

# Contents

|    |  |    |
|----|--|----|
| 1  | sqlxx .....  | 3  |
| 2  | CSQLResult::~CSQLResult — <i>SQL Result Destructor.</i> ....                   | 4  |
| 3  | CSQLResult::close — <i>Closes database connection.</i> .....                   | 5  |
| 4  | CSQLResult::getError — <i>Returns last error message.</i> ....                 | 6  |
| 5  | CSQLResult::query — <i>Query Database.</i> .....                               | 7  |
| 6  | CSQLResult::getNumCols — <i>Return number of columns.</i> .                    | 8  |
| 7  | CSQLResult::getNumRows — <i>Return number of rows.</i> ...                     | 9  |
| 8  | CSQLResult::getColName — <i>Returns column name.</i> .....                     | 10 |
| 9  | CSQLResult::getColType — <i>Returns column type.</i> .....                     | 11 |
| 10 | CSQLResult::fetch — <i>Fetch a row of data.</i> .....                          | 12 |
| 11 | CSQLResult::get — <i>Returns column value as string.</i> .....                 | 13 |
| 12 | CSQLResult::get — <i>Returns column value as string.</i> .....                 | 14 |
| 13 | CSQLResult::getInt — <i>Returns column value as long integer.</i> .....        | 15 |
| 14 | CSQLResult::getInt — <i>Returns column value as long integer.</i> .....        | 16 |
| 15 | CSQLResult::getFloat — <i>Returns column value as float.</i> ..                | 17 |
| 16 | CSQLResult::getFloat — <i>Returns column value as float.</i> ..                | 18 |
| 17 | CSQLResult::isNull — <i>Determine if column value is NULL.</i> 19              |    |
| 18 | CSQLResult::isNull — <i>Determine if column value is NULL.</i> 20              |    |
| 19 | CSQL::CSQL — <i>SQL Constructor.</i> .....                                     | 21 |
| 20 | CSQL::~CSQL — <i>SQL Destructor.</i> .....                                     | 22 |
| 21 | CSQL::getError — <i>Get last error message.</i> .....                          | 23 |
| 22 | CSQL::addResult — <i>Connects a CSQLResult to this CSQL object.</i> .....      | 24 |
| 23 | CSQL::delResult — <i>Disconnects a CSQLResult from this CSQL object.</i> ..... | 25 |
| 24 | CSQL::setDSN — <i>Sets DSN.</i> .....  | 26 |
| 25 | CSQL::setHostname — <i>Sets Hostname.</i> .....                                | 27 |
| 26 | CSQL::setSocket — <i>Sets Socket.</i> .....                                    | 28 |
| 27 | CSQL::setPort — <i>Sets Port.</i> .....  | 29 |
| 28 | CSQL::setPort — <i>Sets Port.</i> .....  | 30 |
| 29 | CSQL::setDatabase — <i>Sets database name.</i> .....                           | 31 |
| 30 | CSQL::setDriver — <i>Sets ODBC driver.</i> .....                               | 32 |
| 31 | CSQL::setUsername — <i>Sets database username.</i> .....                       | 33 |
| 32 | CSQL::setPassword — <i>Sets database password.</i> .....                       | 34 |
| 33 | CSQL::setOptions — <i>Sets database options.</i> .....                         | 35 |
| 34 | CSQL::setType — <i>Sets database type.</i> .....                               | 36 |
| 35 | CSQL::getDSN — <i>Returns DSN.</i> .....                                       | 37 |
| 36 | CSQL::getHostname — <i>Returns Hostname.</i> .....                             | 38 |
| 37 | CSQL::getSocket — <i>Returns Socket.</i> .....                                 | 39 |

## Contents

---

|    |  |    |
|----|--|----|
| 38 | CSQL::getPort — <i>Returns Port.</i> .....                       | 40 |
| 39 | CSQL::getPortN — <i>Returns Port.</i> .....                      | 41 |
| 40 | CSQL::getDatabase — <i>Returns database name.</i> .....          | 42 |
| 41 | CSQL::getDriver — <i>Returns ODBC driver.</i> .....              | 43 |
| 42 | CSQL::getUsername — <i>Returns database username.</i> .....      | 44 |
| 43 | CSQL::getPassword — <i>Returns database password.</i> .....      | 45 |
| 44 | CSQL::getOptions — <i>Returns ODBC options.</i> .....            | 46 |
| 45 | CSQL::getType — <i>Returns database type.</i> .....              | 47 |
| 46 | CSQL::connect — <i>Connect to database.</i> .....                | 48 |
| 47 | CSQL::disconnect — <i>Disconnects from database.</i> .....       | 49 |
| 48 | CSQL::isConnected — <i>Determine if connected.</i> .....         | 50 |
| 49 | CSQL::openQuery — <i>Opens an SQL Query.</i> .....               | 51 |
| 50 | CSQL::execQuery — <i>Execute SQL query.</i> .....                | 52 |
| 51 | CSQL::closeQuery — <i>Closes SQL query.</i> .....                | 53 |
| 52 | CSQL::getNewID — <i>Returns next available Primary key.</i> .    | 54 |
| 53 | CSQL::getQueries — <i>Returns vector with active queries.</i> .. | 55 |
| 54 | CSQL::countQueries — <i>Count active queries.</i> .....          | 56 |

**sqlxx**

This library contains C++ classes for easy database access via ODBC using libiodbc2 or via native access. Currently the native access to mysql and postgresql databases is implemented.

**Author:** Klaus Reimer <k@ailis.de>  
**Version:** 2.1.0

2

**CSQLException::~~CSQLException (void)***SQL Result Destructor.*

SQL Result Destructor. This is the destructor of the SQL Result. It calls CSQLException::close() to close the database connection.

**3**

`void CSQLResult::close (void)`

*Closes database connection.*

Closes database connection. This method closes the database connection and frees all allocated memory.

4

`string CSQLException::getError (void)`

*Returns last error message.*

Returns last error message. This method returns the last error message of the SQL Result.

5

```
void CSQLResult::query (const string &sQuery)
```

*Query Database.*

Query Database. This method sends the SQL query **sQuery** to the database. You don't need to use this method on your own. Use CSQL::openQuery() instead.

**6****unsigned int CSQLException::getNumCols (void)***Return number of columns.*

Return number of columns. This method returns the number of columns of the current SQL Result.



7

unsigned long **CSQLResult::getNumRows** (void)

*Return number of rows.*

Return number of rows. This method returns the number of rows of the current SQL Result.

8

```
string CSQLException::getColName (const unsigned int  
                                iIndex)
```

*Returns column name.*

Returns column name. Return the name of the column with the index **iIndex**.

9

```
int CSQLException::getColType (const unsigned int iIn-  
                                dex)
```

*Returns column type.*

Returns column type. Returns the type of the column with index **iIndex**. The meaning of the type is database dependent.

10

`bool CSQLException::fetch (void)`

*Fetch a row of data.*

Fetch a row of data. This method fetches a row of data from the current SQLException.

11

`string CSQLException::get (const unsigned int iIndex)`

*Returns column value as string.*

Returns column value as string. Returns the value of the column with index **iIndex** as string.

12

`string CSQLException::get (const string sField)`

*Returns column value as string.*

Returns column value as string. Returns the value of the column named **sField** as string.

**13**

`long CSQLException::getInt (const unsigned int iIndex)`

*Returns column value as long integer.*

Returns column value as long integer. Returns the value of the column with index **iIndex** as long integer.

14

`long CSQLException::getInt (const string sField)`

*Returns column value as long integer.*

Returns column value as long integer. Returns the value of the column named **sField** as long integer.



**15**

`float CSQLException::getFloat (const unsigned int iIndex)`

*Returns column value as float.*

Returns column value as float. Returns the value of the column with index **iIndex** as long float.

16

`float CSQLResult::getFloat (const string sField)`

*Returns column value as float.*

Returns column value as float. Returns the value of the column named **sField** as long float.

17

`bool CSQLException::isNull (const unsigned int iIndex)`

*Determine if column value is NULL.*

Determine if column value is NULL. Returns true if the value of the column with index **iIndex** is NULL. Returns false if not.

18

`bool CSQLException::isNull (const string sField)`

*Determine if column value is NULL.*

Determine if column value is NULL. Returns true if the value of the column named **sField** is NULL. Returns false if not.

**19****CSQL::CSQL (void)***SQL Constructor.*

SQL Constructor. This is the constructor of the class CSQL. It initializes the SQL object with empty values.

20

**CSQL::~~CSQL (void)***SQL Destructor.*

SQL Destructor. This is the destructor of the CSQL class. It automatically disconnects all open database connections and frees all allocated memories of all connected CSQLResult objects.

21

`string CSQL::getError (void* Result=NULL)`

*Get last error message.*

Get last error message. This method returns the last error message. If no parameter is given the last database related error message is returned. If a Pointer to a CSQLResult object is specified, the last error message of this SQLResult-Object is returned. Don't use this method if possible. Catch the `sqlxx_error` exceptions which already contains the complete error message.

22

```
void CSQL::addResult (CSQLResult* Result)
```

*Connects a CSQLResult to this CSQL object.*

Connects a CSQLResult to this CSQL object. This method is used internally to connect a CSQLResult object to this CSQL object. The connected CSQLResult objects are freed automatically by the CSQL destructor.



**23**

`void CSQL::delResult (CSQLResult* Result)`

*Disconnects a CSQLResult from this CSQL object.*

Disconnects a CSQLResult from this CSQL object. This method is used internally to disconnect a CSQLResult object from this CSQL object. The connected CSQLResult objects are freed automatically by the CSQL destructor.

24

```
void CSQL::setDSN (const string &sNewDSN)
```

*Sets DSN.*

Sets DSN. Sets the DSN to **sNewDSN**. The DSN is only used by ODBC connections.

25

```
void CSQL::setHostname (const    string    &sNewHost-
                        name)
```

*Sets Hostname.*

Sets Hostname. Sets Hostname to **sNewHostname**.

**26**

```
void CSQL::setSocket (const string &sNewSocket)
```

*Sets Socket.*

Sets Socket. Sets Socket to **sNewSocket**.

**27**

```
void CSQL::setPort (const string &sNewPort)
```

*Sets Port.*

Sets Port. Sets Ports to **sNewPort**. Use this method if you want to specify the port address as a string.

28

```
void CSQL::setPort (const unsigned int iNewPort)
```

*Sets Port.*

Sets Port. Sets Ports to **sNewPort**. Use this method if you want to specify the port address as an integer.

**29**

```
void CSQL::setDatabase (const string &sNewDatabase)
```

*Sets database name.*

Sets database name. Sets database name to **sNewDatabase**.

**30**

```
void CSQL::setDriver (const string &sNewDriver)
```

*Sets ODBC driver.*

Sets ODBC driver. Sets odbc driver to **sNewDriver**. This value is only used for ODBC connections



**31**

```
void CSQL::setUsername (const    string    &sNewUser-
                        name)
```

*Sets database username.*

Sets database username. Sets database username to **sNewUsername**.

**32**

```
void CSQL::setPassword (const string &sNewPassword)
```

*Sets database password.*

Sets database password.      Sets database password to **sNewPassword**.

**33**

```
void CSQL::setOptions (const string &sNewOptions)
```

*Sets database options.*

Sets database options. Sets database options to **sNewOptions**. These options are only used by ODBC connections. Your ODBC driver documentation will tell you what options are possible.

34

```
void CSQL::setType (const unsigned short iNewType)
```

*Sets database type.*

Sets database type. Sets database type to **iNewType**. This can be SQLXX\_MYSQL, SQLXX\_POSTGRES or SQLXX\_ODBC. But this is dependent on how you have compiled the sqlxx library.

**35**

`string CSQL::getDSN (void)`

*Returns DSN.*

Returns DSN. Returns the DSN.

**36**

`string CSQL::getHostname (void)`

*Returns Hostname.*

Returns Hostname. Returns the Hostname.

**37****string CSQL::getSocket (void)***Returns Socket.*

Returns Socket. Returns the Socket.

**38****string CSQL::getPort (void)***Returns Port.*

Returns Port. Returns the port address as string.



**39**

`unsigned int CSQL::getPortN (void)`

*Returns Port.*

Returns Port. Returns the port address as unsigned integer.

40

`string CSQL::getDatabase (void)`

*Returns database name.*

Returns database name. Returns the database name.

41

`string CSQL::getDriver (void)`

*Returns ODBC driver.*

Returns ODBC driver. Returns the filename to the ODBC driver.

42

`string CSQL::getUsername (void)`

*Returns database username.*

Returns database username. Returns the database username.

43

`string CSQL::getPassword (void)`

*Returns database password.*

Returns database password. Returns the database password.

44

`string CSQL::getOptions (void)`

*Returns ODBC options.*

Returns ODBC options. Returns the ODBC options.

45

`unsigned short CSQL::getType (void)`

*Returns database type.*

Returns database type. Returns the database type.

46

`void CSQL::connect (void)`

*Connect to database.*

Connect to database. This method inititates the connection to the database.



47

`void CSQL::disconnect (void)`

*Disconnects from database.*

Disconnects from database. This methode closes the connection to the database.

48

`bool CSQL::isConnected (void)`

*Determine if connected.*

Determine if connected. This method returns true if the database connection is established. False if not.

**49**

```
CSQLResult* CSQL::openQuery (const string &sQuery,  
                               const int    iBuffer-  
                               Size=1024)
```

*Opens an SQL Query.*

Opens an SQL Query. This method opens a new SQL query **sQuery** and returns a pointer to a CSQLResult object.

50

```
void CSQL::execQuery (const string &sQuery)
```

*Execute SQL query.*

Execute SQL query. This method executes the SQL query **sQuery**.

51

```
void CSQL::closeQuery (const CSQLResult* DBResult)
```

*Closes SQL query.*

Closes SQL query. Closes the SQL query **DBResult**.

52

```
long CSQL::getNewID (const string &sTable, const  
                    string &sIDField)
```

*Returns next available Primary key.*

Returns next available Primary key. This method returns the next available primary key value for the field **sIDField** in table **sTable**

**53**

`vector< CSQLResult * > & CSQL::getQueries (void)`

*Returns vector with active queries.*

Returns vector with active queries. This method returns a vector with pointers to all active queries of this CSQL object.

54

`int CSQL::countQueries (void)`

*Count active queries.*

Count active queries. Returns the number of active SQL queries of this CSQL object.