

# rawSQL



Apr 07, 2024 13:24

## Supported Clients

SmartClient WebClient NGClient

## Methods Summary

Boolean	<code>executeSQL(serverName, sql)</code>	Execute any SQL, returns true if successful.
Boolean	<code>executeSQL(serverName, sql, sql_args)</code>	Execute any SQL, returns true if successful.
Array	<code>executeStoredProcedure(serverName, procedureDeclaration, arguments, maxNumberOfRowsToRetrieve)</code>	Execute a stored procedure, return all created result sets.
JSDataset	<code>executeStoredProcedure(serverName, procedureDeclaration, arguments, inOutDirectionality, maxNumberOfRowsToRetrieve)</code>	Execute a stored procedure.
Boolean	<code>flushAllClientsCache(serverName, tableName)</code>	Flush cached database data.
Exception	<code>getException()</code>	If the result from a function was false, it will return the exception object.
Boolean	<code>notifyDataChange(serverName, tableName, pksDataset, action)</code>	Notify clients about changes in records, based on pk(s).

## Methods Details

### executeSQL(serverName, sql)

Execute any SQL, returns true if successful.

#### Parameters

`String` serverName the name of the server  
`String` sql the sql query to execute

#### Returns

`Boolean`

#### Supported Clients

SmartClient,WebClient,NGClient

#### Sample

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var country = 'NL'
var done = plugins.rawSQL.executeSQL("example_data","update employees set country = ?", [country])
if (done)
{
    //flush is required when changes are made in db
    plugins.rawSQL.flushAllClientsCache("example_data","employees")
}
else
{
    var msg = plugins.rawSQL.getException().getMessage(); //see exception node for more info about the
exception obj
    plugins.dialogs.showErrorDialog('Error', 'SQL exception: '+msg, 'Ok')
}

// Note that when this function is used to create a new table in the database, this table will only be seen by
// the Servoy Application Server when the table name starts with 'temp_', otherwise a server restart is needed.
```

### executeSQL(serverName, sql, sql\_args)

Execute any SQL, returns true if successful.

**Parameters**

`String` `serverName` the name of the server  
`String` `sql` the sql query to execute  
`Array` `sql_args` the arguments for the query

**Returns**

`Boolean`

**Supported Clients**

SmartClient,WebClient,NGClient

**Sample**

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var country = 'NL'
var done = plugins.rawQuery.executeSQL("example_data","employees","update employees set country = ?", [country])
if (done)
{
    //flush is required when changes are made in db
    plugins.rawQuery.flushAllClientsCache("example_data","employees")
}
else
{
    var msg = plugins.rawQuery.getException().getMessage(); //see exception node for more info about the
exception obj
    plugins.dialogs.showErrorDialog('Error', 'SQL exception: '+msg, 'Ok')
}

// Note that when this function is used to create a new table in the database, this table will only be seen by
// the Servoy Application Server when the table name starts with 'temp_', otherwise a server restart is needed.

```

**executeStoredProcure(serverName, procedureDeclaration, arguments, maxNumberOfRowsToRetrieve)**

Execute a stored procedure, return all created result sets.

**Parameters**

`String` `serverName` ;  
`String` `procedureDeclaration` ;  
`Array` `arguments` ;  
`Number` `maxNumberOfRowsToRetrieve` ;

**Returns**

`Array` the result sets created by the procedure.

**Supported Clients**

SmartClient,WebClient,NGClient

**Sample**

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var maxReturnedRows = 10; //useful to limit number of rows
var procedure_declaration = '{ get_unpaid_orders_and_their_customers(?) }'
var args = [42]
var datasets = plugins.rawSQL.executeStoredProcedure(databaseManager.getDataSourceServerName(controller.
getDataSource()), procedure_declaration, args, maxReturnedRows);
for (var i = 0; i < datasets.length; i++) {
    var ds = datasets[i]
    // process dataset
}

```

**executeStoredProcedure(serverName, procedureDeclaration, arguments, inOutDirectionality, maxNumberOfRowsToRetrieve)**

Execute a stored procedure.

**Parameters**

```

String  serverName          ;
String  procedureDeclaration ;
Array   arguments           ;
Array   inOutDirectionality ;
Number  maxNumberOfRowsToRetrieve;

```

**Returns**

[JSDataset](#) a dataset with output (in case of output data) or the last result set executed by the procedure.

**Supported Clients**

SmartClient, WebClient, NGClient

**Sample**

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var maxReturnedRows = 10; //useful to limit number of rows
var procedure_declaration = '{?=calculate_interest_rate(?)}'
// define the direction, a 0 for input data, a 1 for output data
var typesArray = [1, 0];
// define the types and values, a value for input data, a sql-type for output data
var args = [java.sql.Types.NUMERIC, 3000]
// A dataset is returned, when no output-parameters defined, the last select-result in the procedure will be
returned.
// When one or more output-parameters are defined, the dataset will contain 1 row with the output data.
var dataset = plugins.rawSQL.executeStoredProcedure(databaseManager.getDataSourceServerName(controller.
getDataSource()), procedure_declaration, args, typesArray, maxReturnedRows);
var interest_rate = dataset.getValue(1, 1);

```

**flushAllClientsCache(serverName, tableName)**

Flush cached database data. Use with extreme care, its affecting the performance of clients!

**Parameters**

```

String serverName;
String tableName ;

```

**Returns**

[Boolean](#)

**Supported Clients**

SmartClient,WebClient,NGClient

**Sample**

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var country = 'NL'
var done = plugins.rawSQL.executeSQL("example_data","employees","update employees set country = ?", [country])
if (done)
{
    //flush is required when changes are made in db
    plugins.rawSQL.flushAllClientsCache("example_data","employees")
}
else
{
    var msg = plugins.rawSQL.getException().getMessage(); //see exception node for more info about the
exception obj
    plugins.dialogs.showErrorDialog('Error', 'SQL exception: '+msg, 'Ok')
}

// Note that when this function is used to create a new table in the database, this table will only be seen by
// the Servoy Application Server when the table name starts with 'temp_', otherwise a server restart is needed.

```

**getException()**

If the result from a function was false, it will return the exception object.

**Returns**[Exception](#)**Supported Clients**

SmartClient,WebClient,NGClient

**Sample**

```

/*****
WARNING! You can cause data loss or serious data integrity compromises!
You should have a THOROUGH understanding of both SQL and your backend
database (and other interfaces that may use that backend) BEFORE YOU USE
ANY OF THESE COMMANDS.
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS
*****/

var country = 'NL'
var done = plugins.rawSQL.executeSQL("example_data","employees","update employees set country = ?", [country])
if (done)
{
    //flush is required when changes are made in db
    plugins.rawSQL.flushAllClientsCache("example_data","employees")
}
else
{
    var msg = plugins.rawSQL.getException().getMessage(); //see exception node for more info about the
exception obj
    plugins.dialogs.showErrorDialog('Error', 'SQL exception: '+msg, 'Ok')
}

// Note that when this function is used to create a new table in the database, this table will only be seen by
// the Servoy Application Server when the table name starts with 'temp_', otherwise a server restart is needed.

```

**notifyDataChange(serverName, tableName, pksDataset, action)**

Notify clients about changes in records, based on pk(s). Use with extreme care, its affecting the performance of clients!

---

**Parameters**

String     serverName;  
String     tableName;  
JSDataSet pksDataset;  
Number    action     ;

**Returns**

Boolean

**Supported Clients**

SmartClient,WebClient,NGClient

**Sample**

```
/******  
WARNING! You can cause data loss or serious data integrity compromises!  
You should have a THOROUGH understanding of both SQL and your backend  
database (and other interfaces that may use that backend) BEFORE YOU USE  
ANY OF THESE COMMANDS.  
You should also READ THE DOCUMENTATION BEFORE USING ANY OF THESE COMMANDS  
*****/  
  
var action = SQL_ACTION_TYPES.DELETE_ACTION //pks deleted  
//var action = SQL_ACTION_TYPES.INSERT_ACTION //pks inserted  
//var action = SQL_ACTION_TYPES.UPDATE_ACTION //pks updates  
var pksdataset = databaseManager.convertToDataSet(new Array(12,15,16,21))  
var ok = plugins.rawSQL.notifyDataChange(databaseManager.getDataSourceServerName(controller.getDataSource()),  
'employees', pksdataset,action)
```