Nirva JDBC Service

Document Version: 1.03

nirva

Table of Contents

Overview

The JDBC service is a NIRVA external service that provides access to any kind of data source providing a JDBC driver.

With this service, a user can search, add, modify and remove records from a JDBC data sources. He can also send any kind of SQL statement or call stored procedures.

Much of the performance issues or capacities are driver related. Therefore the user should choose the drivers with care. Nirva just provides an interface to the driver and cannot provide support on driver issues.

Login and logout

Users can login and logout to JDBC data sources.

Working on data sources

Users can add, modify, remove and search into JDBC data sources.

Security

The service uses the security functions of the JDBC data source allowing protection for users, databases, records and functionality.

Performance

The performance of the database accesses depends of the JDBC driver used. This performance is generally nearly the same as a native access to the database.

JDBC and DATABASE Service compatibility

The Nirva JDBC service has been written in a way to keep it as compatible as possible with the Nirva DATABASE services which relies on ODBC. Unless otherwise specified, the JDBC service commands which also exist in the DATABASE service share the same parameters with the same meaning. The default values and output objects are also the same and share the same structure.

Therefore using the JDBC service inplace of the DATABASE service, whenever only common commands are used, only requires chaning the NV_SERVICE part of the Nirva commands fomr "DATABASE" to "JDBC". Below are given the known compatibility limitations :

- The JDBC does not implement all the commands available in the DATABASE service
- The JDBC service only allows one simultaneously connection per session
- The JDBC:SOURCE:OPEN command differs from the DATABASE:SOURCE:OPEN (this is normal, since the mecanisms to open a connection differ in JDBC and ODBC)

Installation

The JDBC service is delivered as a NIRVA package and can be installed like any NIRVA service directly from the NIRVA configuration web site. Please see the NIRVA configuration chapter in the NIRVA user's guide for further information.

The JDBC service requires DBMS specific drivers. The .jar files containing the drivers should be put in the service's Bin directory (on a standard Nirva installation under windows this directory is c:\nirva\Services\JDBC\Bin).

Configuration

The jar files containing the JDBC drivers used must be put in the JDBC service's Bin directory (i.e. c:\nirva\Services\JDBC\Bin in a standard Windows Nirva installation). Otherwise, no further configuration is required.

Reference

This chapter gives the complete reference of all the JDBC service commands.

Classes

Here are the available JDBC service classes:

Class	Description
JDBC	Main class
SOURCE	Data source information and queries
QUERY	Result set management.

Error codes

JDBC Class

Value	Description
104	Invalid command
106	SQL Error
110	Internal error / Memory error (not enough)
111	Invalid parameter

SOURCE Class

Value	Description
112	Cannot open JDBC data source
113	Unknown JDBC driver

Value	Description
114	JDBC data source not open
119	No current source selected
120	Invalid query
129	Cannot add records
201	Reach maximum number of session allowed by the licence

QUERY Class

Value	Description
121	No record
122	No current query selected

Permissions

Value	Description
SOURCE_QUERY	Allow to open and close a connection and query the database directly from the borwser. Further limitations (eg. the types of authorized queries) can be obtained restraining the rights of the database user used for the connection.

Commands

For each command, the reference gives the command name, the sources for which the command may be used, the command description, the eventual command permissions, the parameter list and the eventual list of objects created by the command.

The parameters described in this chapter are command specific parameters. For general parameters, please refer to the Nirva command syntax chapter.

The available sources are:

- Client for all Nirva client interfaces including Nirva client library (nvc).
- Web for commands from a web browser.

- Procedure for commands from a Nirva procedure.
- Service for commands from service to service

JDBC class

This is the standard service class that provides service scope commands.

NOP

JDBC:JDBC:NOP

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command does nothing but allows to test that the database service is on line and answers correctly.

If the service is not on line, this command returns an error.

Parameters

None

Permissions

None

SOURCE class

All commands of the source class require the SOURCE_QUERY permission when used from the browser.

OPEN

JDBC:SOURCE:OPEN

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command opens a JDBC connection to a data source. This connection becomes the current one. Only one connection can be opened simultaneously for a single NIRVA session. If a connection is already opened, it will be closed before opening the new one.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

DRIVER — Name of the Java class to be used as the JDBC driver for this source. This parameter is mandatory.

URL — URL of the JDBC data source to open. This name must be a valid JDBC data source defined on the server. The URL parameter is mandatory.

USER — Optional user name for connecting the source. This is the specific data source user name that can be different than the user named of the NIRVA session.

PASSWORD — Eventual password for user identification.

CLOSE

JDBC:SOURCE:CLOSE

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command closes a previously opened JDBC data source. All queries eventually associated to the opened data source are automatically closed by this command.

Permissions

SOURCE_QUERY (when called from the browser)

INSERT

JDBC:SOURCE:INSERT

Source	Use Input Container	Use Output Container
Client Web Procedure Service	Yes	No

Description

This command adds new records to the given table of the current opened source.

The table column names of the NIRVA object must correspond to the table column names of the data source.

To load the JDBC service checks that the columns of the Nirva Table objet exist in the target database table. For all the common column, the service creates a SELECT query with an always false WHERE clause (thus returning an empty result set). This result set is used to add the rows to the database table.

To insert the rows the JDBC service works in transaction mode. In case an error occures a rollback is done and the database remains unchanged. Otherwise, once all the rows have been successfully written to the database a final commit is done

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

TABLE — Name of the table for new records. The table name is generally composed of 2 parts: *owner.table* where *owner* is the owner of the table and *table* is the name of the table. For some drivers owner is not necessary. The owner of a table can be retrieved with the TABLE:LIST command. If the owner or the table name contains a blank character, both must be enquoted with the quote character returned by the SOURCE:INFO or TABLE:LIST commands.

RECORDS Name of the NIRVA table object containing new records to insert into the table. The NIRVA object must be in the input container.

SQL

JDBC:SOURCE:SQL

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command executes any SQL statement.

It works in 2 different modes following the SELECT option.

If the SELECT option is set, the command returns information exactly in the same way as a SELECT command. So the result set stays opened and the QUERY class commands can be used to access data. For example, this command is used internally in order to get the “auto_increment” flag of the MYSQL columns (the SQL query is “*SHOW TABLE STATUS WHERE Name='TableName'*”).

If the SELECT command is not set, the SQL command just sends the order to the SGBD without requiring any result.

The SQL command can also be used to execute a stored procedure that returns a result set or not. If the stored procedure doesn’t return parameters, the SQL command must be used without the SELECT option. If the stored procedure returns a result set, the command must be used with the SELECT set to “YES”.

Here is an example of calling a stored procedure returning a value with MySQL :

Definition of the stored procedure on SQL server side that searches into a table and returns the number of records found:

```
CREATE PROCEDURE `GetStateCount`(ParamState int)
BEGIN
  SELECT Count(*) AS StateCount
  FROM test
  WHERE status = ParamState;
END
```

For searching from Nirva, here is the necessary commands (we suppose that the data source is opened):

```
NV_CMD=|JDBC:SOURCE:SQL| SQL=|CALL GetStateCount(200)| SELECT=|YES|
NV_CMD=|JDBC:QUERY:GET_RECORDS| WHAT=|ALL|
```

This returns a result set containing one column named “StateCount”.

The possibility to return values from stored procedures is driver dependant and some drivers may not accept it.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

SQL	SQL statement to execute.
SELECT	SELECT option. If this parameter is set to “YES”, the SQL order is processed like a SELECT command (the result set stay opened and resulting data can be get by using the QUERY class commands). If the SELECT parameter is set to “NO”, the SQL order is just sent to the SGBD without requiring any result. There is no result set. The default is “NO”.
ERROR_NO_DOC	This parameter has a meaning only when the SELECT parameter has been set to “YES”. When ERROR_NO_DOC is set to “NO”, the command doesn’t produce an error if no documents has been found on the database. At this time, the resulting object QUERY_RESULT is created but there is no query identifier and the NUM_RECORDS key is set to 0. When ERROR_NO_DOC is set to “YES”, the command produces an error if no documents has been found. The resulting objects are then not created. The ERROR_NO_DOC parameter implies checking the size of the result set. The default is “YES”.
NUM_DOCS_METHOD	This parameter has a meaning only when the SELECT parameter has been set to “YES”. This parameter defines which method to use for counting documents. It can take values “LOOP” or “SQL”. If “LOOP”, a walk through the result set is done (the performance of the operation is driver dependant). If “SQL”, the service does a SELECT COUNT() query using the original query.

Objects created

SQL_RESULT

This object is created only when the SELECT option is not set. This is a Nirva indexed string list object containing following keys:

- "COMMAND" contains the SQL statement.
- "RESULT" gives the status of the statement. For now, this status is always set to "OK".
- "AFFECTED" gives the number of rows affected by the SQL order when this one is an UPDATE, DELETE or INSERT statement. This

information is not implemented in all JDBC drivers. If not available, the value is 0.

QUERY_RESULT This object is created only when the SELECT option is set. This is a Nirva indexed string list object containing following keys:

- "IDENTIFIER" is the query identifier used in other QUERY class commands.
- "NUM_COLUMNS" is the number of columns returned by this statement. The column names are then given by the QUERY_COLUMNS return object.
- "SQL" is the SQL statement.
- "TABLE" is the table name.
- "NUM_RECORDS" is the number of records the query returned. How it is evaluated depends on the parameter NUM_DOCS_METHOD. When set to "SQL" the number is calculated using a SELECT COUNT() query. When set to "LOOP" the value is calculated using the result set (or an auxilary result set when the original one is FORWARD_ONLY). Otherwise the result is 0 (no counting is done).
- "FETCH" can take the value "FORWARDONLY" if the data source accepts only forward cursors or "ANY" otherwise.

QUERY_COLUMNS This object is created only when the SELECT option is set. This is a Nirva string list object containing the list of columns retrieved by the SQL statement.

SELECT

JDBC:SOURCE:SELECT

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command executes an SQL SELECT statement. Please read the SQL reference for the syntax of SQL statements. The returned query becomes the current query. The name of the table eventually used in the select statement is generally composed of 2 parts: *owner.table* where *owner* is the owner of the table and *table* is the name of the table. For some drivers owner is not necessary. The owner of a table can be retrieved with the TABLE:LIST command. If the owner or the table name contains a blank character, both must be enquoted with the quote character returned by the SOURCE:INFO or TABLE:LIST commands.

The command creates a NIRVA indexed string list object that gives some information about the query and a NIRVA string list object that gives the list of columns.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

SELECT — Complete SELECT SQL statement to proceed including the SELECT keyword at the beginning.

ERROR_NO_DOC — When set to “NO”, the command doesn’t produce an error if no documents has been found on the database. At this time, the resulting object QUERY_RESULT is created but there is no query identifier and the NUM_RECORDS key is set to 0. When ERROR_NO_DOC is set to “YES”, the command produces an error if no documents has been found. The resulting objects are then not created.

The default is “YES”.

NUM_DOCS_METHOD — This parameter defines which method to use for counting documents. It can take values “LOOP” or “SQL”. If “LOOP”, Nirva is directly counting by moving from first to last record. If “SQL”, Nirva is adding some SQL code to the given query in order to retrieve the number of documents. “LOOP” is faster in terms of database server resource but is slower on Nirva server CPU because it has to iterate threw all records. SQL is fast at Nirva level but slower on database server because it has to execute 2 queries. You can use generally the “SQL” method except if you know by advance that you will get a small number of records. At this time use the “LOOP” method. The default is SQL.

Objects created

QUERY_RESULT — This is a Nirva indexed string list object containing following keys:

- "IDENTIFIER" is the query identifier used in other QUERY class commands.
- "NUM_COLUMNS" is the number of columns returned by this statement. The column names are then given by the QUERY_COLUMNS return object.
- "SQL" is the SQL statement.
- "TABLE" is the table name.
- "NUM_RECORDS" is the number of records found. This value is generally correct but some drivers may be not able to return it. At this time, it’s set to 0 and the command QUERY:NUM_RECORDS may be used to retrieve the exact number of records. In fact, the SELECT

command returns the number of records via a call to the SQL COUNT() function.

- "FETCH" can take the value "FORWARDONLY" if the data source accepts only forward cursors or "ANY" otherwise.

QUERY_COLUMNS This is a Nirva string list object containing the list of columns retrieved by the SELECT statement.

TABLE class

COLUMN_LIST

JDBC:TABLE:COLUMN_LIST

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command returns the list of columns of a single table of the current data source. The command creates a table object returning following column information:

- Name.
- Description.
- Data type.
- Type name.
- Precision.
- Length.
- Scale.
- Radix.
- Nullable flag.
- Automatic flag.
- Primary flag.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

TABLE This parameter is the real name of the table to get columns without quote characters or owner name. This parameter is mandatory.

Objects created

COLUMN_LIST This is a Nirva table object containing following columns:

- "NAME" is the column name. The driver returns an empty string for a column that does not have a name.
- "DESCRIPTION" is a a description of the column.
- "DATATYPE" is the SQL data type. This can be an JDBC SQL data type or a driver-specific SQL data type. Please consult the JDBC or SQL references for available data types.
- "TYPENAME" is the data source – dependent data type name; for example, “CHAR”, “VARCHAR”, “MONEY”, “LONG VARBINAR”, or “CHAR ( ) FOR BIT DATA”.
- "PRECISION" is the column size.
- "LENGTH" is the length of the column.
- "SCALE" is the number of decimal digits of the column.
- "RADIX" is for numeric data types, either 10 or 2. If it is 10, the values in PRECISION and SCALE give the number of decimal digits allowed for the column. For example, a DECIMAL(12,5) column would return a RADIX of 10, a PRECISION of 12, and a SCALE of 5; a FLOAT column could return a RADIX of 10, a PRECISION of 15 and a SCALE of NULL. If it is 2, the values in PRECISION and SCALE give the number of bits allowed in the column. For example, a FLOAT column could return a RADIX of 2, a PRECISION of 53, and a SCALE of NULL. NULL is returned for data types where RADIX is not applicable.
- "NULLABLE" is set to "Yes" if the column can be empty. Otherwise, it is set to No.
- "AUTO" is set to "Yes" if the column is automatically calculated by the database system. Otherwise, it is set to No.
- "PRIMARY" is set to "Yes" if the column is a primary key. Otherwise, it is set to No.

QUERY class

The QUERY class provides commands for queries. A query is the result set generated by a SOURCE:SELECT command. All commands of the QUERY class require the SOURCE_QUERY permission when used from the browser.

GET_CURRENT

JDBC:QUERY:GET_CURRENT

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command retrieves the current query identifier if there is one.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

None

Objects created

QUERY This is a Nirva string object containing the current query identifier.

GET_RECORDS

JDBC:QUERY:GET_RECORDS

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command gets the requested records from the current query. The current query must exist.

The records go to a table object and the table column names of the NIRVA object correspond to the table column names of the data source.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

RECORDS — Name of the NIRVA object that will contain the selected documents. This can be a file or table object following the value of the FILE parameter. The default is “RECORDS”.

WHAT — Records to get. This parameter can take the following values:

- ALL : retrieves all the records of the query. This is the default.
- FIRST: retrieves the first record.
- LAST: retrieves the last record.
- NEXT: retrieves the next record.
- PREV: retrieves the previous record.
- CURRENT: retrieves the current record.
- n: retrieves nth record. The first record has number 1.
- n-m: retrieves records number n to m. The first record has number 1.

ROWBUFSIZE — Size of the row buffer. This is only available with Nirva version at least 2.5.011. It allows to group records into a buffer and to send a unique command to Nirva to populate the resulting table object. This can save time when retreiving an important amount of records. The default value is 20. This parameter has meaning only when table output is requested.

Objects created

RECORDS — Selected records. The name of this object can be changed by using the RECORDS parameter. This is a Nirva table object. Please see the command description for information about the RECORDS object.

NUM_RECORDS

JDBC:QUERY:NUM_RECORDS

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	Yes

Description

This command returns the number of records of the current query. This command may be used if the JDBC:SOURCE:SELECT command did not return a value for the number of found records. Depending of the driver used, this command can be slow.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

EXACT — If this parameter is set to “YES”, the database service will use result set functions in order to retrieve the number of documents of the current query. Otherwise, it uses the result of an SQL COUNT() function call.

The default value is “NO”.

Objects created

NUM_RECORDS — This is a Nirva string object containing the number of records of the current query.

PURGE

JDBC:QUERY:PURGE

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command removes the given query. After being removed, a query is not usable any more. All queries maintained by a given data source are automatically removed when the data source is closed. If the removed query is the current one, then the current query is undefined.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

QUERY — Query identifier to remove. The query identifier is returned by the SOURCE:SELECT command. If this parameter is not provided, the current query is removed. Then, the current query is undefined.

If this parameter is set to “ALL”, all the queries are removed.

SET_CURRENT

JDBC:QUERY:SET_CURRENT

Source	Use Input Container	Use Output Container
Client Web Procedure Service	No	No

Description

This command sets the current query. This command is only useful if several queries have to be used (and if the source accepts it) because the SOURCE:SELECT command always sets the result query as the current one.

Permissions

SOURCE_QUERY (when called from the browser)

Parameters

QUERY — Query identifier to set as the current one. The query identifier is returned by the SOURCE:SELECT command.