code that returns just the version of an IBM Db2 Analytics Accelerator stored procedure without executing its actual function. The XML code is used as the value of the **MESSAGE** input parameter.
AQTSXQHI	XML definition that serves as an input value for the QUERY_SELECTION parameter of the SYSPROC.ACCEL_GET_QUERIES stored procedure.
AQTSXSD1	All XML schema definitions (xsd files) for the input and output parameters of the IBM Db2 Analytics Accelerator stored procedures.
AQTSXTCO	XML definition that serves as an input value for the DIAGNOSTIC_INPUT parameter of the SYSPROC.ACCEL_TEST_CONNECTION stored procedure.
AQTSXTS0	XML definition of a table set as input for various stored procedures.
AQTSXTSL	XML definition that serves as an input value for the TABLE_LOAD_SPECIFICATION parameter of the SYSPROC.ACCEL_LOAD_TABLES stored procedure.
AQTSXTSU	XML definition for an update of selected table partitions. The definition serves as an input value for the TABLE_LOAD_SPECIFICATION parameter of the SYSPROC.ACCEL_LOAD_TABLES stored procedure.
AQTTIJSP	JCL for the installation of the IBM Db2 Analytics Accelerator stored procedures.

End of NOT Programming Interface information

Appendix D. Sample C/C++ header for external loader callback interface

Use the following C/C++ header to define an interface for callback functions to be run by an external load application called by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. Make sure to compile your interface as a 31-bit XPLINK application.

C/C++ header

```
//------------------------------------------------------------------------------
//
// (C) COPYRIGHT International Business Machines Corp. 2018, 2019.
// All Rights Reserved
// Licensed Materials - Property of IBM
// 5697-DAA
// US Government Users Restricted Rights - Use, duplication or
// disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
//
// Function = Include file defining a C callback interface for the
//            SYSPROC.ACCEL_LOAD_TABLES stored procedure.
//
//------------------------------------------------------------------------------

#if !defined(__AQT_Load_Callback_hpp__)
#define __AQT_Load_Callback_hpp__

extern "C" {
typedef struct CallbackMessage {
    unsigned int length;  //maxmimum length if passed to callback,
                          //actual length must be set on return by the callback
    char*        text;    //buffer provided by stored procedure
};


typedef unsigned int (*version_callback_t) ();

/** Callback that returns the interface version for other callbacks
* run from ACCEL_LOAD_TABLES.
* ATTENTIION: THE INTERFACE OF THIS VERSION CALLBACK MUST NEVER CHANGE
* Other callback function might be added, existing callback function may be modified.
* All such changes would require an increase of the callback interface version
* returned by this version callback.
* Furthermore, the associated new or modified function types must be supported
* by the ACCEL_LOAD_TABLES procedure.
* If this is not the case ACCEL_LOAD_TABLES will errror out.
* @return 1, if the expected callback interface is
*            according to the specification in this header
*            (version_callback_t, init_callback_t, load_callback_t, term_callback_t)
*/

typedef unsigned int (*init_callback_t) (char const * const callerid,
                                         unsigned int degreeOfParallelism,
                                         unsigned int lockMode,
                                         unsigned int* allowChangeOfParallelism,
                                         CallbackMessage* message);

/** Callback that provides information about load seetings from ACCEL_LOAD_TABLES.
*
* @param[in] callerid: identifies the load request to the caller of
*                      ACCEL_LOAD_TABLES. The procedure is obliged to
*                      use the callerid value specified in its
*                      <externalTool> XML provided in the
*                      TABLE_LOAD_SPECIFICATION parameter.
* @param[in] callbackid: associates the request with a load request
*                        (see load_callback_t).
* @param[in] degreeOfParallelism: specifies the degree of parallelism
*                                 used for loading partitions.
*                                 In version 1 of this interface
*                                 this number remains unchanged during a load.
* @param[in] lockmode specifies the lock mode requested by the user
*                     when calling ACCEL_LOAD_TABLES:
*                     1=TABLESET, all tables in input are locked globally
*                     2=TABLE, only current table being unloaded is locked
*                     3=PARTITIONS, only the set of partitions
*                                   currently being unloaded  is locked
*                     4=CSROW, only committed data is read.
*                              If a row is locked by another app it is not skipped
*                     5=NONE, no locking,
*                             only committed rows are read, locked rows are skipped
* @param[out] allowChangeOfParallelism specifies, if degree of parallelism may be
*                     changed while the ACCEL_LOAD_TABLES procedures proceeds.
*                     0: changes are not allowed
*                     1 or any other value: changes are allowed
* @param[inout] message: message (UTF-8) is returned in a SP message
*                        AQT10513I - severity informational for returncode 0,
*                                    if message is not empty.
*                        AQT10514W - severity warning returncode 4.
*                        AQT10515E - severity error for other returncodes.
*                        As input parameter, the maximum message length in bytes is
*                             passed together with a pointer to a messagetext buffer.
*                        As output parameter, the length is modified to the actual
*                             message length and the text buffer is modified.
* @return 0, if the callback succeeded.
*         8, if the terminate operation did not succeed (for any reason)
*            message might be also transferred to SP in an output parameter
*/

typedef int (*load_callback_t) (char const * const callerid,
                      char const * const callbackid,
                     char const * const tableSchema,
                                 char const * const tableName,
                                 unsigned int partNum,
                                 char const * const pipeName,
                                 unsigned long long *numRows,
                CallbackMessage* message);

/** Callback interface for ACCEL_LOAD_TABLES.
 * This callback interface allows to provide data from non-DB2
 * sources into an accelerated table. If the caller of
 * SYSPROC.ACCEL_LOAD_TABLES uses an extension of the input XML
 * to specify tables to load, the callback will be used rather
 * than calling DSNUTILU(UNLOAD).
 * Example:
 *    <externalTool name =
 *       "third-party tool name"
 *       enforceTableLoadSequence = "true"
 *       <callback
 *          callerid = "0x4e353456"
 *          libary = "TPCALLB"
 *          versionFunction="VersionCallback"
 *          initFunction="InitCallback"
 *          loadFunction="LoadCallBack"
 *          terminateFunction="TermCallBack" />
 *    </externalTool>
 *
 * Prerequisites:
 * 1. The implementation of the LoadCallBack function has to be 31bit C/C++
 *    using XPLINK.
 * 2. The imlementation must emulate the DSNUTILU(UNLOAD)
 *    behaviour related to opening, writing and closing the pipe
 *   (also in error cases)
 *
 * @param[in] callerid: identifies the load request to the caller of
 *                      ACCEL_LOAD_TABLES. The procedure is obliged to
 *                      use the callerid value specified in its
 *                      <externalTool> XML provided in the
 *                      TABLE_LOAD_SPECIFICATION parameter.
 * @param[in] callbackid: identifies the callback invocation for possible
 *                      later termination.
 * @param[in] tableSchema: creator/schema of the table to load according to
 *                         SYSIBM.SYSTABLES(CREATOR), UTF-8 null terminated.
 * @param[in] tableName: name of the table to load according to
 *                       SYSIBM.SYSTABLES(NAME), UTF-8 null terminated.
 * @param[in] partNum: physical partition number.
 * @param[in] pipeName: name of a USS pipe where the data is to be written to.
 *                      UTF-8, null terminated. Pipe is owned by
 *                      ACCEL_LOAD_TABLES.
 * @param[out] numRows: the number of rows written to the pipe
 * @param[inout] message: message (UTF-8) is returned in a SP message
 *                        AQT10513I - severity informational for returncode 0,
 *                                    if message is not empty.
 *                        AQT10514W - severity warning returncode 4.
 *                        AQT10515E - severity error for other returncodes.
 *               As input parameter,
 *                  the maximum message length in bytes is passed together
 *                  with a pointer to a messagetext buffer.
 *               As output parameter,
 *                  the length is modified to the actual message length
 *                  and the text buffer is modified.
 *
 * @return 0, if the callback succeeded.
 *         4, if the callback indicates a warning
 *            (whose accompanied message is returned by the SP),
 *            unload operation was successful.
 *         8, if the unload operation failed,
 *            i.e. the data were not sent completely to the pipe.
 */

typedef int (*term_callback_t) (char const * const callerid,
                                char const * const callbackid,
                CallbackMessage* message);

/** Termination of a callback run from ACCEL_LOAD_TABLES.
 * Terminates a load request and returns if the termination is done.
 * This means that an opened pipe in a running load request is closed.
 * In case there is no active load requestfor the specified callbackid,
 * return immediately with a non-zero return code.
 * @param[in] callerid: identifies the load request to the caller of
 *                      ACCEL_LOAD_TABLES. The procedure is obliged to
 *                      use the callerid value specified in its
 *                      <externalTool> XML provided in the
 *                      TABLE_LOAD_SPECIFICATION parameter.
 * @param[in] callbackid: associates the request with a load request
 *                        (see load_callback_t).
 * @param[inout] message: message (UTF-8) is returned in a SP message
 *                        AQT10513I - severity informational for returncode 0,
 *                                    if message is not empty.
 *                        AQT10514W - severity warning for other returncodes.
 *               As input parameter,
 *                  the maximum message length in bytes is passed together
 *                  with a pointer to a messagetext buffer.
 *               As output parameter,
 *                  the length is modified to the actual message length
 *                  and the text buffer is modified.
 * @return 0, if the termination succeeded.
 *         8, if the terminate operation did not succeed (for any reason)
 *
 */

}#endif // __AQT_Load_Callback_hpp__
```

Appendix E. Sample C/C++ header for rerouting callback interface

Use the following C/C++ header to define an interface for callback functions to be run by an external rerouting application called by the SYSPROC.ACCEL_LOAD_TABLES stored procedure. Make sure to compile your interface as a 31-bit XPLINK application.

C/C++ header

```
//------------------------------------------------------------------------------
//
// (C) COPYRIGHT International Business Machines Corp. 2018, 2019.
// All Rights Reserved
// Licensed Materials - Property of IBM
// 5697-DAA
// US Government Users Restricted Rights - Use, duplication or
// disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
//
// Function = Include file defining a C callback interface for the
//            SYSPROC.ACCEL_LOAD_TABLES stored procedure.
//
//------------------------------------------------------------------------------

#if !defined(__AQT_LoadRerouteCallbackInterface_hpp__)
#define __AQT_LoadRerouteCallbackInterface_hpp__
#include "LoadCallbackInterface.hpp"

/** Callback that returns the interface version for other callbacks
* run from ACCEL_LOAD_TABLES.
* ATTENTION: THE INTERFACE OF THIS VERSION CALLBACK MUST NEVER CHANGE
* Other callback function might be added, existing callback function may be modified.
* All such changes would require an increase of the callback interface version
* returned by this version callback.
* Furthermore, the associated new or modified function types must be supported
* by the ACCEL_LOAD_TABLES procedure.
* If this is not the case ACCEL_LOAD_TABLES will error out.
* @return 1, if the expected callback interface is
*            according to the specification in this header
*/
typedef unsigned int (*version_callback_rdc_t) ();

/** Callback that provides information about load settings from ACCEL_LOAD_TABLES.
*
* @param[in] callerid: identifies the load request to the caller of
*                      ACCEL_LOAD_TABLES. The procedure is obliged to
*                      use the callerid value specified in its
*                      <externalTool> XML provided in the
*                      TABLE_LOAD_SPECIFICATION parameter.
* @param[in] degreeOfParallelism: specifies the degree of parallelism
*                                 used for loading partitions.
*                                 In version 1 of this interface
*                                 this number remains unchanged during a load.
* @param[out] allowChangeOfParallelism specifies, if degree of parallelism may be
*                     changed while the ACCEL_LOAD_TABLES procedures proceeds.
*                     0: changes are not allowed
*                     1 or any other value: changes are allowed
* @param[inout] message: message (UTF-8) is returned in a SP message
*                        AQT10513I - severity informational for returncode 0,
*                                    if message is not empty.
*                        AQT10514W - severity warning returncode 4.
*                        AQT10515E - severity error for other returncodes.
*                        As input parameter, the maximum message length in bytes is
*                             passed together with a pointer to a messagetext buffer.
*                        As output parameter, the length is modified to the actual
*                             message length and the text buffer is modified.
* @return 0, if the init callback succeeded.
*         8, if the init callback operation did not succeed (for any reason)
*/
typedef unsigned int (*init_callback_rdc_t) (char const * const callerid,
                                             unsigned int degreeOfParallelism,
                                             unsigned int* allowChangeOfParallelism,
                                             CallbackMessage* message);
```

```
/** Pre-unload Callback interface for ACCEL_LOAD_TABLES.
 * This callback interface allows to define a pipe name
 * to which the UNLOAD utility writes the data to.
 *
 * Example:
 *    <externalTool name =
 *       "third-party tool name"
 *       enforceTableLoadSequence = "true"
 *       <rerouteDataCallback
 *          callerid = "0x4e353456"
 *          libary = "TPCALLB"
 *          versionFunctionRDC="VersionCallbackRDC"
 *          initFunctionRDC="InitCallbackRDC"
 *          preUnloadFunctionRDC="PreUnloadCallbackRDC"
 *          postUnloadFunctionRDC="PostUnloadCallbackRDC" />
 *    </externalTool>
 *
 * Prerequisites:
 * 1. The implementation of the function has to be 31bit C/C++
 *    using XPLINK.
 * 2. The UNLOAD utility will write the data to the pipe specified by the callback
 *    implementation
 * 3. The callback implementation must provide the data to the loadPipeName
 *   (data to be loaded to the accelerator)
 * 4. The UNLOAD utility will be called after the pre_unload_callback_rdc_t
 *    function returns
 *
 * @param[in] callerid: identifies the load request to the caller of
 *                      ACCEL_LOAD_TABLES. The procedure is obliged to
 *                      use the callerid value specified in its
 *                      <externalTool> XML provided in the
 *                      TABLE_LOAD_SPECIFICATION parameter.
 * @param[in] callbackid: uniquely identifies the thread that calls the UNLOAD
 * @param[in] tableSchema: creator/schema of the table to load according to
 *                         SYSIBM.SYSTABLES(CREATOR), UTF-8 null terminated.
 * @param[in] tableName: name of the table to load according to
 *                       SYSIBM.SYSTABLES(NAME), UTF-8 null terminated.
 * @param[in] partNum: physical partition number.
 * @param[in] loadPipeName: name of a USS pipe where data to be loaded must be written
 *                      to. UTF-8, null terminated. Pipe is owned by
 *                      ACCEL_LOAD_TABLES.
 * @param[in] unloadPipeName: name of a USS pipe to which the UNLOAD utility writes
 *                             the data to. Pipe is owned by ACCEL_LOAD_TABLES.
 * @param[inout] message: message (UTF-8) is returned in a SP message
 *                        AQT10513I - severity informational for returncode 0,
 *                                    if message is not empty.
 *                        AQT10514W - severity warning returncode 4.
 *                        AQT10515E - severity error for other returncodes.
 *               As input parameter,
 *                  the maximum message length in bytes is passed together
 *                  with a pointer to a messagetext buffer.
 *               As output parameter,
 *                  the length is modified to the actual message length
 *                  and the text buffer is modified.
 *
 * @return 0, if the callback succeeded.
 *         4, if the callback indicates a warning
 *            unload operation was successful.
 *         8, if the unload operation failed,
 *            i.e. the data were not sent completely to the pipe.
 */

typedef int (*pre_unload_callback_rdc_t) (char const * const callerid,
                                            char const * const callbackid,
                                           char const * const tableSchema,
                                           char const * const tableName,
                                           unsigned int partNum,
                                           char const * const loadPipeName,
                                           char const * const unloadPipeName,
                                          CallbackMessage* message);

/** Unload complete notification from ACCEL_LOAD_TABLES.
 * After the UNLOAD utility was executed the PostUnloadCallback function
 * will be called.
 * @param[in] callerid: identifies the load request to the caller of
 *                      ACCEL_LOAD_TABLES. The procedure is obliged to
 *                      use the callerid value specified in its
 *                      <externalTool> XML provided in the
 *                      TABLE_LOAD_SPECIFICATION parameter.
 * @param[in] callbackid: uniquely identifies the thread that has called the
 *                        UNLOAD utility (see pre_unload_callback_rdc_t ).
 * @param[in] successful: was the UNLOAD Utility executed successfully
 *                        0: No
 *                        1: Yes
 * @param[inout] message: message (UTF-8) is returned in a SP message
 *                        AQT10513I - severity informational for returncode 0,
 *                                    if message is not empty.
 *                        AQT10514W - severity warning for other returncodes.
 *               As input parameter,
 *                  the maximum message length in bytes is passed together
 *                  with a pointer to a messagetext buffer.
 *               As output parameter,
 *                  the length is modified to the actual message length
 *                  and the text buffer is modified.
 * @return 0, if the reroute succeeded.
 *        8, if the reroute operation did not succeed (for any reason)
 *
 */
typedef int (*post_unload_callback_rdc_t) (char const * const callerid,
                                           char const * const callbackid,
                                           int successful,
                                           CallbackMessage* message);


#endif // __AQT_LoadRerouteCallbackInterface_hpp__
```

Appendix F. Transcript of the SAQTSAMP(AQTSXSD1) data set member

Search this transcript for information about the structure of the XML input and output elements that are used by IBM Db2 Analytics Accelerator for z/OS stored procedures.

AQTSXSD1

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:dwa="http://www.ibm.com/xmlns/prod/dwa/2011"
    targetNamespace="http://www.ibm.com/xmlns/prod/dwa/2011">
  <!--@copyright(disclaimer)
Licensed Materials - Property of IBM
5697-DA7
(C) Copyright IBM Corp. 2011, 2023

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

DISCLAIMER OF WARRANTIES :
Permission is granted to copy and modify this Sample code provided that
both the copyright notice, and this permission notice and warranty
disclaimer, appear in all copies and modified versions.

THIS SAMPLE CODE IS LICENSED TO YOU AS-IS.
IBM AND ITS SUPPLIERS AND LICENSORS DISCLAIM ALL WARRANTIES, EITHER EXPRESS
OR IMPLIED, IN SUCH SAMPLE CODE, INCLUDING THE WARRANTY OF NON-INFRINGEMENT
AND THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR
PURPOSE. IN NO EVENT WILL IBM OR ITS LICENSORS OR SUPPLIERS BE LIABLE FOR
ANY DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE THE SAMPLE CODE
OR COMBINATION OF THE SAMPLE CODE WITH ANY OTHER CODE. IN NO EVENT SHALL
IBM OR ITS LICENSORS AND SUPPLIERS BE LIABLE FOR ANY LOST REVENUE, LOST
PROFITS OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL
OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF
LIABILITY, EVEN IF IBM OR ITS LICENSORS OR SUPPLIERS HAVE BEEN ADVISED OF
THE POSSIBILITY OF SUCH DAMAGES.

@endCopyright
      -->
  <xs:element name="acceleratorOptions">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
           Global settings applied to an accelerator instance that is created
           when pairing
           an accelerator with a DB2 subsystem.
          </xs:documentation>
        </xs:annotation>
        <xs:element maxOccurs="unbounded" minOccurs="0"
            name="acceleratorSetting" type="dwa:acceleratorSettingType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
                Lists the specific settings applied when creating the
                accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="optimizationProfileList">
    <xs:complexType>
```

```
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            List of the optimization profiles
          </xs:documentation>
        </xs:annotation>
        <xs:element name="optimizationProfile"
            type="dwa:optimizationProfileType" minOccurs="0"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="optimizationProfileType">
    <xs:sequence>
      <xs:element name="optimizationProfileSchema" type="xs:string"/>
      <xs:element name="optimizationProfileName" type="xs:string"/>
      <xs:element name="optimizationProfileXml" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="configOptimizationProfileCommand">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Optimization commnad that should be processed by the stored
            procedure with optional qualifiers
          </xs:documentation>
        </xs:annotation>
        <xs:element name="optimizationProfileCommand"
            type="dwa:configOptimizationProfileCommandType" minOccurs="1"
            maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="controlCommand">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="setTraceConfig" type="dwa:traceConfigUpdateType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Add new trace profiles, remove trace profiles or set the active
              trace profile.
              Does not
              generate output.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getTraceConfig">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Return the active trace configuration in the output parameter
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="clearTraceData">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Clear historic trace data on the accelerator. Does not generate
              output.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getTraceData" type="dwa:getTraceDataType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Collect trace data from the accelerator and return it in a file
              or a stored
              procedure
              result set.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getAcceleratorInfo" type="dwa:getInfoTasksType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
                            Return accelerator status and active software
                            levels in the output parameter.
                            Optionally return information about encryption.
                       </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getAcceleratorTasks">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Return currently executing tasks on the accelerator in the output
              parameter.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="cancelTasks" type="dwa:cancelTasksType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Requests cancellation of one or more currently executing tasks on
              the accelerator.
              Does not
              generate output.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="startReplication">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Start data replication for all tables that are enabled for
              replication and loaded.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="stopReplication" type="dwa:stopReplicationType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Stop data replication for all tables on the accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getReplicationEvents"
            type="dwa:getReplicationEventsType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Requests status messages for table replication.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="waitForReplication"
            type="dwa:waitForReplicationType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Wait until all replicated tables have been synchronized to the
              current log position.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="enableProcedurePackage"
            type="dwa:procedurePackageNameType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Enable a stored procedure package that was transferred and
              applied to the accelerator.
              Added with version 1.4.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="disableProcedurePackage"
            type="dwa:procedurePackageNameType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Disable a stored procedure package for an accelerator.
              Added with version 1.4.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getActivationLog" type="dwa:activationLogType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
                            Return the latest installation log as UTF-8 in
                            result set with cursor name
                            ACCELERATOR_TRACE_CURSOR.
                            Added with version 1.5, enhancedd with version 1.8
                        </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getAdditionalSupport"
            type="dwa:getAdditionalSupportType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
                            Activate backend document collection
                        </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
            <xs:enumeration value="1.2"/>
            <xs:enumeration value="1.3"/>
            <xs:enumeration value="1.4"/>
            <xs:enumeration value="1.5"/>
            <xs:enumeration value="1.6"/>
            <xs:enumeration value="1.7"/>
            <xs:enumeration value="1.8"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="traceConfigUpdateType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Requests updates of the accelerator tracing configuration.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="removeTraceProfile" type="dwa:traceProfileNameType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Remove the specified trace profiles from the accelerator. Note: it
            is possible to first
            remove
            a trace profile and then add a new profile with the same name in
            one configuration update.
            If the profile does not exist, the remove command will be ignored
            with a warning
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="addTraceProfile" type="dwa:traceProfileType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Add the specified trace profiles to the accelerator. A profile with
            the given name must not
            yet"
            exist on the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="activateTraceProfile" type="dwa:traceProfileNameType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Activate the specified trace profile. Note: it is possible to first
            add a profile and then
            activate it in one configuration update.
            Specify DEFAULT as profile name to reset trace settings to
            defaults. Activating a trace profile
            other than DEFAULT may compromise system performance.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="getTraceDataEnum">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:complexType name="getTraceDataType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Requests the collection of trace data from the accelerator. The
        generated trace
        archive is a gzipped, structured tar archive, intended for problem
        analysis by IBM support.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="content" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:simpleContent>
            <xs:extension base="dwa:getTraceDataEnum">
              <xs:attribute name="component" type="xs:string" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -component options
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
              <xs:attribute name="spec" type="xs:string" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -component component
                    specification
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
              <xs:attribute name="symptom" type="xs:string" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -symptom specification
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
              <xs:attribute name="days" type="xs:integer" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -days specification
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
              <xs:attribute name="from" type="xs:dateTime" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -from specification
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
              <xs:attribute name="to" type="xs:dateTime" use="optional">
                <xs:annotation>
                  <xs:documentation xml:lang="en">
                    IBM support use only. nzlogcollector -to specification
                  </xs:documentation>
                </xs:annotation>
              </xs:attribute>
            </xs:extension>
          </xs:simpleContent>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="outputLocation" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Output location where the trace archive should be stored, as HSF file
          name. If not
          provided, the trace archive is returned in the stored procedure
          result set.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="keepConfiguration" type="xs:boolean" default="true"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          If false, reset the trace profile to DEFAULT after traces have been
          collected.
          If true, keep current trace profile.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="sinceLastSaveOnly" type="xs:boolean" default="false"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          If false, then all additional support documents will be download.
          If true, only new additional support documents will be downloaded
          since the last save.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="cancelTasksType">
    <xs:sequence>
      <xs:element name="task" type="dwa:taskIdentifierType" minOccurs="1"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="getAdditionalSupportType">
    <xs:attribute name="planID" type="xs:unsignedLong" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Plan ID for specific documents that need to be collected
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="traceCommand" type="dwa:traceCommandType"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The type of command to run on the backend for document collection
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="timeoutInMinutes" type="xs:unsignedLong" default="0">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Maximum wait time in minutes. If the timeout is exceeded, the command
          will end
          and return a partial trace. If the value is zero or not specified,
          the default timeout will be set to 30 minutes
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="profileSchema" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The profile schema to be ran with the DB2 support trace
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="profileName" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The profile name to be ran with the DB2 support trace
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="topQueries" type="xs:unsignedLong" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The number of queries for db2 explain from section extraction
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:simpleType name="traceCommandType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="CDE_PERFORMANCE_TRACE"/>
      <xs:enumeration value="DB2SUPPORT"/>
      <xs:enumeration value="SYSTEM_PERFORMANCE_DATA"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="getInfoTasksType">
    <xs:attribute default="false" name="includeEncryptionInfo"
        type="xs:boolean" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
                   If true, return information about encryption in the
                   accelerator info.
                   This includes certificates and active tunnels for encryption
                   of data in motion (IPsec).
                   If true, keep current trace profile.
                </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="activationLogType">
    <xs:attribute default="false" name="diagnostics" type="xs:boolean"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
                   If true, return additional diagnostics software update
                   activation logs, for example the
                   contents of apupgrade.log.tracelog and overall_status_log.
                </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="controlResult">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The result of an accelerator command depends on the command that
            was specified
            in the stored procedure input. See the controlCommand element.
          </xs:documentation>
        </xs:annotation>
        <xs:element name="acceleratorInfo" type="dwa:infoType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Reports accelerator status information if a getAcceleratorInfo
              command was specified.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="traceConfig" type="dwa:traceConfigType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Reports the accelerator trace configuration if a getTraceConfig
              command was specified.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="acceleratorTasks" type="dwa:acceleratorTasksType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Reports currently running accelerator tasks if a
              getAcceleratorTasks command was specified.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="replicationEvents"
            type="dwa:replicationStatusMessages">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Reports replication event messages of the accelerator tasks if a
              getReplicationEvents command was specified.
              Added with version 1.1 of this schema.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="additionalSupport" type="dwa:additionalSupportType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Activate backend document collection.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
            <xs:enumeration value="1.2"/>
            <xs:enumeration value="1.3"/>
            <xs:enumeration value="1.4"/>
            <xs:enumeration value="1.5"/>
            <xs:enumeration value="1.6"/>
```

```
            <xs:enumeration value="1.7"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="infoType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Information about the current state and configuration of the
        accelerator.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="versionInformation" type="dwa:versionInformationType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists the version information for components active in the
            accelerator.
            The following versions will always be present: Accelerator,
            Netezza, HPF, FDT,
            Access Server, Replication Engine, Netezza Log Collector, and Linux
            Operating System.
            Added with version 1.5 of this schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="acceleratorSetting" type="dwa:acceleratorSettingType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists the specific settings applied when querying data.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="replicationInfo" type="dwa:replicationInfoType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about replication on the accelerator. This element is
            only present if replication
            has been configured on the accelerator.
            Added with version 1.1 of this schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="procedurePackages" type="dwa:procedurePackagesType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists the procedure packages that exist in accelerator, together
            with an
            indicator if the package is enabled for the DB2 subsystem.
            Added with version 1.6 of this schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element maxOccurs="1" minOccurs="0" name="encryptionInformation"
          type="dwa:encryptionInformationType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists information about encryption active on the accelerator.
            Added with version 1.6 of this schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="state" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The current status of the accelerator.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="UNKNOWN">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The server status can not be determined.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Initializing">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The server is currently starting up and is not ready to receive
                work.
                No query processing can occur.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Online">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The server is fully operational and can receive work.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Offline">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The accelerator server is online but the database is not
                available.
                No query processing can occur. XSD_DOCUMENT_VERSION14("")

              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Maintenance">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                A maintenance operation (e.g. Software Update) is currently
                running.
                No query processing can occur
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Paused">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Deprecated. This value is used by accelerator versions prior to
                V4.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Stopped">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Deprecated. This value is used by accelerator versions prior to
                V4.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Down">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Deprecated. This value is used by accelerator versions prior to
                V4.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Unknown">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Deprecated. This value is used by accelerator versions prior to
                V4.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="Initialized">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Deprecated. This value is used by accelerator versions prior to
                V4.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="activeTraceProfile" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
```

```
          The currently active trace profile on the accelerator. More
          information about the trace
          configuration can be retrieved with the getTraceConfig command in the
          controlCommand element
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeVersion" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of the accelerator software.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Accelerator" in XML element 'versionInformation' instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeNetezzaVersion" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of Netezza database management system
          used by the accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Netezza" in XML element 'versionInformation' instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeFDTVersion" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of Netezza firmware (FDT) used by the
          accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "FDT" in XML element 'versionInformation' instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeHPFVersion" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of Netezza platform (HPF) used by the
          accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "HPF" in XML element 'versionInformation' instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeAccessServerVersion" type="xs:string"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of the Access Server used by the
          accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Access Server" in XML element 'versionInformation'
          instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeReplicationEngineVersion" type="xs:string"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The currently active version of the Replication Engine used by the
          accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Replication Engine" in XML element 'versionInformation'
          instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="authenticationTokenTimestamp" type="xs:dateTime"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The timestamp when the authentication token for this accelerator was
          last modified.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="currentAcceleratorTimestamp" type="xs:dateTime"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The current timestamp of the accelerator. Normally, accelerator time
          is synchronized with DB2 time;
          if multiple DB2 systems with differing times are paired with the same
          accelerator, the accelerator
          time may differ from DB2.
          Added with version 1.1 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="serialNumber" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
              The serial number of the accelerator.
          Added with version 1.5 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="traceConfigType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Reports the active accelerator tracing configuration.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="traceProfile" type="dwa:traceProfileType" minOccurs="1"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists the specifications of all available trace profiles.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="activeTraceProfile" type="dwa:traceProfileNameType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Specifies the currently active trace profile.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="acceleratorTasksType">
    <xs:sequence>
      <xs:element name="task" type="dwa:taskType" minOccurs="0"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="additionalSupportType">
    <xs:attribute name="planID" type="xs:unsignedLong" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Plan ID for specific documents that need to be collected, not
          required for DB2EXPLANSECTION
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="absoluteAdditionalSupportZipFilePath" type="xs:string"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          File path on the server where the trace is located
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="fileSize" type="xs:unsignedLong" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Size of the files saved onto the server
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="replicationInfoType">
    <xs:sequence>
```

```
      <xs:element name="sourceAgent" type="dwa:replicationAgentStatisticType"
          minOccurs="1" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Replication statistics for the source agent that captures updates
            on DB2.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="targetAgent"
          type="dwa:replicationAgentStatisticTypeExtended" minOccurs="1"
          maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Replication statistics for the target agent that applies updates on
            the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="state" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Current status of replication for the accelerator.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="lastChangeTimestamp" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Last timestamp when replication was explicitly started or stopped.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="latencyInSeconds" type="xs:long" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Approximate time it takes for an update in the DB2 source table until
          it is replicated on the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeAccessServerVersion" type="xs:string"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Current software version of the Access Server replication component
          installed on the accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Access Server" in XML element 'versionInformation'
          instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="activeReplicationEngineVersion" type="xs:string"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Current software version of the Replication Engine replication
          component installed on the accelerator.
          This XML attribute has been deprecated. Refer to version information
          for
          component "Replication Engine" in XML element 'versionInformation'
          instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="encryptionInformationType">
    <xs:sequence>
      <xs:element maxOccurs="1" minOccurs="1" name="encryptionOfDataInMotion"
          type="dwa:encryptionOfDataInMotionType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about network encryption on this accelerator including
            certificates and active tunnels.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="encryptionOfDataInMotionType">
    <xs:sequence>
      <xs:element maxOccurs="1" minOccurs="0" name="certificates"
          type="dwa:certificatesType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about network encryption certificates configured on
            this accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element maxOccurs="1" minOccurs="0" name="peers"
          type="dwa:peersType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about active network encryption tunnels on this
            accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="certificatesType">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="certificate"
          type="dwa:certificateType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about network encryption certificates configured on
            this accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="certificateType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="friendlyName" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:complexType name="peersType">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="peer"
          type="dwa:peerType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about encrypted network connections configured on this
            accelerator
            including active tunnels.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="peerType">
    <xs:sequence>
      <xs:element maxOccurs="unbounded" minOccurs="0" name="tunnel"
          type="dwa:tunnelType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
          Information about a single encrypted network tunnel active on this
          accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
        The user-defined name associated with the z/OS remote IP address, for
        example the
        LPAR name.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="remoteIP" type="xs:string" use="required">
      <xs:annotation>
```

```
        <xs:documentation xml:lang="en">
        The z/OS remote IP address, for which this peer is configured
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="cert" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
        The certificate used to authenticate the accelerator in this
        connection.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="tunnelType">
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="localIP" type="xs:string" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
          The IP address used on the accelerator for this tunnel.
          </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="since" type="xs:dateTime" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
          The date/time when this tunnel was established.
          </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="inBytes" type="xs:integer" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
          Number of bytes flowing through this tunnel since it was established.
          </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="outBytes" type="xs:integer" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
          Number of bytes flowing through this tunnel since it was established.
          </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
  <xs:element name="diagnosticCommand">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="ping" type="dwa:testPingType" minOccurs="0"
            maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Perform a ping from the z/OS host to an accelerator IP address,
              using the same TCP/IP
              configuration like the stored procedure.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="connect" type="dwa:testConnectType" minOccurs="0"
            maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Test the connection to the accelerator, using the same handshake
              mechanism like any other
              accelerator procedure. If this test succeeds all other procedures
              are also capable to connect to the accelerator
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="mtupath" type="dwa:testMTUPathType" minOccurs="0"
            maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Test max segment size along the network path from the z/OS
              machine to the accelertor.
              All parts of the network should be configured to the same size
              and use jumbo frames for max performance.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="networkSpeed" type="dwa:testNetworkSpeedType"
            minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Test the speed of the network between the accelerator and the
              z/OS host, using the same
              TCP/IP configuration and code like the stored procedures. This
              can be used to determine network bottlenecks.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="version" type="dwa:testVersionType" minOccurs="0"
            maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Connect to the accelerator server without authenticating and just
              return the
              descriptive meta-information about the server and stored
              procedure, for example version,
              protocol version and capabilities.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="testPingType">
    <xs:attribute name="ip" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The IP address of the host to ping. This should be the IP of the
          accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="testConnectType">
    <xs:attribute name="accelerator" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The name of the accelerator to test the connectivity. This is the
          same name like the input to
          many stored procedures.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="testMTUPathType">
    <xs:attribute name="ip" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The IP address of the host to test the max segment size with. This
          should be the IP of the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="port" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The port of the host where the accelerator service is waiting for
          connections.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="testVersionType">
    <xs:attribute name="accelerator" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The name of the accelerator to test the connectivity. This is the
          same name like the input to
          many stored procedures. If the accelerator is not yet paired, use the
          ip and port attributes instead.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ip" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The IP address of the hostname to be connected to. This should be the
          IP of the accelerator.
          The IP address/port pair can be used instead of the accelerator name,
          if the accelerator is
          not yet paired with this DB2 subsystem.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="port" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The port of the host where the accelerator service is waiting for
          connections.
          The IP address/port pair can be used instead of the accelerator name,
          if the accelerator is
          not yet paired with this DB2 subsystem.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="testNetworkSpeedType">
    <xs:attribute name="accelerator" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The name of the accelerator to test the network speed against. This
          is the same name like the input to
          many stored procedures.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="seed" type="xs:unsignedInt" default="12345678"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          A seed for the data generated randomly during the networks speed
          test. The seed allows to generated
          the same data for the tests.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="totalNumberBytes" type="xs:unsignedLong"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The total amount of bytes transfered during the test per connection.
          The real amount of data transfered is multiplied with the number of
          parallelConnections.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="dataBatchSize" type="xs:unsignedInt" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The size of data chunks in bytes for the test. A good value is 320000
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="parallelConnections" type="xs:unsignedByte" default="1"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The number of parallel connections (threads) used to transfer the
          data.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="diagnosticOutput">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="diagnostics" type="dwa:diagnosticsElementType"
            minOccurs="0" maxOccurs="unbounded">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Contains diagnostic information in plain text for every
              diagnostic action performed by
              the stored procedure.
              Since Version 1.1: In case of "version" action it contains nested
              elements
              describing the version of the accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="diagnosticsElementType" mixed="true">
    <xs:sequence>
      <xs:element name="accelerator" type="dwa:diagnosticsVersionType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
              Returns version information and capabilities of this accelerator.
            </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="storedProcedure" type="dwa:diagnosticsVersionType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
              Returns version information and capabilities of the accelerator
              administrative stored procedures ACCEL_...
            </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="action" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
              The name of the action that was performed by the stored
              procedure.
            </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="success" type="xs:boolean" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
              Flag whether the diagnostic action succeeded or had an error.
            </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="diagnosticsVersionType">
    <xs:sequence>
      <xs:element name="versionInformation" type="dwa:versionInformationType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
          Key value pairs name/version of version information for the different
          components in the
          accelerator server or stored procedures.
        </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="DRDAProtocolLevel" type="xs:string" minOccurs="0"
          maxOccurs="1"/>
      <xs:element name="highestSupportedCompatibilityLevel" type="xs:string"
          minOccurs="0" maxOccurs="1"/>
      <xs:element name="setting" type="dwa:acceleratorSettingType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Lists the specific capabilities and settings for the accelerator
            server or
            stored procedures.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
```

```
<xs:element name="federatedTableSetInput">
  <xs:complexType>
    <xs:sequence>
      <xs:annotation>
        <xs:documentation xml:lang="en">
          List of all DB2 tables that should be processed by the stored
          procedure with optional qualifiers
          such as if the table is valid in the DB2 subsystem and the AOT name
          of the table for a
          referencing DB2 subsystem.
        </xs:documentation>
      </xs:annotation>
      <xs:element name="table" type="dwa:federatedTableInputType"
          minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="1.0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:element name="federatedTableSetOutput">
  <xs:complexType>
    <xs:sequence>
      <xs:annotation>
        <xs:documentation xml:lang="en"/>
      </xs:annotation>
      <xs:element name="table" type="dwa:federatedTableOutputType"
          minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="1.0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:element name="messageControl">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="compatibilityLevel" type="xs:string" minOccurs="0"
          maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Requests that the stored procedure should maintain compatible
            behavior for a client that was
            written against the specified interface level. In particular, XML
            output of the stored procedure
            will not include any tags or attributes that were introduced in a
            later stored procedure
            interface to avoid breaking clients that perform XML validation
            against a back-level XML schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="language" type="xs:string" minOccurs="0"
          maxOccurs="1" default="en_US">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The language in which localized stored procedure output texts
            should be generated.
            Currently, only english is supported and this value is ignored.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="traceConfig" type="dwa:spTraceConfigType"
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The trace configuration that should be used for this invocation
            of a stored procedure.
            Specific trace configurations will by provided by IBM support if
            necessary.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="environment" type="dwa:spEnvironmentType"
```

```
          minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The environment variables that are defined for all IDAA
            procedures can be overwritten per call.
            Since the procedure already is active, resetting some environment
            variables may not have an effect.
            (e.g. variables for Language Environment). Intended usage is for
            overriding IDAA variables defined in AQTENV.
            Added with version 1.2 of messageControl.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="versionOnly" type="xs:boolean" use="optional"
        default="false"/>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="1.0"/>
          <xs:enumeration value="1.1"/>
          <xs:enumeration value="1.2"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:complexType name="spTraceConfigType">
  <xs:sequence>
    <xs:element name="component" type="dwa:spTraceComponentType"
        minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="traceFileSizeInMB" type="xs:unsignedInt" default="1"
      use="optional"/>
  <xs:attribute name="location" type="xs:string" use="optional"
      default="GENERATED"/>
  <xs:attribute name="keepTrace" type="dwa:spKeepTraceType" default="false"
      use="optional"/>
  <xs:attribute name="forceFlush" type="xs:boolean" default="false"
      use="optional"/>
</xs:complexType>
<xs:complexType name="spTraceComponentType">
  <xs:attribute name="name" type="xs:string" use="required"/>
  <xs:attribute name="level" type="dwa:spTraceLevelType" use="required"/>
</xs:complexType>
<xs:simpleType name="spTraceLevelType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="OFF"/>
    <xs:enumeration value="ERROR"/>
    <xs:enumeration value="WARN"/>
    <xs:enumeration value="INFO"/>
    <xs:enumeration value="DEBUG"/>
    <xs:enumeration value="TRACE"/>
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="spKeepTraceType">
  <xs:union memberTypes="xs:boolean">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="onError"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:union>
</xs:simpleType>
<xs:complexType name="spEnvironmentType">
  <xs:sequence>
    <xs:element name="environmentVariable" type="dwa:spEnvVarType"
        minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="spEnvVarType">
  <xs:attribute name="name" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Name of an environment variable.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
  <xs:attribute name="value" type="xs:string" use="required">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Value of the environment variable.
      </xs:documentation>
    </xs:annotation>
  </xs:attribute>
</xs:complexType>
<xs:element name="messageOutput">
  <xs:complexType>
    <xs:sequence>
      <xs:annotation>
        <xs:documentation xml:lang="en">
          List of all stored procedure output messages, warnings and errors
        </xs:documentation>
      </xs:annotation>
      <xs:element name="message" type="dwa:messageType" minOccurs="1"
          maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="1.0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:complexType name="messageType">
  <xs:sequence>
    <xs:element name="text" type="xs:string" minOccurs="1" maxOccurs="1"/>
    <xs:element name="description" type="xs:string" minOccurs="0"
        maxOccurs="1"/>
    <xs:element name="action" type="xs:string" minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="severity" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="informational"/>
        <xs:enumeration value="warning"/>
        <xs:enumeration value="error"/>
        <xs:enumeration value="severe"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="reason-code" type="xs:string" use="required"/>
</xs:complexType>
<xs:element name="queryList">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="query" type="dwa:queryShortInformationType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            List of queries executed on the accelerator with associated
            information.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="1.0"/>
          <xs:enumeration value="1.1"/>
          <xs:enumeration value="1.2"/>
          <xs:enumeration value="1.3"/>
          <xs:enumeration value="1.4"/>
          <xs:enumeration value="1.5"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
</xs:element>
<xs:complexType name="queryShortInformationType">
  <xs:annotation>
    <xs:documentation xml:lang="en">
      Accelerator information for an offloaded query.
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="clientInfo" type="dwa:queryClientInformationType"/>
    <xs:element name="execution" type="dwa:queryExecutionInformationType"/>
    <xs:element name="task" type="dwa:taskIdentifierType" minOccurs="0"
        maxOccurs="1">
      <xs:annotation>
      <xs:documentation xml:lang="en">
       The accelerator task that is executing the query.
      </xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="sql" type="xs:string">
     <xs:annotation>
      <xs:documentation xml:lang="en">
       The first 128 bytes of the SQL text of the query. Longer queries
       are truncated at character boundaries.
      </xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
   <xs:attribute name="user" type="xs:string" use="required">
    <xs:annotation>
     <xs:documentation xml:lang="en">
      The DB2 user ID that submitted the query
     </xs:documentation>
    </xs:annotation>
   </xs:attribute>
   <xs:attribute name="planID" type="xs:string" use="required">
    <xs:annotation>
     <xs:documentation xml:lang="en">
      The accelerator plan ID that can be used to retrieve the full SQL
      query text
      and detailed plan information.
     </xs:documentation>
    </xs:annotation>
   </xs:attribute>
   <xs:attribute name="sessionID" type="xs:string" use="optional">
    <xs:annotation>
     <xs:documentation xml:lang="en">
      The accelerator session ID that can be used to retrieve the full SQL
      query text
      and detailed plan information.
     </xs:documentation>
    </xs:annotation>
   </xs:attribute>
  </xs:complexType>
  <xs:complexType name="queryClientInformationType">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     Client info provided by the application and passed on by DB2. All
     attributes are optional and
     are only provided if set by the client application.
    </xs:documentation>
   </xs:annotation>
   <xs:sequence>
    <xs:element name="accountingInformation"
      type="dwa:accountingInformationType" minOccurs="0"
      maxOccurs="unbounded">
     <xs:annotation>
      <xs:documentation xml:lang="en">
       Generic list of account information available for the query.
      </xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
   <xs:attribute name="productID" type="xs:string" use="optional">
    <xs:annotation>
     <xs:documentation xml:lang="en">
      Identification of the product that routed the query, like a JDBC
      driver.
     </xs:documentation>
    </xs:annotation>
   </xs:attribute>
   <xs:attribute name="user" type="xs:string" use="optional">
    <xs:annotation>
     <xs:documentation xml:lang="en">
      The client-side user ID that submitted the query.
     </xs:documentation>
    </xs:annotation>
   </xs:attribute>
   <xs:attribute name="workstation" type="xs:string" use="optional"/>
   <xs:attribute name="application" type="xs:string" use="optional"/>
   <xs:attribute name="packageName" type="xs:string" use="optional"/>
   <xs:attribute name="collectionID" type="xs:string" use="optional"/>
   <xs:attribute name="networkID" type="xs:string" use="optional"/>
   <xs:attribute name="locationName" type="xs:string" use="optional"/>
   <xs:attribute name="luName" type="xs:string" use="optional"/>
   <xs:attribute name="connName" type="xs:string" use="optional"/>
```

```
    <xs:attribute name="connType" type="xs:string" use="optional"/>
    <xs:attribute name="corrID" type="xs:string" use="optional"/>
    <xs:attribute name="authID" type="xs:string" use="optional"/>
    <xs:attribute name="planName" type="xs:string" use="optional"/>
    <xs:attribute name="accounting" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Optional client specific accounting information that the client
          application provides.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="dataSharingGroupName" type="xs:string" use="optional"/>
    <xs:attribute name="dataSharingMemberName" type="xs:string"
        use="optional"/>
    <xs:attribute name="subSystemID" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          SSID of the DB2 subsystem or member in the data sharing group. The
          value is derived from
          the client accounting information if available.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="queryExecutionInformationType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Information about execution of the query on the accelerator
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="state" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The execution state of the query.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="UNKNOWN">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                This value is deprecated and will no longer be used.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="PENDING">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                This value is deprecated and will no longer be used.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="QUEUED">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The query is waiting to be executed. The accelerator is busy
                processing other requests.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="RUNNING">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The query is currently being executed, but no result rows have
                been sent to DB2 yet.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="FETCHING">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The query is currently being executed and result rows are being
                sent to DB2, or
                query execution has already finished and only remaining result
                rows are sent to DB2.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="DONE">
            <xs:annotation>
              <xs:documentation xml:lang="en">
```

```
                The query completed successfully and no error occurred.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="ABORTED">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The query terminated with a failure.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="CANCELED">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The query execution was canceled, either by canceling the
                accelerator's task or by
                prematurely closing the connection from DB2 to the accelerator.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="submitTimestamp" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The accelerator timestamp at which the query was submitted to the
          accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="elapsedTimeSec" type="xs:unsignedInt" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The total processing time of the query on the accelerator (including
          wait time).
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="executionTimeSec" type="xs:unsignedInt" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The actual execution time of the query in the Netezza backend
          database system of the accelerator.
          The value may include wait time if the query gets queued in Netezza.
          It does not include preprocessing
          time before the query is sent to Netezza, and it also does not
          include fetch time required after Netezza
          has finished its processing.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="waitTimeSec" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The time required for preprocessing the query in the accelerator
          (before it can be sent to the Netezza
          backend database system) and any queue wait time from Netezza itself.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="fetchTimeSec" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The time measured from receiving the first row from the Netezza
          backend database system until
          the last row was requested by DB2 and send to DB2. The fetch time
          usually overlaps with the
          execution time because the first row is received from Netezza before
          query execution in Netezza
          can terminate.
          The value is provided only if fetching result rows from the the
          Netezza backend database system "
          has already started.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="cpuTimeSec" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          CPU time required for executing the query in the Netezza backend
          database system and on the
          accelerator.
          The value is provided only if CPU time information was already
          collected in the accelerator and
          from the Netezza backend database system.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="priority" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The priority of the query execution on the accelerator. The attribute
          is only set if an
          explicit priority was associated with the query received from DB2.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="resultRows" type="xs:unsignedLong" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Number of rows returned by the query. This value is an estimate if
          the query is not completed.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="resultBytes" type="xs:unsignedLong" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Number of bytes returned by the query. This value is an estimate if
          the query is not completed.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="progressPercent" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          This attribute is deprecated and it will no longer be populated by
          the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="sqlState" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          SQL state returned by the query. This attribute is only provided when
          the query state is ABORTED.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="sqlCode" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          SQL code returned by the query. This attribute is only provided after
          the query completed in the
          Netezza backend database system. Query result processing in the
          accelerator may still be ongoing.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="errorDescription" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Description of the SQL error in case the query did not complete
          successfully.
          If the query succeeded, the attribute is omitted.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="waitTimeHTAPSec" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The amount of time spent waiting for completion of data replication
          to the accelerator using the
                  delay protocol (HTAP) before it can be sent to the backend
                  database system
          The value is provided only if time waited for Delay Protocol
          information was already collected in
          the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="timeoutsHTAP" type="xs:unsignedLong" use="optional">
      <xs:annotation>
```

```
        <xs:documentation xml:lang="en">
          The number of requests where the replication of data to the
          accelerator needed to execute
                  the query did not complete before the delay protocol time
                  limit expired (HTAP).
          The value is provided only if timeouts for Delay Protocol information
          was already collected in
          the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="prepareTimeSec" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The time measured from prepare time
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="applicationStallTime" type="xs:unsignedInt"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The total continue query wait time. This is the total time taken to
          resume
          the return of the result set generated from the open query. This is
          when the application
          is stalled and known as the application stall time.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="accountingInformationType">
    <xs:attribute name="name" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Name identifying the parameter.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="value" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Value applied for the parameter.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="querySelection">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="filter" type="dwa:querySelectionFilterType"
            minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Optional list of selection conditions to filter queries
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="result" type="dwa:querySelectionResultType"
            minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Optional sorting criteria and maximum size for the result list
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="querySelectionFilterType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Defines selection conditions to filter queries. All conditions are
        optional,
        queries are returned only if they match all given filter conditions.
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="scope" default="all" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Defines the scope of queries that are returned; allows to select only
          queries that are
          currently executing on the accelerator or queries that have completed
          execution.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="all"/>
          <xs:enumeration value="active"/>
          <xs:enumeration value="completed"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="fromTimestamp" type="xs:dateTime" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only queries that have been submitted after the given
          timestamp
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="toTimestamp" type="xs:dateTime" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only queries that have been submitted before the given
          timestamp
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="minElapsedTimeSec" type="xs:unsignedInt"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only queries whose elapsed time is at least the given number
          of seconds
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="minExecutionTimeSec" type="xs:unsignedInt"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only queries whose execution time is at least the given number
          of seconds
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="user" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only queries that were submitted by the given DB2 user ID
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="querySelectionResultType">
    <xs:attribute name="order" default="submitTimestamp" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Defines by which criterion the returned queries should be sorted and
          therefore
          which queries should be left out if the maxRows attribute is
          specified.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="submitTimestamp">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending submit timestamp (later queries first)
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="elapsedTime">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending elapsed time (longer running queries first)
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="executionTime">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending execution time (longer running queries
                first)
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="queueTime">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending queue time (difference between elapsed and
                execution time,
                longer queue time first)
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="resultRows">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending number of result rows
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="resultBytes">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Sort by descending number of result bytes
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="maxRows" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Specifies the maximum number of queries to return
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="softwareUpdate">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="deployPackage"
            type="dwa:SoftwareUpdateDeployPackageType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Transfer a single package to the accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="listZPackageDirectory"
            type="dwa:SoftwareUpdateListZPackageDirectoryType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              List packages in a USS file system directory on the z/OS host the
              SP is executed on.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="listDeployedPackages"
            type="dwa:SoftwareUpdateListDeployedPackagesType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              List binary files (aka packages) deployed to an accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="removeDeployedPackages"
            type="dwa:SoftwareUpdateRemoveDeployedPackagesType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Remove multiple packages (that were not installed yet) from the
              accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="activateDeployedPackage"
            type="dwa:SoftwareUpdateActivateDeployedPackageType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Activate a single package that was not installed yet on the
              accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="SoftwareUpdateDeployPackageType">
    <xs:attribute name="source" type="dwa:PackageSourceType" use="required"/>
    <xs:attributeGroup ref="dwa:PackageIdentifyingAttributes"/>
    <xs:attribute name="fileSizeInBytes" type="xs:integer" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file size of the binary package in bytes.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="hash" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The hash for the package computed with the algorithm denoted by the
          algorithm attribute.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="algorithm" type="dwa:HashAlgorithmType" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The algorithm used to compute the hash of the package.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="SoftwareUpdateListZPackageDirectoryType"/>
  <xs:complexType name="SoftwareUpdateListDeployedPackagesType"/>
  <xs:complexType name="SoftwareUpdateRemoveDeployedPackagesType">
    <xs:sequence>
      <xs:element name="package"
          type="dwa:SoftwareUpdateRemoveSinglePackageType" minOccurs="0"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The package file to be removed from the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="SoftwareUpdateActivateDeployedPackageType">
    <xs:attributeGroup ref="dwa:PackageIdentifyingAttributes"/>
    <xs:attribute name="essentialsOnly" type="xs:boolean" default="false"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          If true, skip firmware and update only accelerator components - which
          is faster.
          If false, update all available components - including firmware.
          Added with version 1.1 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="SoftwareUpdateRemoveSinglePackageType">
    <xs:attributeGroup ref="dwa:PackageIdentifyingAttributes"/>
  </xs:complexType>
  <xs:simpleType name="PackageSourceType">
    <xs:restriction base="xs:string">
```

```
      <xs:enumeration value="GTT">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The binary file is transferred via a Global Temporary Table
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="FILE">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The binary file is transferred from a USS file system
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="HashAlgorithmType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="SHA256">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The algorithm used to compute a hash is sha256.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="SHA1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The algorithm used to compute a hash is sha1.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="MD5">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The algorithm used to compute a hash is md5.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:attributeGroup name="PackageIdentifyingAttributes">
    <xs:attribute name="fileName" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file name defining the binary package.
          It must be the name under which it is downloadable from FixCentral
          servers
          and serves as an ID of the package.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:element name="softwareUpdateResult">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="ZPackageDirectoryListing"
            type="dwa:ZPackageDirectoryListingType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Returned directory listing for deployable packages on a z/OS USS
              file system
              if the input command was listZPackageDirectory
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="deployedPackagesListing"
            type="dwa:DeployedPackagesListingType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Returned listing for deployed packages on an accelerator
              if the input command was listDeployedPackages
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="ZPackageDirectoryListingType">
    <xs:sequence>
      <xs:element name="deployablePackage"
          type="dwa:SoftwareUpdateDeployPackageType" minOccurs="0"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The details for a package file residing in the z/OS USS file system
            directory.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="directoryName" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The directory name with the binary packages.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="DeployedPackagesListingType">
    <xs:sequence>
      <xs:element name="deployedPackage"
          type="dwa:SoftwareUpdateDeployedPackageType" minOccurs="0"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The details for a package file residing in the z/OS USS file system
            directory.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="SoftwareUpdateDeployedPackageType">
    <xs:sequence>
      <xs:element name="activationImpact"
          type="dwa:SoftwareUpdateActivationImpactType" minOccurs="0"
          maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The details describing the impact when activating the package.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="fileName" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file name defining the binary package.
          It must be the name under which it is downloadable from FixCentral
          servers
          and serves as an ID of the package
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="fileSizeInBytes" type="xs:integer" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file size of the binary package in bytes.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="deploymentTimestamp" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The timestamp of the deployment (transfer) of the package to the
          accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="requiresManualInstall" type="xs:boolean"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Indicator, if the installation of the package requires manual
          intervention (rather than just calling a stored procedure)
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="SoftwareUpdateActivationImpactType">
    <xs:sequence>
      <xs:element name="impactDescription" type="xs:string" minOccurs="0"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            formatted input (CDATA) with a detailed impact description.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="estimatedTimeForActivationInMinutes" type="xs:integer"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The estimated time needed for activation of the package
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:element name="softwareMaintenanceCommand">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="listAvailablePackages"
            type="dwa:listAvailablePackagesType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              List all accelerator-related packages that are available on the
              z/OS system and can be
              deployed to the accelerator with the deployPackages command. This
              includes software
              updates for the accelerator itself as well as software updates
              for the underlying Netezza system.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="deployPackages" type="dwa:deployablePackageListType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Transfer one or more of the available packages to the
              accelerator.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="getDeployedPackageInformation"
            type="dwa:getDeployedPackagesType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              List the packages that have been deployed to the accelerator and
              to which the products
              can be upgraded or downgraded using the activateDeployedPackage
              command.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="activateDeployedPackage"
            type="dwa:deployedPackageType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Activate a deployed package on the accelerator and upgrade or
              downgrade the product
              software level to the version of the selected package
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="removeDeployedPackages"
            type="dwa:deployedPackageListType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Remove one or more deployed packages from the accelerator.
              Added with version 1.1 of this schema.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="deployedPackageType">
    <xs:attribute name="target" type="dwa:packageTargetType"
        default="ACCELERATOR">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The target that defines the type of the software product that should
          be activated or removed.
          Added with version 1.1 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="version" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The software component version that should be activated
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="fileName" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file name defines the name of a file.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="deployedPackageListType">
    <xs:sequence>
      <xs:element name="package" type="dwa:deployedPackageType" minOccurs="1"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="listAvailablePackagesType">
    <xs:complexContent>
      <xs:extension base="dwa:packageTargetListType">
        <xs:attribute name="acceleratorInstallPrefix" type="xs:string"
            use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Deprecated.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="hostPackageDirectory" type="xs:string"
            use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Allows to specify a directory which should be searched for
              available packages.
              If the attribute is not set, the value of
              AQT_HOST_PACKAGE_DIRECTORY from the AQTENV data set is used.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="getDeployedPackagesType">
    <xs:complexContent>
      <xs:extension base="dwa:packageTargetListType">
        <xs:attribute name="scope" type="dwa:getDeployedPackagesScope"
            default="ACTIVATE">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Allows to specify the scope of packages that should be listed.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:simpleType name="getDeployedPackagesScope">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ACTIVATE">
        <xs:annotation>
          <xs:documentation xml:lang="en">
```

```
            Show the deployed packages that can be activated.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="REMOVE">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Show the deployed packages that can be removed.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="packageTargetListType">
    <xs:sequence>
      <xs:element name="target" type="dwa:packageTargetType" minOccurs="0"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Defines the target types for which available or deployed packages
            should be listed.
            If no element is present, all package types are listed.
            Added with version 1.1 of this schema.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="package">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="information" type="dwa:informationInternalType"/>
        <xs:element name="files" type="dwa:filesType" minOccurs="0"/>
        <xs:element name="migration" type="dwa:migrationType" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="informationInternalType">
    <xs:complexContent>
      <xs:extension base="dwa:informationType">
        <xs:sequence>
          <xs:element name="localizedDescription"
              type="dwa:localizedDescriptionType" minOccurs="0"
              maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="localizedDescriptionType">
    <xs:complexContent>
      <xs:extension base="dwa:descriptionType">
        <xs:attribute name="lang" type="xs:string" use="required"/>
        <xs:attribute name="simpleVersion" type="xs:string" use="optional"/>
        <xs:attribute name="buildLabel" type="xs:string" use="optional"/>
        <xs:attribute name="version" type="xs:string" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="filesType">
    <xs:sequence maxOccurs="unbounded">
      <xs:element name="entry" type="dwa:entryType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="entryType">
    <xs:attribute name="path" type="xs:string" use="required"/>
    <xs:attribute name="permission" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="migrationType">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="sourceProductVersion" type="dwa:sourceVersionType"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="softwareMaintenanceResult">
    <xs:complexType>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element name="availablePackages"
            type="dwa:deployablePackageListType">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Returned list of deployable packages on z/OS HFS if the input
              command
              was listAvailablePackages
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:sequence>
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Returned list of deployed packages on the accelerator if the
              input command
              was getDeployedPackageInformation
              Changed with version 1.1 of this schema
            </xs:documentation>
          </xs:annotation>
          <xs:element name="sourceProductVersion" type="dwa:sourceVersionType"
              minOccurs="0" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Information about currently active and other deployed versions
                for each product
                installed on the accelerator.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:choice>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="tableSetForArchiving">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            list of all tables that should be archived.
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableArchiveSpecificationType"
            minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="usePhysicalPartitionNumbers" type="xs:boolean"
          default="false">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Interprets all specified partition numbers as physical partition
            numbers.
            If the attribute is set to false or not used,
            the specified partition numbers are interpreted as logical
            partitions.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableArchiveSpecificationType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence minOccurs="0" maxOccurs="1">
          <xs:element name="partitions" type="dwa:partitionNumberListType">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Optional list of partitions to be archived. Default is to
                archive all partitions
                Partitions that are already archived are ignored
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="tableSetDetails">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            List of accelerated tables with associated detail information.
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableDetailsType" minOccurs="1"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
            <xs:enumeration value="1.2"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableDetailsType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:choice>
          <xs:sequence>
            <xs:element name="partInformation"
                type="dwa:partitionInformationType"/>
            <xs:element name="part" type="dwa:partitionWithDetailsType"
                minOccurs="1" maxOccurs="unbounded"/>
            <xs:element name="archivePartitionPredicate" type="xs:string"
                minOccurs="0" maxOccurs="1"/>
          </xs:sequence>
          <xs:element name="changeInformation"
              type="dwa:tableChangeInformationType">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Information about changes in an unpartitioned table.
                For partitioned tables, changes are reported at the partition
                level
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:choice>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="partitionWithDetailsType">
    <xs:sequence>
      <xs:element name="archiveInformation"
          type="dwa:tableArchiveInformationType" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Detail information about archived partitions
            This element is only provided if the partition has been archived
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="changeInformation"
          type="dwa:tableChangeInformationType" minOccurs="0" maxOccurs="1">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Information about changes in the partition
            This element may also be present if the partition has been
            archived; archived partitions
            should not be changed any more in DB2, therefore a change type
            other than NoChange
            indicates a potential consistency problem.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
```

```
    <xs:attributeGroup ref="dwa:sharedPartitionAttibutes"/>
  </xs:complexType>
  <xs:complexType name="tableChangeInformationType">
    <xs:attributeGroup ref="dwa:sharedChangeInformationAttibutes"/>
    <xs:attribute name="category" type="dwa:tableChangeCategoryType"
        use="required"/>
    <xs:attribute name="type" type="dwa:tableChangeType" use="required"/>
    <xs:attribute name="lastLoadTimestamp" type="xs:dateTime">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Timestamp when the partition was last loaded into the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="tableArchiveInformationType">
    <xs:complexContent>
      <xs:extension base="dwa:backupImageList">
        <xs:attribute name="archiveTimestamp" type="xs:dateTime">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Timestamp when the partition was archived.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="dataSizeInMB" type="xs:unsignedLong">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Approximate size of archived data.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:simpleType name="tableChangeCategoryType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        The categories of changes that can be detected in a table or partition.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="NONE">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            No changes in the table or partition were detected.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RELOAD_REQUIRED">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes were detected that require a reload of the table or
            partition.
            This occurs if the partitioning of the table is not in synch with
            the current data loaded on the accelerator, e.g. if a partition was
            added.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="UNKNOWN">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Change information about a table or partitions cannot be
            determined:
            The available information is insufficient or inconsistent.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RELOAD_RECOMMENDED">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes in table or partition data were detected but a reload is
            not
            required if the data differences are tolerable.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="tableChangeType">
    <xs:annotation>
```

```
      <xs:documentation xml:lang="en">
        Type of change that was detected
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="NoChange">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            real-time statistics were evaluated and show no change
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="DataChange">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            real-time statistics were evaluated and show
            changes that should be reloaded but may be deselected
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="LoadReplacePerformed">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The DB2 Load Utility was used to do a load replace on the table or
            partition and the
            load replace timestamp has changed
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NoRealtimeStatsInDBMS">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes that should be reviewed but do not have to be reloaded
            (potential false positives):
            no real-time statistics are available in DB2
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NoRealtimeStatsOnAccel">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes that should be reviewed but do not have to be reloaded
            (potential false positives):
            real-time statistics are available in DB2 but not the accelerator
            catalog;
            i.e. they were not available when the table or partition was loaded
            the last time.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RealtimeStatsInconsistent">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes that should be reviewed but do not have to be reloaded
            (potential false positives):
            real-time statistics are available but are not consistent, e.g.
            update counters have decreased
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ReorgWithNoSubsequentUpdate">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes that should be reviewed but do not have to be reloaded
            (potential false positives):
            reorg was detected but update counters are zero - unclear whether
            data was modified
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TableLoadedWithOldVersion">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Table was non-partitioned when added to the accelerator and was
            changed to
            partition-by-growth since then. The accelerator does not have the
            necessary
            meta-data for change detection.
            To enable change detection, the table needs to be removed from the
            accelerator
            and added again.
            This value is deprecated and only used for backward compatibility.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NonPartitionedTableChangedToByGrowth">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Table was non-partitioned when added to the accelerator and was
            changed to
            partition-by-growth since then. The accelerator does not have the
            necessary
            meta-data for change detection and the table needs to be fully
            reloaded.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="DataChangeInReplicatedTable">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Changes that should be reviewed but do not have to be reloaded
            (potential false positives):
            Real-time statistics were evaluated and show
            changes that should be reloaded but table is enabled for
            replication,
            so the changes may already have been propagated to the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TableNotLoaded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            table is not currently loaded on accelerator
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="PartitioningNotSynchronized">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            the list of partitions or limit keys in the DB2 catalog and the
            accelerator catalog differ -
            e.g. after REORG REBALANCE
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="PartitionAddedOrRotated">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            partition does not exist on accelerator
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TableDataExchanged">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            EXCHANGE TABLE DATA was performed (clone table)
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="PartitionedTableModifiedByReplication">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            This code is not used anymore by the current product version.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="TableCreatedWithOldVersion">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Table was non-partitioned when added to the accelerator and was
            changed to
            partition-by-growth since then. The accelerator does not have the
            necessary
            meta-data for change detection.
            To enable change detection, the table needs to be removed from the
            accelerator
            and added again.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="AcceleratorOnly">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            This is an accelerator-only table, for which no data changes in DB2
            can occur.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="tableInformation">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="table" type="dwa:tableInformationType" minOccurs="1"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
            <xs:enumeration value="1.2"/>
            <xs:enumeration value="1.3"/>
            <xs:enumeration value="1.4"/>
            <xs:enumeration value="1.5"/>
            <xs:enumeration value="1.6"/>
            <xs:enumeration value="1.7"/>
            <xs:enumeration value="1.8"/>
            <xs:enumeration value="1.9"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableInformationType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Accelerator-specific information about a table.
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence>
          <xs:element name="status" type="dwa:tableStatusType" minOccurs="1"
              maxOccurs="1"/>
          <xs:element name="statistics" type="dwa:tableStatisticsType"
              minOccurs="1" maxOccurs="1"/>
          <xs:element name="remote" type="dwa:remoteTableType" minOccurs="0"
              maxOccurs="1"/>
          <xs:element name="task" type="dwa:taskType" minOccurs="0"
              maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="tableStatusType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        General status information about an accelerated table.
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="loadStatus" type="dwa:tableLoadStatusType"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The load status defines which operations are possible with a table.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="type" type="dwa:tableTypeType" use="optional"
        default="Regular">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The table type defines if it is a regular accelerated table, an
          accelerator only table,
          or another (future) table type.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="accelerationStatus" type="xs:boolean" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The acceleration status defines whether a table can be used for query
          offloading.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="integrityStatus" type="dwa:tableIntegrityType"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The integrity status defines whether meta-data in DB2 and on the
          accelerator is consistent.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="replicationStatus" type="xs:boolean" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The replication status defines whether the table is enabled to
          replicate data if it is
          loaded and if replication is started. Whether data replication
          currently takes place for
          a table can be determined by the 'replicationDetails' attribute.
          Added with version 1.1 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="replicationDetails" type="dwa:replicationDetailsType"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The replication details provide information about the status of the
          actual replication
          processing that takes place for the table. In partiuclar, it
          indicates whether replication
          for the table is currently taking place or whether errors have
          occurred that caused replication
          of the table to stop
          Added with version 1.3 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="archiveStatus" type="xs:boolean" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The archive status defines whether the table has archived partitions
          that exist on the accelerator
          but do not exist or contain no data in DB2.
          Added with version 1.2 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="archiveProblemDetected" type="xs:boolean"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          This attribute indicates if archiving problems have been detected
          for the table, for example partitions that have been incompletely
          archived or restored. Detailed problem information about the problems
          can be obtained by calling the ACCEL_GET_TABLES_DETAILS procedure.
          Note that some problems can only be detected by calling the
          ACCEL_GET_TABLES_DETAILS procedure, so a value of false does not
          necessarily
          mean that no problems exist, but only that no problems have been
          detected.
          Added with version 1.3 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="archiveSynchronizationStatus"
        type="dwa:archiveSynchronizationStatusType" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          This attribute indicates if the table is archived on this or another
          accelerator.
          Added with version 1.6 of this schema.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="tableStatisticsType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Statistics information about table data on the accelerator.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
```

```
      <xs:element name="statisticsProfile" type="xs:string">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Content of SYSCAT.TABLES.STATISTICS_PROFILE on the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="usedDiskSpaceInMB" type="xs:unsignedLong"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Approximate disk space that is occupied by the table on the
          accelerator.
          The value is zero if the table is not loaded on the accelerator.
          The value does not include disk space for archived partitions.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="rowCount" type="xs:unsignedLong" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Deprecated. We plan to remove this attribute in future versions.
          Approximate count of table rows currently loaded on the accelerator
          based
          on statistics taken as a snapshot in time.
          Note, that statistics only recognizes transactions which have a
          transaction ID (xid)
          that is less than the stable (last committed) transaction ID.
          For example, some of the
          rows may already be logically deleted in a later transaction so the
          rowCount value may be larger than
          the number of rows accessible by a query or with a select count(*).
          The value does not include rows in archived partitions.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="skew" type="xs:decimal" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Measures whether the distribution of the table between accelerator
          nodes is balanced. A value of zero
          corresponds to a perfectly balanced distribution. A high skew value
          indicates that the distribution key""
          of the table may need to be adjusted.
          The value is zero if the table is not loaded on the accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="organizedPercent" type="xs:decimal" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Measures how well the table is organized according to the specified
          organization key. This value is only
          meaningful for large tables.
          The value is zero if the table is not loaded on the accelerator or if
          the table has no organizing key
          columns defined.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="lastLoadTimestamp" type="xs:dateTime" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Timestamp when the table was last loaded or updated by a stored
          procedure.
          The value is only provided when the table is loaded on the
          accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="capturePointTimestamp" type="xs:dateTime"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Timestamp from which logged changes in DB2 can be captured and
          replicated for a replicated table.
          The value is only provided when the table is loaded on the
          accelerator and enabled for replication.
          Added with version 1.1 of this schema.
        </xs:documentation>
      </xs:annotation>
```

```
  </xs:attribute>
  <xs:attribute name="archiveDiskSpaceInMB" type="xs:unsignedLong"
    use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     Approximate disk space that is occupied by archived table partitions
     on the accelerator.
     Added with version 1.2 of this schema.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="archiveRowCount" type="xs:unsignedLong" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     Count of table rows in archived partitions on the accelerator.
     Added with version 1.2 of this schema.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="tableBackendStatisticsCollectionTimestampCollected"
    type="xs:boolean" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     If timestamp from the last change made to recorded statistics for the
     load table version has been returned.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="archiveBackendStatisticsCollectionTimestampCollected"
    type="xs:boolean" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     If timestamp from the last change made to recorded statistics for the
     archive table version has been returned.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="tableBackendStatisticsCollectionTimestamp"
    type="xs:dateTime" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     The timestamp from the last change made to recorded statistics for
     the load table.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="archiveBackendStatisticsCollectionTimestamp"
    type="xs:dateTime" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     The timestamp from the last change made to recorded statistics for
     the archive table.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="lastAccessTimestamp" type="xs:dateTime" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     The timestamp from the last access to the table on the accelerator as
     part of
     queries or the execution of INSERT, UPDATE, or DELETE statements on
     an
     accelerator-only table.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
  <xs:attribute name="accessCount" type="xs:unsignedLong" use="optional">
   <xs:annotation>
    <xs:documentation xml:lang="en">
     The number of times this table was accessed on the accelerator by
     queries or
     by the execution of INSERT, UPDATE, or DELETE statements on an
     accelerator-only table.
    </xs:documentation>
   </xs:annotation>
  </xs:attribute>
 </xs:complexType>
 <xs:complexType name="remoteTableType">
  <xs:annotation>
   <xs:documentation xml:lang="en">
    Three part remote table name for federated tables on the accelerator.
    The attributes are populated when table information is retrieved from
```

```
        the referring subsystem.
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="location" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The location name of the owning subsystem for federated tables.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="schema" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The schema of the table on the owning subsystem for federated tables.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="name" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The table name on the owning subsystem for federated tables.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:simpleType name="tableLoadStatusType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Creating"/>
      <xs:enumeration value="InitialLoadPending"/>
      <xs:enumeration value="InitialLoadInProgress"/>
      <xs:enumeration value="Loaded"/>
      <xs:enumeration value="LoadInProgress"/>
      <xs:enumeration value="ReplicationInProgress"/>
      <xs:enumeration value="SchemaModificationInProgress"/>
      <xs:enumeration value="ArchivingInProgress"/>
      <xs:enumeration value="Error"/>
      <xs:enumeration value="Operational"/>
      <xs:enumeration value="Dropping"/>
      <xs:enumeration value="FederatedTableOperational"/>
      <xs:enumeration value="SelectPermissionRevoked"/>
      <xs:enumeration value="Unknown"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="tableIntegrityType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Unimpaired">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The table exists in DB2 and on the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ViolatedOnAccelerator">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The table exists in DB2 but not on the accelerator.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ViolatedInDatabaseManagementSystem">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The table does not exist in DB2.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ChangedSchema">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The table schema was changed in DB2.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="replicationDetailsType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:element name="tableSetForLoad">
    <xs:complexType>
      <xs:sequence>
        <xs:sequence>
```

```
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Specification for external tool exploiters that replace DB2
              UNLOAD.
            </xs:documentation>
          </xs:annotation>
          <xs:element name="externalTool"
              type="dwa:externalToolSpecificationType" minOccurs="0"
              maxOccurs="1"/>
        </xs:sequence>
        <xs:sequence>
          <xs:annotation>
            <xs:documentation xml:lang="en">
              list of all tables that should be loaded or reloaded.
            </xs:documentation>
          </xs:annotation>
          <xs:element name="table" type="dwa:tableLoadSpecificationType"
              minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:sequence>
      <xs:attribute name="usePhysicalPartitionNumbers" type="xs:boolean"
          default="false">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Interprets all specified partition numbers as physical partition
            numbers.
            If the attribute is set to false or not used,
            the specified partition numbers are interpreted as logical
            partitions.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
            <xs:enumeration value="1.2"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableLoadSpecificationType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence minOccurs="0" maxOccurs="1">
          <xs:element name="partitions" type="dwa:partitionNumberListType">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Optional list of partitions to be loaded. Default is to load
                all partitions
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="forceFullReload" type="xs:boolean" default="false">
          <xs:annotation>
            <xs:documentation xml:lang="en">
                            This attribute is deprecated. To completely reload
                            a table
                            do not specify a partition list and do not specify
                            the
                            detectChanges attribute.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="detectChanges">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Controls which changes in the table the stored procedure detects.
              If the value
              is not specified or PARTITIONS, only added partitions in
              partitioned tables are detected
              and loaded.
              Added with version 1.1 of this schema.
            </xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:enumeration value="PARTITIONS"/>
              <xs:enumeration value="DATA"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="tableSetForRemove">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Specification of all DB2 tables that should be removed from the
            accelerator
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableReferenceType" minOccurs="1"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="force" type="xs:boolean" use="optional">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Force removing tables even if this may result irregular states like
            leftover"
            restrictive table space states and unrecovered archive data."
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="tableSetForRestoreArchiving">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            list of all tables that should be restored.
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableRestoreSpecificationType"
            minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="usePhysicalPartitionNumbers" type="xs:boolean"
          default="false">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Interprets all specified partition numbers as physical partition
            numbers.
            If the attribute is set to false or not used,
            the specified partition numbers are interpreted as logical
            partitions.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableRestoreSpecificationType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence minOccurs="0" maxOccurs="1">
          <xs:element name="partitions" type="dwa:partitionNumberListType">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Optional list of partitions to be restore. Default is to
                restore all archived partitions
                Partitions that are not archived are ignored
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="tableSetForSetTablesReplication">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Specification of all DB2 tables whose replication status is to be
            changed
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableReferenceType" minOccurs="1"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="overrideExistingKeys" type="xs:boolean"
          default="false" use="optional">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Allows for replacing existing keys by new ones optimized for
            performance of setTablesReplication.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="tableSet">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            List of all DB2 tables that should be processed by the stored
            procedure
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableReferenceType" minOccurs="1"
            maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:element name="tableSpecifications">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Specification of all DB2 tables that should be added to the
            accelerator
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:tableSpecificationType"
            minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
            <xs:enumeration value="1.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableSpecificationType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence>
          <xs:element name="distributionKey" type="dwa:columnReferenceList"
              minOccurs="0" maxOccurs="1">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                The distribution key determines how rows of tables get assigned
                to processing nodes of the accelerator.
                Query performance for joins between two large tables can be
                improved by selecting the same distribution
                key for both tables. Inappropriate choice of the distribution
                key can lead to unbalanced amounts of
                table data on the compute nodes.
                If no distribution key is specified, a random distribution is
                chosen which is always balanced.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="organizingKey" type="dwa:columnReferenceType"
              minOccurs="0" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                An organizing key determines how rows of tables are clustered
                together on the accelerator.
                A table can have multiple, independent organizing keys. Each
                key consists of a single column.
                Query performance for frequent filtering conditions can be
                improved by selecting the filtering columns
                as organizing keys.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="informationalUniqueKey"
              type="dwa:columnReferenceList" minOccurs="0" maxOccurs="1">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                An informational unique key is a specification of a column
                combination that guarantees uniqueness,
                though this is not enforced by Db2 on z/OS (e.g. by a unique
                index).
                WARNING: If this uniqueness proposition is violated, incorrect
                output of queries is possible.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="useConstraintsForQueryOptimization"
              type="xs:boolean" minOccurs="0" maxOccurs="1">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Constraints in Db2 for z/OS are used by the Db2 optimizer in
                the appliance. Also information constraints specified by the
                user are used.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="tableSetForStatisticsCollection">
    <xs:complexType>
      <xs:sequence>
        <xs:sequence>
          <xs:annotation>
            <xs:documentation xml:lang="en">
              list of all tables for which statistics collection is requested
            </xs:documentation>
          </xs:annotation>
          <xs:element name="table"
              type="dwa:tableStatisticsCollectionSpecificationType"
              minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:sequence>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="tableStatisticsCollectionSpecificationType">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:sequence minOccurs="0" maxOccurs="1">
```

```
          <xs:element name="passThrough" type="xs:string">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Optional RUNSTATS clause that can be passed through to the
                accelerator database system.
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:element name="tableSetForSynchronizeSchema">
    <xs:complexType>
      <xs:sequence>
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Specification of all DB2 tables that should be synchronized with
            the accelerator
          </xs:documentation>
        </xs:annotation>
        <xs:element name="table" type="dwa:synchronizeSchemaTable"
            minOccurs="1" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="continueReplication" type="xs:boolean"
          default="false" use="optional">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Allows replication to continue for tables in table set.
          </xs:documentation>
        </xs:annotation>
      </xs:attribute>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="synchronizeSchemaTable">
    <xs:complexContent>
      <xs:extension base="dwa:tableReferenceType">
        <xs:attribute name="continueReplication" type="xs:boolean"
            default="false"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="traceProfileNameType">
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="traceProfileType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Defines an accelerator trace profile.
        Trace profile specifications are provided by IBM support
        when necessary.
      </xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="dwa:traceProfileNameType">
        <xs:sequence>
          <xs:element name="description" type="xs:string" minOccurs="1"
              maxOccurs="1"/>
          <xs:element name="component" minOccurs="0" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Defines trace levels for individual tracing components in the
                accelerator
                that deviate from
                the default trace level specified below.
              </xs:documentation>
            </xs:annotation>
            <xs:complexType>
              <xs:attribute name="name" type="xs:string" use="required"/>
              <xs:attribute name="level" type="dwa:traceLevelType"
                  use="required"/>
            </xs:complexType>
          </xs:element>
          <xs:element name="nzdbgParameters" type="dwa:nzdbgParametersString"
              minOccurs="0" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Defines command line parameters for the nzdbg tool that
                controls debug
                tracing of the
                Netezza host processes. For each element, a call to nzdbg will
                be made
                with the specified
                parameters. If no nzdbg element is given,
                Netezza debug tracing is turned
                off for all
                host processes
                No special characters except for '-', '*' and space may occur
                in
                the command
                line parameters
              </xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="backendDbsDiagnosticParameters" type="xs:string"
              minOccurs="0" maxOccurs="unbounded">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Defines parameters for diagnostic purposes in the backend DBS
              </xs:documentation>
            </xs:annotation>
          </xs:element>
        </xs:sequence>
        <xs:attribute name="defaultLevel" type="dwa:traceLevelType"
            use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Defines the default trace level for all accelerator components
              that are
              not explicitly
              listed as sub-elements.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="traceFileSizeInMB" type="xs:unsignedInt"
            use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Defines the aggregate size of the output trace files that are
              generated.
              When the
              specified
              size is exceeded, tracing wraps around and overwrites old trace
              files.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="forceRecordFlush" type="xs:boolean" use="optional"
            default="false">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Defines whether trace records will be immediately flushed to
              disk. If
              set to true, this
              impacts performance but ensures that no trace records are lost in
              the
              case of system crashed.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:simpleType name="nzdbgParametersString">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Restricts parameters for nzdbg to the characters that may actually
        appear in
        valid arguments
        and
        forbids shell special character (* is implicitly escaped by the
        executing
        code)
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:pattern value="[\-* 0-9a-zA-Z]*"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="traceLevelType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="OFF"/>
      <xs:enumeration value="FATAL"/>
      <xs:enumeration value="ERROR"/>
      <xs:enumeration value="WARN"/>
      <xs:enumeration value="INFO"/>
      <xs:enumeration value="DEBUG"/>
      <xs:enumeration value="DEBUG_EXTENDED"/>
      <xs:enumeration value="TRACE"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="replicationSeverityType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ALL"/>
      <xs:enumeration value="INFORMATION"/>
      <xs:enumeration value="WARNING"/>
      <xs:enumeration value="ERROR"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="stopReplicationType">
    <xs:attribute name="mode" use="optional" default="CONTROLLED">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Mode how replication is stoped.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:enumeration value="CONTROLLED">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Completes in-progress work and ends replication. This may take
                some time
                if there are long
                transactions that must be completely applied
                on the accelerator.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="IMMEDIATE">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Stops all in-progress work and ends replication. Starting
                replication
                after stopping
                with
                IMMEDIATE can take longer than if it would have been stopped
                using CONTROLLED.
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
          <xs:enumeration value="ABORT">
            <xs:annotation>
              <xs:documentation xml:lang="en">
                Stops all in-progress work and aborts replication. This option
                is the
                fastest way
                to end
                replication, but starting replication again can take much
                longer
              </xs:documentation>
            </xs:annotation>
          </xs:enumeration>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="getReplicationEventsType">
    <xs:attribute name="fromTimestamp" type="xs:dateTime" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Select only replication messages that were generated after the given
          timestamp
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="toTimestamp" type="xs:dateTime" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
```

```
          Select only replication messages that were generated before the given
          timestamp
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="minSeverity" type="dwa:replicationSeverityType"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The minimum severity the replication messages; only messages of
          specified or higher
          severity
          are returned.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="replicationStatusMessages">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Messages generated by the replication subsystem on the accelerator.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="event" type="dwa:replicationEventType" minOccurs="0"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="replicationEventType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Replication status message of the accelerator.
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="id" type="xs:unsignedLong" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Id of the event.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="message" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Message for the event.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="originator" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Name of the agent that originated the event.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="time" type="xs:dateTime" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Timestamp when the event occured.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="severity" type="dwa:replicationSeverityType"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Severity of the event.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="replicationAgentStatisticType">
    <xs:attribute name="insertCount" type="xs:long" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Number of inserts that have been captured or applied by the agent.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="updateCount" type="xs:long" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
```

```
          Number of updates that have been captured or applied by the agent.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="deleteCount" type="xs:long" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Number of deletes that have been captured or applied by the agent.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="replicationAgentStatisticTypeExtended">
    <xs:complexContent>
      <xs:extension base="dwa:replicationAgentStatisticType">
        <xs:attribute name="spilledRows" type="xs:long" use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Number of spilled rows.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="spilledRowsApplied" type="xs:long" use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Number of spilled rows applied by the agent.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="tableReferenceType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Identifies a DB2 table by schema (creator) and table name
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="schema" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="federatedTableInputType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Identifies a Federated table by schema (owner) and DB2 table name
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="schema" type="xs:string" use="required"/>
    <xs:attribute name="valid" type="xs:boolean" use="optional"/>
    <xs:attribute name="returnCode" type="xs:string" use="optional"/>
    <xs:attribute name="userSpecifiedAotName" type="xs:string" use="optional"/>
    <xs:attribute name="userSpecifiedAotSchema" type="xs:string"
        use="optional"/>
    <xs:attribute name="userSpecifiedDatabase" type="xs:string"
        use="optional"/>
    <xs:attribute name="referenceSynonym" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="federatedTableOutputType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Identifies a Federated table by schema (owner) and DB2 table name
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="schema" type="xs:string" use="required"/>
    <xs:attribute name="valid" type="xs:boolean" use="optional"/>
    <xs:attribute name="returnCode" type="xs:string" use="required"/>
    <xs:attribute name="userSpecifiedAotName" type="xs:string" use="optional"/>
    <xs:attribute name="userSpecifiedAotSchema" type="xs:string"
        use="optional"/>
    <xs:attribute name="columnList" type="xs:string" use="optional"/>
    <xs:attribute name="referenceSynonym" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="columnReferenceList">
    <xs:sequence>
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Specifies a list of columns in a DB2 table.
        </xs:documentation>
      </xs:annotation>
      <xs:element name="column" type="dwa:columnReferenceType" minOccurs="1"
```

```
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="columnReferenceType">
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="taskIdentifierType">
    <xs:attribute name="id" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="taskType">
    <xs:complexContent>
      <xs:extension base="dwa:taskIdentifierType">
        <xs:attribute name="type" type="xs:string" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              The type of the task, like Load or Query
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="ageInSeconds" type="xs:unsignedLong"
            use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              The time span for which the task has already been executing
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="progressMessage" type="xs:string" use="required">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Task-specific information about what the task is currently doing,
              if available
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="progress" type="xs:decimal" use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              The progress of the task in percent, if available
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
        <xs:attribute name="user" type="xs:string" use="optional">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              The DB2 user ID that originated the task. The attribute is not
              set
              for accelerator internal
              system tasks.
            </xs:documentation>
          </xs:annotation>
        </xs:attribute>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="sourceVersionType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Identifies a software product version which is currently active.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="description" type="dwa:descriptionType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The description of the currently active version.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="targetProductVersion" type="dwa:targetVersionType"
          minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Identifies the product versions that are available on the
            accelerator but
            not active.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="target" type="dwa:packageTargetType"
        default="ACCELERATOR">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The target that defines the type of this software product.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="fileName" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The file name defines the name of a file.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attributeGroup ref="dwa:productVersionAttributes"/>
  </xs:complexType>
  <xs:complexType name="targetVersionType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Identifies the version where we can go jump to.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="description" type="dwa:descriptionType">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The description of the target version.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="impact" type="dwa:impactType"/>
    </xs:sequence>
    <xs:attribute name="minimumRequiredVersion" type="xs:string"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          For backward compatibility.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attributeGroup ref="dwa:productVersionAttributes"/>
  </xs:complexType>
  <xs:complexType name="impactType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        The impact(s) associated with a switch to the target version.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="text" type="xs:string">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            A description of all impacts.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="reboot" type="xs:boolean" fixed="false">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Retained for backward compatibility.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="descriptionType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        The description of THIS version.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="short" type="xs:string">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The short (1 line) description of THIS version.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element name="long" type="xs:string">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The detailed description of THIS version.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="deployablePackageListType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Specifies a list of deployable packages that are available in z/OS HFS
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="package" type="dwa:deployablePackageType" minOccurs="0"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="deployablePackageType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Describes an update package that can be deployed on the accelerator or
        the
        Netezza host.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="information" type="dwa:informationType" minOccurs="0">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Package information from the manifest. Only available if the
            package was
            an accelerator
            package
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="location" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The absolute HFS file name for the deployable package that can be
          transferred to the
          accelerator.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="target" type="dwa:packageTargetType" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The target for deploying the package.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="informationType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Describes an update package for an accelerator software product.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="description" type="dwa:descriptionType"/>
    </xs:sequence>
    <xs:attributeGroup ref="dwa:productVersionAttributes"/>
  </xs:complexType>
  <xs:simpleType name="packageTargetType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="ACCELERATOR">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Accelerator server kit will be deployed on the accelerator and
            can then be activated
            with the
            activateDeployedPackage command for the stored procedure.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="HOST">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The file will be deployed on the Neteza host system and can then be
            processed manually.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NETEZZA_NPS">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Netezza Server (NPS) will be deployed on the Neteza host system
            and can then be
            processed manually.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NETEZZA_FDT">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Netezza Firmware (FDT) will be deployed on the Neteza host
            system
            and can then be
            processed manually.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="NETEZZA_HPF">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Netezza platform (HPF) will be deployed on the Neteza host
            system
            and can then be
            processed manually.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ACCESS_SERVER">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Incremental Update Access Server kit will be deployed on the
            accelerator and can then be
            activated with the
            activateDeployedPackage command for the stored procedure.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="REPLICATION_ENGINE">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The Incremental Update Replication Engine kit will be deployed on
            the
            accelerator and can
            then be activated with the
            activateDeployedPackage command for the stored procedure.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RPM">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            RPM packages will be deployed on the accelerator and can
            then be activated with the
            activateDeployedPackage command for the stored procedure.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:attributeGroup name="productVersionAttributes">
    <xs:attribute name="version" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Version number that uniquely idenfifies a deployed or deployable
          version
          of the software
          product.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="buildLabel" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Optional build lable for the product version.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="simpleVersion" type="xs:string" use="optional">
```

```
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The simple version referencing this version.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:complexType name="partitionInformationType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Describes partitioning information about a table.
        If this information is not present, the
        table is not partitioned.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Specifies a list of partitioning columns.
        </xs:documentation>
      </xs:annotation>
      <xs:element name="column" type="dwa:columnReferenceType" minOccurs="0"
          maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="type" type="dwa:partitioningType" use="required"/>
  </xs:complexType>
  <xs:simpleType name="partitioningType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="BY_RANGE"/>
      <xs:enumeration value="BY_GROWTH"/>
      <xs:enumeration value="NOT_PARTITIONED"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="partitionNumberListType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        A comma-separated list of logical partition numbers. Partition number
        ranges can be
        specified
        by two numbers separated by a colon.
        Negative partition numbers count from the end, -1 specifies the last
        partition.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
  <xs:complexType name="backupImageList">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Describes backup images for an archived table.
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="backupImage" type="xs:string" minOccurs="0"
          maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="waitForReplicationType">
    <xs:annotation>
      <xs:documentation xml:lang="en">
        Parameters defining how long the accelerator waits for the
        synchronization.
        of replicated
        tables.
      </xs:documentation>
    </xs:annotation>
    <xs:attribute name="timeoutInSeconds" type="xs:unsignedLong" default="0">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Maximum wait time. If the timeout is exceeded, the command completes
          with a warning even if
          not all changes have been replicated.
          If the value is zero or not specified, the accelerator may wait
          indefinitely.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:simpleType name="archiveErrorStatusType">
    <xs:restriction base="xs:string">
      <xs:annotation>
        <xs:documentation xml:lang="en">
```

```
          The value 'Unimpaired' indicates that no problem was detected;
          other values indicate
          potential problems. The specific error conditions
          and corresponding values are subject to change.
        </xs:documentation>
      </xs:annotation>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="archiveSynchronizationStatusType">
    <xs:restriction base="xs:string">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Is at least on partition of the table archived on this or another
          accelerator
        </xs:documentation>
      </xs:annotation>
      <xs:enumeration value="NotArchived">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            No partition of this table is archived on any accelerator.
            In SYSACCEL.SYSACCELERATEDTABLES(ARCHIVE) = ' '.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="ArchivedOnAnotherAccelerator">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            At least one partition is archived on another accelerator
            In SYSACCEL.SYSACCELERATEDTABLES(ARCHIVE) = 'C'.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="PartiallyArchived">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            On this accelerator some partitions are archived, but it exists
            another accelerator
            with more archived partitions or re-archived partitions.
            In SYSACCEL.SYSACCELERATEDTABLES(ARCHIVE) = 'B'.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="Archived">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            This accelerator is up-to-date and is a reference accelerator.
            In SYSACCEL.SYSACCELERATEDTABLES(ARCHIVE) = 'A'.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="RestoredOnAnotherAccelerator">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Partitions were restored on another accelerator. This accelerator
            can not answer
            queries until it is being brought up in sync again.
            In SYSACCEL.SYSACCELERATEDTABLES(ARCHIVE) = 'X'.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
      <xs:enumeration value="UnkownDueToBackLevelSPs">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            The stored procedures are backlevel, therefore the value can not be
            determined.
          </xs:documentation>
        </xs:annotation>
      </xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:attributeGroup name="sharedChangeInformationAttibutes">
    <xs:attribute name="dataSizeInMB" type="xs:unsignedLong">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Approximate size of data that would be loaded
          when loading the table or partition
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="sharedTablespace" type="xs:boolean">
      <xs:annotation>
        <xs:documentation xml:lang="en">
```

```
          True if the table resides in a shared table space. In this case,
          change
          information may be
          incorrect because the changes may refer to other
          tables
          in the tablespace
          This attribute can only be set for table change information,
          never for
          partition change information
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:attributeGroup name="sharedPartitionAttibutes">
    <xs:attribute name="dbmsPartNr" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Physical partition number in DB2.
          The attribute is not present for archived partitions that
          have been
          rotated
          in DB2 after they were archived.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="logicalPartNr" type="xs:unsignedInt" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Logical partition number in DB2 for a range-partitioned table.
          For a table that is
          partitioned by growth, this attribute is identical
          to the physical partition number.
          The attribute is not present for archived partitions
          that have been
          rotated
          in DB2 after they were archived.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="endingAt" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Range partition specification for a range-partitioned table.
          For archived partitions, this
          represents the range partition
          specification
          at the time when the partition was archived.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="archiveErrorStatus" type="dwa:archiveErrorStatusType"
        use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Status string of archiving for specified partitions.
          For archived partitions, this represents
          the status of the
          archiving. If not specified, the partitions is not archived.
          Added with version 1.1 of tableSetDetails.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:attributeGroup>
  <xs:complexType name="externalToolCallbackType">
    <xs:attribute name="callerid" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Identifies the load request to the caller of
            ACCEL_LOAD_TABLES. The stored procedure is obliged to
            use the specified callerid value when invoking the load callback.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="library" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            The WLM startup JCL for the IDAA procedures has to concatenate a
            third-party LIBRARY dataset to STEPLIB containing the needed
            callback functions
            in a dataset member.
            This attribute specifies the name of this dataset member.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="versionFunction" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the version function of the
            callback interface.
            The interface of the version callback never changes: it returns an
            unsigned int
            and allows the ACCEL_LOAD_TABLES procedure to use the correct
            callback interface version.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="initFunction" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the init function of the load
            callback interface
            equivalent to a DSNUTILU(UNLOAD) call to unloading a Db2 tablespace
            partition to a pipe.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="loadFunction" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the load function of the load
            callback interface
            equivalent to a DSNUTILU(UNLOAD) call to unloading a Db2 tablespace
            partition to a pipe.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="terminateFunction" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the terminate function of the
            load callback interface
            equivalent to a -TER UTIL() call to terminate a running utility
            unloading a Db2 tablespace partition to a pipe.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="externalToolRerouteDataCallbackType">
    <xs:attribute name="callerid" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Identifies the load request to the caller of
            ACCEL_LOAD_TABLES. The stored procedure is obliged to
            use the specified callerid value when invoking the load reroute
            data callback.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="library" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            The WLM startup JCL for the IDAA procedures has to concatenate a
            third-party LIBRARY dataset to STEPLIB containing the needed
            callback functions
            in a dataset member.
            This attribute specifies the name of this dataset member.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="versionFunctionRDC" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the version function of the
            reroute data callback interface.
            The interface of the version callback never changes: it returns an
            unsigned int
            and allows the ACCEL_LOAD_TABLES procedure to use the correct
            callback interface version.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="initFunctionRDC" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the init function of the load
            reroute data callback interface
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="preUnloadFunctionRDC" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the function of the load
            reroute data callback interface
            exchanging the pipe names for re-routing the DSNUTILU(UNLOAD) data
            for a Db2 tablespace partition.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="postUnloadFunctionRDC" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            Name of the function that implements the function of the reroute
            data load callback interface
            to communicate the success or failure of a DSNUTILU(UNLOAD) call
            routing data to the caller of ACCEL_LOAD_TABLES.
            Details on this interface can be found in the IDAA documentation.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="externalToolSpecificationType">
    <xs:sequence>
      <xs:choice>
        <xs:element name="callback" type="dwa:externalToolCallbackType"
            minOccurs="0" maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Specification of details for external load utility callback
              interface functions.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
        <xs:element name="rerouteDataCallback"
            type="dwa:externalToolRerouteDataCallbackType" minOccurs="0"
            maxOccurs="1">
          <xs:annotation>
            <xs:documentation xml:lang="en">
              Specification of details for load reroute data callback interface
              functions.
            </xs:documentation>
          </xs:annotation>
        </xs:element>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          The name of an external load utility that provides the data source.
          If tools vendors
          intercept DB2 unload utilities to provide data from
          other sources
          they must provide their tool name/identifier
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="enforceTableLoadSequence" type="xs:boolean"
        use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Enforces the load the tables in the sequence given of the specified
          table
          elements.
          If set to
          FALSE the accelerator may reorder the load of individual
          tables
          to shorten the load time. By setting this attribute to TRUE, the
          accelerator will
          not do this and load the tables in the pre-defined sequence.
          This may decrease the performance of the load and also
          prevent to
          accelerator from handling
          more load operations in parallel.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="loadAppend" type="xs:boolean" use="optional"
        default="false">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Flag indicating if data to be loaded shall replace existing data or
          only be appended.
          If set
          to FALSE, existing data is removed and then the new data is
          loaded. If set to TRUE,
          all existing data is retained and the new data is
          added.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
    <xs:attribute name="utilityNamePrefix" type="xs:string" use="optional">
      <xs:annotation>
        <xs:documentation xml:lang="en">
            The prefix for the utility name. Must not be longer than 3
            characters.
            If not set the default AQT will be used.
        </xs:documentation>
      </xs:annotation>
    </xs:attribute>
  </xs:complexType>
  <xs:simpleType name="tableTypeType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="Regular"/>
      <xs:enumeration value="AcceleratorOnly"/>
      <xs:enumeration value="FederatedReference"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="versionInformationType">
    <xs:sequence>
      <xs:element name="component" type="dwa:componentVersionType"
          minOccurs="1" maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            Version information for a single component identified by the name.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="componentVersionType">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="version" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="procedurePackagesType">
    <xs:sequence>
      <xs:element name="package" type="dwa:procedurePackageType" minOccurs="1"
          maxOccurs="unbounded">
        <xs:annotation>
          <xs:documentation xml:lang="en">
            A singel procedure package.
          </xs:documentation>
        </xs:annotation>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="procedurePackageType">
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="enabled" type="xs:boolean" use="required"/>
    <xs:attribute name="version" type="xs:string" use="optional"/>
  </xs:complexType>
  <xs:complexType name="acceleratorSettingType">
    <xs:attribute name="name" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
              Name identifying the parameter.
                name="FACTREL_SIZE_THRESHOLD"
                value=Defines the minimum number of rows for a table considered
                to be a
                  fact table or large table that should not be re-distributed
                  for a join.
                name="UNICODE_SORT_ORDER"
```

```
                value=Sort order for character columns with Unicode encoding.
                Possible values are:
                  BinaryUTF8
                    default setting - recommended for installations that store
                    Unicode strings in MBCS CHAR columns
                  BinaryUTF16
                    deprecated (not supported - mapped to BinaryUTF8)
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="value" use="required">
      <xs:annotation>
        <xs:documentation xml:lang="en">
          Value applied for the parameter.
        </xs:documentation>
      </xs:annotation>
      <xs:simpleType>
        <xs:restriction base="xs:string"/>
      </xs:simpleType>
    </xs:attribute>
  </xs:complexType>
  <xs:complexType name="procedurePackageNameType">
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="configOptimizationProfileCommandType">
    <xs:attribute name="command"
        type="dwa:configOptimizationProfileCommandCommandType" use="required"/>
    <xs:attribute name="optimizationProfileSchema" type="xs:string"
        use="optional"/>
    <xs:attribute name="optimizationProfileName" type="xs:string"
        use="optional"/>
  </xs:complexType>
  <xs:simpleType name="configOptimizationProfileCommandCommandType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="CREATE"/>
      <xs:enumeration value="UPDATE"/>
      <xs:enumeration value="DELETE"/>
      <xs:enumeration value="LIST"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

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 grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
USA

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

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 jurisdictions 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 websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide 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:

IBM Director of Licensing
IBM Corporation
North Castle Drive, MD-NC119
Armonk, NY 10504-1785
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 document 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.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

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.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains 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.

Each copy or any portion of these sample programs or any derivative work must include a copyright notice as follows:

© (your company name) (year).
Portions of this code are derived from IBM Corp. Sample Programs.
© Copyright IBM Corp. _enter the year or years_.

Programming interface information

This *IBM Db2 Analytics Accelerator for z/OS: Stored Procedures Reference* primarily documents intended Programming Interfaces that allow the customer to write programs to obtain services of IBM Db2 Analytics Accelerator for z/OS.

This *IBM Db2 Analytics Accelerator for z/OS: Stored Procedures Reference* also documents information that is NOT intended to be used as Programming Interfaces of IBM Db2 Analytics Accelerator for z/OS. This information is identified where it occurs, either by an introductory statement to a chapter or section or by the following marking:

- *NOT Programming Interface information*
- *End of NOT Programming Interface information*

Trademarks

IBM, the IBM logo, and 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" at www.ibm.com/legal/copytrade.shtml.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

GDPR considerations

For PID(s): 5697-DA7, 5697-DA5, 5697-DAB

Notice: This document is intended to help you in your preparations for GDPR readiness. It provides information about features of IBM Db2 Analytics Accelerator for z/OS that you can configure, and aspects of the product's use, that you should consider to help your organization with GDPR readiness. This information is not an exhaustive list, due to the many ways that clients can choose and configure features, and the large variety of ways that the product can be used in itself and with third-party applications and systems.

Clients are responsible for ensuring their own compliance with various laws and regulations, including the European Union General Data Protection Regulation. Clients are solely responsible for obtaining advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulations that may affect the clients' business and any actions the clients may need to take to comply with such laws and regulations.

The products, services, and other capabilities described herein are not suitable for all client situations and may have restricted availability. IBM does not provide legal, accounting, or auditing advice or represent or warrant that its services or products will ensure that clients are in compliance with any law or regulation.

GDPR

General Data Protection Regulation (GDPR) has been adopted by the European Union (“EU”) and applies from May 25, 2018.

Why is GDPR important?

GDPR establishes a stronger data protection regulatory framework for processing of personal data of individuals. GDPR brings:

- New and enhanced rights for individuals
- Widened definition of personal data
- New obligations for processors
- Potential for significant financial penalties for non-compliance
- Compulsory data breach notification

Read more about GDPR

- EU GDPR Information Portal (https://www.eugdpr.org/)
- ibm.com®/GDPR website (https://ibm.com/GDPR)

The EU General Data Protection Regulation (GDPR) regulates data privacy in the European Union (EU). For details, see: https://www.eugdpr.org

The regulations laid out in the GDPR apply if the data controller (an organization that collects data from EU residents), or the processor (an organization that processes data on behalf of a data controller, such as a cloud service provider), or the data subject (an individual) is based in the EU (source: Wikipedia. See: https://en.wikipedia.org/wiki/General_Data_Protection_Regulation). A violation of GDPR regulations might incur severe financial penalties (fines) and reputation damages. This documentation is intended for data controllers or processors who use Db2 for z/OS together with IBM Db2 Analytics Accelerator for z/OS with the aim of storing personal data of EU residents (individuals). As regards IBM Db2 Analytics Accelerator for z/OS, the IBM company is neither the data controller, nor the data processor. Nevertheless, the IBM company feels obliged to give such entities guidance on how to follow GDPR regulations.

Your responsibilities as a data controller or processor

If you act on behalf of a data controller or data processor, see the frequently asked questions (FAQ) on the official GDPR website to find out what your responsibilities are:

https://www.eugdpr.org/gdpr-faqs.html

Furthermore, refer to chapter 4 in the GDPR regulations:

https://www.eugdpr.org/article-summaries.html

Product Configuration for GDPR

The following sections provide considerations for configuring IBM Db2 Analytics Accelerator for z/OS to help your organization with GDPR readiness.

How to configure the offering so that it can be used in a GDPR environment

The following sections describe how personal data is stored and processed by IBM Db2 Analytics Accelerator for z/OS (called 'accelerator' in the text that follows).

If personal data is to be stored on an accelerator, the following configuration steps might help you meet your compliance objectives with regard to GDPR:

- Activate the encryption of data in motion (data that is transferred over a data network between Db2 for z/OS and the accelerator).
- Activate the encryption of data at rest (encryption of disks and other storage devices).
- Do not share passwords for the operation of the IBM Db2 Analytics Accelerator Console between multiple persons. Instead, create separate user IDs with different passwords for each console user.
- Give read access to personal data (the privilege to run queries) only to authorized Db2 for z/OS users. To this end, define appropriate Db2 authorizations (SELECT, and so on) in the Db2 for z/OS catalog.
- Restrict the right to unload data from and to create image copies in Db2 for z/OS. Give this right to authorized administrators only.
- Restrict access to the 'Save Trace' function in Db2 Analytics Accelerator for z/OS because the collected trace data might contain personal data. Access should be given to authorized administrators only.
- Restrict access to the table SYSIBM.USERNAMES in the catalog of the Db2 for z/OS communication database because this table contains authentication credentials. If these credentials are read or copied by unauthorized users, these users might obtain access to sensitive personal data.

Data Life Cycle

You can create copies of any Db2 for z/OS table on an accelerator (accelerator-shadow tables). As soon as you create a copy of a Db2 for z/OS table and load it with personal data, the GDPR regulations apply in the same way as they apply to personal data in the original Db2 for z/OS tables.

An accelerator also allows you to load and store external data outside of Db2 for z/OS, such as the data of VSAM files, in accelerator-shadow tables. Products like the Db2 Analytics Accelerator Loader can be used to this end. If the external data contains personal information, the GDPR regulations apply to the data that is trasferred to the accelerator.

What types of data?

The administrative environment of an accelerator (client and configuration software) does not store personal data, except for the user IDs and authentication tokens of its administrators. However, the GDPR regulations apply to all personal data that you store in database tables on the accelerator.

Where in the process?

An accelerator is loaded with data in the course of individual load operations or by the incremental update function (automated load of table updates). Adding a table to an accelerator does not yet copy any data. It merely defines the table's structure (metadata) on the accelerator. However, if you load the table or enable it for incremental updates, the data of the source table - which might contain personal data - is copied to the accelerator.

For what purpose?

An accelerator, like Db2 for z/OS, is a general database management system for SQL queries. Such queries are run with the aim of retrieving stored information and gaining further insights (analytical queries that uncover the dependencies or connections between sets of values). An accelerator can return the results of SQL queries to authorized and authenticated users of the system. The results usually consist of subsets or aggregations of numerous data records. All these records might include personal data.

Personal data used for online contact with IBM

IBM Db2 Analytics Accelerator for z/OS clients can submit online comments/feedback/requests to contact IBM about accelerator subjects in a variety of ways, primarily:

- Public comments area on pages in the IBM Integration community on IBM developerWorks®
- Public comments area on pages of the IBM Db2 Analytics Accelerator for z/OS product documentation in the IBM Documentation
- Feedback forms in the IBM Integration community

Typically, only the client name and email address are used, to enable personal replies for the subject of the contact, and the use of personal data conforms to the *IBM Online Privacy Statement* (https://www.ibm.com/privacy/cc/).

Data Collection

IBM Db2 Analytics Accelerator for z/OS can be used to collect personal data. When assessing your use of IBM Db2 Analytics Accelerator for z/OS and the requirements of GDPR, you should consider the types of personal data which in your circumstances are stored on an accelerator.

The use of an accelerator does not impose new requirements for the collection of personal data. It does not introduce new users or new privileges, but makes use of existing users and their privileges as defined in Db2 for z/OS. The use of an accelerator does not introduce additional obligations for the safekeeping or adminstration of data if you already observe the GDPR regulations for the storage of data in Db2 for z/OS.

Make sure that the process of collecting data and storing it in Db2 for z/OS is GDPR-compliant. The use of an accelerator will then be equally compliant.

Data Storage

The use of IBM Db2 Analytics Accelerator for z/OS involves the storage of data in a persistent manner.

An accelerator is a general-purpose data processing and retrieval engine. It is not aware of the meaning of data or any meaning that can be inferred from it. It is thus the responsibility of the data controller or processor to classify the data as personal data - for example account data - and make sure that the collection, access and processing of this data is in accordance with the GDPR regulations.

Storage in backups:

An accelerator has no backup function. However, image copies are created in Db2 for z/OS when the accelerator's archiving function is used (high performance storage saver). If these image copies contain personal data, the GDPR rules apply. This means, for example, that if individuals request their data to be deleted, it must also be deleted from all the image copies that may exist.

Storage in archives:

The high performance storage saver deletes table data from Db2 for z/OS and stores it exclusively on an accelerator. If this data contains personal information, the GDPR regulations might require that you delete the personal information from a number of records. The current version of the accelerator software does not allow DELETE operations on archived partitions. So in case you must delete records, first restore the archived partition to Db2 for z/OS, then delete then delete the personal data from Db2 for z/OS. Finally, archive the partition again (which now does not include the personal data anymore).

Data Access

Access to data on an accelerator requires a Db2 for z/OS user ID and a password. The accelerator recognizes the privileges defined in Db2 for z/OS and does not allow a user to run functions that she or he is not entitled to use.

As long as the Db2 for z/OS authorizations, such as SELECT, INSERT, UPDATE, or DELETE privileges, are defined in accordance with the GDPR regulations, access to data on the accelerator will likewise be in accordance with these regulations.

So it is crucial that the definition of roles and access rights in Db2 for z/OS effectuates an adequate separation of duties and responsibilities.

One accelerator-specific function might provide access to personal data on the accelerator: This is the "Save Trace" function, which collects diagnostic information for troubleshooting. The resulting trace files might contain personal data because they sometimes include the code of SQL queries to which personal information has been passed in the form of query literals. Trace files can also include memory dumps that contain personal data.

It is therefore important that access privileges to data, such as SELECT privileges or superuser privileges like SYSADM and DBADM, are granted with great care. The permission to use the "Save Trace" function should be given to selected administrators only.

The privilege to create image copies in Db2 for z/OS must also be handled with care. It is required, for example, if a user wants to archive table partitions on an accelerator.

Data Processing

Privileges defined in the Db2 for z/OS catalog control data processing on an accelerator.

In addition to that, the encryption features (encryption of data in motion and encryption of data at rest) should be activated. This prevents an unauthorized person without proper privileges from gaining direct access to personal data by interferring with the network communication or by obtaining physical access to the storage devices (hard disks or solid state disks).

Encryption features and safeguarding precautions:

Encryption of data in motion:

This feature encrypts the network traffic between an IBM Z® system and an accelerator.

Encryption of data at rest:

This feature encrypts the data on an accelerator's physical storage devices (hard disks and solid state disks). Depending on the model of the hardware, the feature might already be turned on by default.

Encryption key ownership:

Restrict access to the encryption keys because the keys enable a person to read or decrypt personal data on a network or storage device. Put the safekeeping and maintenance (rotation) of the encryption keys in the hands of a trusted administrator.

Data Deletion

When data is deleted from Db2 for z/OS, it might still exist in table copies on the accelerator. To delete the data on an accelerator, you have the following options:

The data in accelerator-shadow tables is deleted:

- If the data in the original Db2 for z/OS tables has been deleted, and if the affected accelerator-shadow tables have been reloaded (with the contents of the now empty Db2 tables).
- If the accelerator-shadow tables are removed from the accelerator.
- If the incremental update function is used and the function replicates empty table content to the accelerator because the original Db2 for z/OS table data has been deleted.

The data in an accelerator-only table (AOT) is deleted:

- If the AOT is removed from the accelerator.
- If an explicit DELETE statement is submitted on the Db2 for z/OS side, which references the AOT and contains a predicate that specifies the row to be deleted.

The data in partitions archived by the High Performance Storage Saver is deleted:

- If the table partitions are removed from the accelerator.
- If an archived partition is restored to Db2 for z/OS.
- If the data is deleted from Db2 for z/OS because this means that the archived partition is reloaded or that the partition is archived again after the deletion from Db2 for z/OS (empty table content will be reloaded or archived).

Data Monitoring

Db2 for z/OS provides powerful auditing and monitoring functions that allow you to track the processing of personal data. Because IBM Db2 Analytics Accelerator for z/OS is a logical component of Db2 for z/OS, and keeps only copies of Db2 for z/OS data, the monitoring functions for data processing in Db2 for z/OS are sufficient.

Responding to Data Subject Rights

To identify an individual's data that is stored on an accelerator, you can submit SQL queries from Db2 for z/OS. These queries ought to search for the name, the account number, or for other identifying information about the subject using SQL syntax. Since an accelerator works on copies of Db2 for z/OS data, the same queries can be run to retrieve the data either from Db2 for z/OS or from the accelerator. Specific advice or examples of such queries can not be provided here because the structure and the content of the database tables, as defined by the data controller or data processor, is not known.

Does the offering facilitate being able to meet data subject rights?

Right to Access

Can the client provide individuals access to their data?

Db2 for z/OS and accelerators can process SQL queries in which an individual's ID can be specified as a search criterion. This way, all data stored about that individual can be retrieved. In Db2 for z/OS, a client's data access can be restricted, so that the client can only search and retrieve a subset of the available data records, that is, those records that belong to a specific group of individuals. To this end, a feature called label based access control (LBAC) can be employed. IBM Db2 Analytics

Accelerator provides a similar functionality, which restricts access to a selected set of rows based on the CURRENT SQLID special register.

Can the client provide individuals information about what data the client has about the individual?

Yes. To identify an individual's data stored on an accelerator, the client can submit SQL queries from Db2 for z/OS. These queries ought to search for the name, the account number, or for other identifying information about the subject using SQL syntax.

Right to Modify

Can the client allow an individual to modify or correct their data?

Db2 for z/OS allows a client to update the data that is stored about an individual. Depending on the architecture of the application, it might even be possible to authorize the individuals, so that these can update their data by themselves. After an update in Db2 for z/OS, the modified data is copied to the accelerators attached to the database during the next reload operation, which can be started manually or automatically (by the incremental update function). During that process, incorrect data will be overwritten.

Can the client correct an individual's data for them?

Yes (see the previous question *Can the client allow an individual to modify or correct their data?*).

Right to Restrict Processing

Can the client stop processing an individual's data?

Yes. In Db2 for z/OS, a client can submit DELETE statements to remove data records of individuals. The removal will be reflected in the data that is stored on an attached accelerator when the next reload operation (manual or automatic) is taking place. During that process, records previously deleted from Db2 for z/OS will also be deleted from the accelerator. Furthermore, an individual's data can be excluded from SQL queries through the use of filters in query predicates. For example, a query that reads `SELECT CUSTOMER_DATA FROM CUSTOMER_TABLE where custid NOT IN (a,b,...c)` returns only data records that are not related to (do not belong to) the users in the specified group `(a,b,...c)`.

Right to Object

Same as "Right to Restrict Processing" on page 310.

Right to Be Forgotten

Can the client delete an individual's data?

Yes.

Right to Data Portability

Can the customer provide an individual with the information that they have about the individual in a user-friendly/machine readable format?

Yes.

Glossary

NOT Programming Interface information

This glossary includes terms and definitions for IBM Db2 Analytics Accelerator stored procedures.

The following cross-references are used in this glossary:

- *See* refers you from a term to a preferred synonym, or from an acronym or abbreviation to the defined full form.
- *See also* refers you to a related or contrasting term.

A

access plan graph
A visual representation of a query that shows the database objects that are accessed by the query and the order in which this is done.

C

coordinator node
The primary node of the accelerator. It distributes the workload among the other nodes.

D

data slice
A portion of an accelerator-shadow table that is processed by a worker node.

DDF
See distributed data facility (DDF).

distributed data facility (DDF)
A set of Db2 for z/OS components by which Db2 for z/OS communicates with another relational database management system.

Dynamic Statement Cache
A pool in which Db2 saves prepared SQL statements that can be shared among different threads, plans, and packages. By sharing these statements, applications can avoid unnecessary preparation processes and thus improve performance.

G

global temporary table
See temporary table.

H

handshake
The exchange of predetermined signals when a connection is established between two data set devices.

I

IFI
See instrumentation facility interface.

IFI call
An invocation of the instrumentation facility interface (IFI) by means of one of its defined functions.

inhouse processing
A query is processed by Db2 for z/OS and not routed to an accelerator.

instrumentation facility interface (IFI)
A programming interface that enables programs to obtain online trace data about Db2, to submit Db2 commands, and to pass data to Db2.

L

lock
A means of preventing uncommitted changes made by one application process from being perceived by another application process and for preventing one application process from updating data that is being accessed by another process. A lock ensures the integrity of data by preventing concurrent users from accessing inconsistent data.

P

plan ID
The identifier of an execution plan (part of the file name).

S

skew
An imbalanced distribution of table rows among the processing nodes of the accelerator.

T

table space
A logical unit of storage in a database. In Db2 for z/OS, a table space is a page set and can contain one or more tables.

temporary table
A table that holds temporary data. Temporary tables are useful for holding or sorting intermediate results from queries that contain many rows. The two kinds of temporary tables, which are created by different SQL statements, are the created temporary table and the declared temporary table.

throughput
A measure of the amount of information transmitted over a network in a given period of time. Throughput is generally measured in bits per second (bps), kilobits per second (Kbps), or megabits per second (Mbps).

End of NOT Programming Interface information

Index

Product Number: 5697-DA7

SH12-7103-13