# general_ledger

Generated by Doxygen 1.8.1.2

Thu Jun 19 2014 20:57:09

# Contents


**1 General Ledger. 1**

**2 Todo List 3**

**3 Bug List 5**

**4 Module Index 7**

4.1 Modules . . . 7

**5 Class Index 9**

5.1 Class Hierarchy . . . 9

**6 Class Index 11**

6.1 Class List . . . 11

**7 File Index 13**

7.1 File List . . . 13

**8 Module Documentation 15**

8.1 Program configuration module . . . 15

8.1.1 Detailed Description . . . 15

8.2 Database interaction module . . . 16

8.2.1 Detailed Description . . . 17

8.2.2 Function Documentation . . . 17

8.2.2.1 get_connection . . . 17

8.2.2.2 get_database_type . . . 17

8.2.2.3 get_field_names . . . 17

8.2.2.4 get_row . . . 17

8.3 SQL statements module . . . 19

8.3.1 Detailed Description . . . 19

8.4 General Ledger database module. . . . 20

8.4.1 Detailed Description . . . 20

8.5 General purpose utilities. . . . 21

8.5.1 Detailed Description . . . 21

8.5.2 Function Documentation . . . 21
8.5.2.1 content_lines . . . 21
8.5.2.2 join . . . 21
8.5.2.3 next_content_line . . . 22
8.5.2.4 replace . . . 22
8.5.2.5 split . . . 22
8.5.2.6 split . . . 23
8.5.2.7 split_lines . . . 23
8.5.2.8 trim . . . 23
8.5.2.9 trim_back . . . 23
8.5.2.10 trim_front . . . 24
8.6 Database program. . . . 25
8.6.1 Detailed Description . . . 25
8.6.2 Function Documentation . . . 25
8.6.2.1 check_db_parameters . . . 25
8.6.2.2 check_help_and_version . . . 25
8.6.2.3 login . . . 26
8.6.2.4 main . . . 26
8.6.2.5 set_configuration . . . 26
8.7 Reporting program. . . . 27
8.7.1 Detailed Description . . . 27
8.7.2 Function Documentation . . . 27
8.7.2.1 login . . . 27
8.7.2.2 main . . . 27
8.7.2.3 set_configuration . . . 28
8.8 User administration program. . . . 29
8.8.1 Detailed Description . . . 29
8.8.2 Function Documentation . . . 29
8.8.2.1 check_db_parameters . . . 29
8.8.2.2 check_help_and_version . . . 30
8.8.2.3 check_user_password . . . 30
8.8.2.4 enable_user . . . 30
8.8.2.5 get_user . . . 30
8.8.2.6 login . . . 30
8.8.2.7 main . . . 31
8.8.2.8 set_configuration . . . 31
8.8.2.9 set_user_password . . . 31
8.8.2.10 show_user_details . . . 31

**9 Class Documentation** **33**

9.1 genleg::Config Class Reference . . . 33
9.1.1 Detailed Description . . . 33
9.1.2 Constructor & Destructor Documentation . . . 33
9.1.2.1 Config . . . 33
9.1.2.2 ∼Config . . . 34
9.1.3 Member Function Documentation . . . 34
9.1.3.1 add_cmdline_option . . . 34
9.1.3.2 is_set . . . 34
9.1.3.3 operator[] . . . 34
9.1.3.4 populate_from_cmdline . . . 34
9.1.3.5 populate_from_file . . . 35
9.1.4 Member Data Documentation . . . 35
9.1.4.1 m_opts_set . . . 35
9.1.4.2 m_opts_supp . . . 35
9.2 genleg::ConfigBadConfigFile Class Reference . . . 35
9.2.1 Detailed Description . . . 36
9.2.2 Constructor & Destructor Documentation . . . 36
9.2.2.1 ConfigBadConfigFile . . . 36
9.3 genleg::ConfigBadOption Class Reference . . . 37
9.3.1 Detailed Description . . . 37
9.3.2 Constructor & Destructor Documentation . . . 37
9.3.2.1 ConfigBadOption . . . 38
9.4 genleg::ConfigCouldNotOpenFile Class Reference . . . 38
9.4.1 Detailed Description . . . 39
9.4.2 Constructor & Destructor Documentation . . . 39
9.4.2.1 ConfigCouldNotOpenFile . . . 39
9.5 genleg::ConfigException Class Reference . . . 39
9.5.1 Detailed Description . . . 39
9.5.2 Constructor & Destructor Documentation . . . 40
9.5.2.1 ConfigException . . . 40
9.6 genleg::ConfigOptionNotSet Class Reference . . . 40
9.6.1 Detailed Description . . . 41
9.6.2 Constructor & Destructor Documentation . . . 41
9.6.2.1 ConfigOptionNotSet . . . 41
9.7 gldb::DBConn Class Reference . . . 41
9.7.1 Detailed Description . . . 42
9.7.2 Constructor & Destructor Documentation . . . 42
9.7.2.1 DBConn . . . 42
9.7.2.2 DBConn . . . 42
9.7.2.3 DBConn . . . 42

9.7.3 Member Function Documentation . . . 42
9.7.3.1 operator= . . . 42
9.7.3.2 operator= . . . 42
9.7.3.3 query . . . 42
9.7.3.4 select . . . 43
9.7.4 Member Data Documentation . . . 43
9.7.4.1 m_imp . . . 43
9.8 gldb::DBConnCouldNotConnect Class Reference . . . 43
9.8.1 Detailed Description . . . 44
9.8.2 Constructor & Destructor Documentation . . . 44
9.8.2.1 DBConnCouldNotConnect . . . 44
9.9 gldb::DBConnCouldNotQuery Class Reference . . . 44
9.9.1 Detailed Description . . . 45
9.9.2 Constructor & Destructor Documentation . . . 45
9.9.2.1 DBConnCouldNotQuery . . . 45
9.10 gldb::DBConnDummy Class Reference . . . 46
9.10.1 Detailed Description . . . 47
9.10.2 Constructor & Destructor Documentation . . . 47
9.10.2.1 DBConnDummy . . . 47
9.10.2.2 DBConnDummy . . . 47
9.10.2.3 ~DBConnDummy . . . 47
9.10.3 Member Function Documentation . . . 47
9.10.3.1 operator= . . . 47
9.10.3.2 query . . . 47
9.10.3.3 select . . . 47
9.11 gldb::DBConnException Class Reference . . . 48
9.11.1 Detailed Description . . . 48
9.11.2 Constructor & Destructor Documentation . . . 48
9.11.2.1 DBConnException . . . 48
9.12 gldb::DBConnImp Class Reference . . . 49
9.12.1 Detailed Description . . . 49
9.12.2 Constructor & Destructor Documentation . . . 49
9.12.2.1 DBConnImp . . . 49
9.12.2.2 ~DBConnImp . . . 49
9.12.3 Member Function Documentation . . . 50
9.12.3.1 query . . . 50
9.12.3.2 select . . . 50
9.13 gldb::DBConnMySQL Class Reference . . . 50
9.13.1 Detailed Description . . . 51
9.13.2 Constructor & Destructor Documentation . . . 51

9.13.2.1 DBConnMySQL . . . 51
9.13.2.2 DBConnMySQL . . . 52
9.13.2.3 DBConnMySQL . . . 52
9.13.2.4 ~DBConnMySQL . . . 52
9.13.3 Member Function Documentation . . . 52
9.13.3.1 operator= . . . 52
9.13.3.2 operator= . . . 52
9.13.3.3 query . . . 52
9.13.3.4 select . . . 52
9.13.4 Member Data Documentation . . . 53
9.13.4.1 m_conn . . . 53
9.14 genleg::DBSQLDummy Class Reference . . . 53
9.14.1 Detailed Description . . . 54
9.15 genleg::DBSQLMySQL Class Reference . . . 54
9.15.1 Detailed Description . . . 54
9.16 genleg::DBSQLStatements Class Reference . . . 55
9.16.1 Detailed Description . . . 56
9.16.2 Constructor & Destructor Documentation . . . 56
9.16.2.1 DBSQLStatements . . . 56
9.16.2.2 ~DBSQLStatements . . . 56
9.16.3 Member Function Documentation . . . 56
9.16.3.1 create_table . . . 56
9.16.3.2 create_view . . . 56
9.16.3.3 drop_table . . . 56
9.16.3.4 drop_view . . . 57
9.16.3.5 get_perms . . . 57
9.16.3.6 grant . . . 57
9.16.3.7 revoke . . . 57
9.16.3.8 update_user . . . 58
9.16.3.9 user_by_id . . . 58
9.16.3.10 user_by_username . . . 58
9.17 genleg::GLDatabase Class Reference . . . 58
9.17.1 Detailed Description . . . 60
9.17.2 Constructor & Destructor Documentation . . . 60
9.17.2.1 GLDatabase . . . 60
9.17.2.2 ~GLDatabase . . . 60
9.17.3 Member Function Documentation . . . 60
9.17.3.1 backend . . . 60
9.17.3.2 create_structure . . . 61
9.17.3.3 create_user . . . 61

9.17.3.4 destroy_structure . . . 61
9.17.3.5 get_user_by_id . . . 61
9.17.3.6 get_user_by_username . . . 61
9.17.3.7 grant . . . 62
9.17.3.8 load_sample_data . . . 62
9.17.3.9 revoke . . . 62
9.17.3.10 update_user . . . 62
9.17.4 Member Data Documentation . . . 62
9.17.4.1 m_dbc . . . 63
9.17.4.2 m_sql . . . 63
9.17.4.3 m_tables . . . 63
9.17.4.4 m_views . . . 63
9.18 genleg::GLDBException Class Reference . . . 63
9.18.1 Detailed Description . . . 63
9.18.2 Constructor & Destructor Documentation . . . 63
9.18.2.1 GLDBException . . . 63
9.19 genleg::GLUser Class Reference . . . 64
9.19.1 Detailed Description . . . 65
9.19.2 Constructor & Destructor Documentation . . . 65
9.19.2.1 GLUser . . . 65
9.19.2.2 ~GLUser . . . 65
9.19.3 Member Function Documentation . . . 65
9.19.3.1 check_password . . . 66
9.19.3.2 enabled . . . 66
9.19.3.3 firstname . . . 66
9.19.3.4 id . . . 66
9.19.3.5 lastname . . . 66
9.19.3.6 pass_hash . . . 66
9.19.3.7 pass_salt . . . 67
9.19.3.8 permissions . . . 67
9.19.3.9 set_enabled . . . 67
9.19.3.10 set_firstname . . . 67
9.19.3.11 set_lastname . . . 67
9.19.3.12 set_password . . . 67
9.19.3.13 set_username . . . 67
9.19.3.14 username . . . 68
9.19.4 Member Data Documentation . . . 68
9.19.4.1 m_enabled . . . 68
9.19.4.2 m_firstname . . . 68
9.19.4.3 m_id . . . 68

9.19.4.4 m_lastname . . . 68
9.19.4.5 m_pass_hash . . . 68
9.19.4.6 m_pass_salt . . . 68
9.19.4.7 m_perms . . . 68
9.19.4.8 m_username . . . 68
9.20 gldb::MySQLResult Class Reference . . . 69
9.20.1 Detailed Description . . . 69
9.20.2 Constructor & Destructor Documentation . . . 69
9.20.2.1 MySQLResult . . . 69
9.20.2.2 ~MySQLResult . . . 69
9.20.2.3 MySQLResult . . . 69
9.20.2.4 MySQLResult . . . 70
9.20.3 Member Function Documentation . . . 70
9.20.3.1 num_fields . . . 70
9.20.3.2 operator= . . . 70
9.20.3.3 operator= . . . 70
9.20.3.4 result . . . 70
9.20.4 Member Data Documentation . . . 70
9.20.4.1 m_num_fields . . . 70
9.20.4.2 m_result . . . 70
9.21 gldb::Table Class Reference . . . 70
9.21.1 Detailed Description . . . 72
9.21.2 Constructor & Destructor Documentation . . . 72
9.21.2.1 Table . . . 72
9.21.2.2 Table . . . 72
9.21.2.3 Table . . . 72
9.21.2.4 Table . . . 72
9.21.2.5 ~Table . . . 73
9.21.3 Member Function Documentation . . . 73
9.21.3.1 append_record . . . 73
9.21.3.2 append_record . . . 73
9.21.3.3 create_from_file . . . 73
9.21.3.4 get_field . . . 73
9.21.3.5 get_headers . . . 74
9.21.3.6 insert_query . . . 74
9.21.3.7 num_fields . . . 74
9.21.3.8 num_records . . . 74
9.21.3.9 operator= . . . 74
9.21.3.10 operator= . . . 75
9.21.3.11 operator[] . . . 75

9.21.3.12 set_quoted . . . 75
9.21.3.13 set_quoted . . . 75
9.21.4 Member Data Documentation . . . 75
9.21.4.1 m_headers . . . 75
9.21.4.2 m_quoted . . . 76
9.21.4.3 m_records . . . 76
9.22 gldb::TableBadInputFile Class Reference . . . 76
9.22.1 Detailed Description . . . 77
9.22.2 Constructor & Destructor Documentation . . . 77
9.22.2.1 TableBadInputFile . . . 77
9.23 gldb::TableCouldNotOpenInputFile Class Reference . . . 77
9.23.1 Detailed Description . . . 78
9.23.2 Constructor & Destructor Documentation . . . 78
9.23.2.1 TableCouldNotOpenInputFile . . . 78
9.24 gldb::TableException Class Reference . . . 78
9.24.1 Detailed Description . . . 79
9.24.2 Constructor & Destructor Documentation . . . 79
9.24.2.1 TableException . . . 79
9.25 gldb::TableField Class Reference . . . 79
9.25.1 Detailed Description . . . 81
9.25.2 Constructor & Destructor Documentation . . . 81
9.25.2.1 TableField . . . 81
9.25.2.2 TableField . . . 81
9.25.2.3 TableField . . . 81
9.25.2.4 TableField . . . 81
9.25.2.5 TableField . . . 81
9.25.2.6 ~TableField . . . 82
9.25.3 Member Function Documentation . . . 82
9.25.3.1 length . . . 82
9.25.3.2 operator std::string . . . 82
9.25.3.3 operator+= . . . 82
9.25.3.4 operator+= . . . 82
9.25.3.5 operator= . . . 82
9.25.3.6 operator= . . . 83
9.25.3.7 operator= . . . 83
9.25.3.8 operator= . . . 83
9.25.3.9 operator= . . . 83
9.25.3.10 operator[] . . . 84
9.25.3.11 operator[] . . . 84
9.25.4 Friends And Related Function Documentation . . . 84

9.25.4.1 operator<< . . . 84
9.25.5 Member Data Documentation . . . 84
9.25.5.1 m_data . . . 84
9.26 gldb::TableMismatchedRecordLength Class Reference . . . 85
9.26.1 Detailed Description . . . 85
9.26.2 Constructor & Destructor Documentation . . . 85
9.26.2.1 TableMismatchedRecordLength . . . 86
9.27 gldb::TableNoSuchField Class Reference . . . 86
9.27.1 Detailed Description . . . 87
9.27.2 Constructor & Destructor Documentation . . . 87
9.27.2.1 TableNoSuchField . . . 87
9.28 gldb::TableNoSuchRecord Class Reference . . . 87
9.28.1 Detailed Description . . . 88
9.28.2 Constructor & Destructor Documentation . . . 88
9.28.2.1 TableNoSuchRecord . . . 88
9.29 gldb::TableRow Class Reference . . . 88
9.29.1 Detailed Description . . . 89
9.29.2 Constructor & Destructor Documentation . . . 89
9.29.2.1 TableRow . . . 89
9.29.2.2 TableRow . . . 89
9.29.2.3 TableRow . . . 90
9.29.2.4 TableRow . . . 90
9.29.2.5 TableRow . . . 90
9.29.2.6 TableRow . . . 90
9.29.2.7 TableRow . . . 90
9.29.2.8 ~TableRow . . . 90
9.29.3 Member Function Documentation . . . 90
9.29.3.1 append_field . . . 90
9.29.3.2 append_field . . . 91
9.29.3.3 append_field . . . 91
9.29.3.4 append_field . . . 91
9.29.3.5 append_field . . . 91
9.29.3.6 begin . . . 91
9.29.3.7 begin . . . 91
9.29.3.8 end . . . 92
9.29.3.9 end . . . 92
9.29.3.10 operator= . . . 92
9.29.3.11 operator= . . . 92
9.29.3.12 operator[] . . . 92
9.29.3.13 operator[] . . . 93

9.29.3.14 print . . . 93
9.29.3.15 record_string . . . 93
9.29.3.16 record_string . . . 93
9.29.3.17 size . . . 93
9.29.4 Member Data Documentation . . . 93
9.29.4.1 m_fields . . . 93

**10 File Documentation** **95**

10.1 lib/config/config.cpp File Reference . . . 95
10.1.1 Detailed Description . . . 95
10.2 lib/config/config.h File Reference . . . 96
10.2.1 Detailed Description . . . 97
10.3 lib/config/config_getopt.cpp File Reference . . . 97
10.3.1 Detailed Description . . . 97
10.3.2 Macro Definition Documentation . . . 98
10.3.2.1 _XOPEN_SOURCE . . . 98
10.4 lib/database/data_structures.h File Reference . . . 98
10.4.1 Detailed Description . . . 99
10.5 lib/database/database.h File Reference . . . 99
10.5.1 Detailed Description . . . 100
10.6 lib/database/dbconn.cpp File Reference . . . 101
10.6.1 Detailed Description . . . 101
10.7 lib/database/dbconn.h File Reference . . . 102
10.7.1 Detailed Description . . . 103
10.8 lib/database/dbconnimp.h File Reference . . . 103
10.8.1 Detailed Description . . . 105
10.9 lib/database/table.cpp File Reference . . . 105
10.9.1 Detailed Description . . . 105
10.10 lib/database/table.h File Reference . . . 106
10.10.1 Detailed Description . . . 107
10.11 lib/database/tablefield.cpp File Reference . . . 107
10.11.1 Detailed Description . . . 108
10.12 lib/database/tablefield.h File Reference . . . 108
10.12.1 Detailed Description . . . 109
10.13 lib/database/tablerow.cpp File Reference . . . 109
10.13.1 Detailed Description . . . 110
10.14 lib/database/tablerow.h File Reference . . . 110
10.14.1 Detailed Description . . . 111
10.15 lib/database_imp/database_imp.h File Reference . . . 112
10.15.1 Detailed Description . . . 113

10.16lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference . . . . . 114
10.16.1 Detailed Description . . . . . 114
10.17lib/database_imp/dummy/dbconn_dummy_imp.h File Reference . . . . . 115
10.17.1 Detailed Description . . . . . 116
10.18lib/database_imp/mysql/dbconn_mysql_functions.cpp File Reference . . . . . 117
10.18.1 Detailed Description . . . . . 117
10.19lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference . . . . . 118
10.19.1 Detailed Description . . . . . 118
10.20lib/database_imp/mysql/dbconn_mysql_imp.h File Reference . . . . . 119
10.20.1 Detailed Description . . . . . 120
10.21lib/database_imp/mysql/dbconn_mysql_result.cpp File Reference . . . . . 121
10.21.1 Detailed Description . . . . . 121
10.22lib/database_imp/mysql/dbconn_mysql_result.h File Reference . . . . . 122
10.22.1 Detailed Description . . . . . 122
10.23lib/dbsql/dbsql.h File Reference . . . . . 123
10.23.1 Detailed Description . . . . . 124
10.24lib/dbsql/dbsql_dummy.h File Reference . . . . . 124
10.24.1 Detailed Description . . . . . 125
10.25lib/dbsql/dbsql_implementations.h File Reference . . . . . 126
10.25.1 Detailed Description . . . . . 126
10.26lib/dbsql/dbsql_mysql.h File Reference . . . . . 127
10.26.1 Detailed Description . . . . . 128
10.27lib/dbsql/dbsqlstatements.cpp File Reference . . . . . 128
10.27.1 Detailed Description . . . . . 128
10.28lib/dbsql/dbsqlstatements.h File Reference . . . . . 129
10.28.1 Detailed Description . . . . . 130
10.29lib/gldb/gldatabase.cpp File Reference . . . . . 130
10.29.1 Detailed Description . . . . . 131
10.29.2 Function Documentation . . . . . 131
10.29.2.1 boolstring_to_bool . . . . . 131
10.30lib/gldb/gldatabase.h File Reference . . . . . 131
10.30.1 Detailed Description . . . . . 133
10.31lib/gldb/gldb.h File Reference . . . . . 133
10.31.1 Detailed Description . . . . . 134
10.32lib/gldb/glexception.h File Reference . . . . . 134
10.32.1 Detailed Description . . . . . 135
10.33lib/gldb/gluser.cpp File Reference . . . . . 136
10.33.1 Detailed Description . . . . . 136
10.34lib/gldb/gluser.h File Reference . . . . . 136
10.34.1 Detailed Description . . . . . 137

10.35lib/gldb/gluser_pass.cpp File Reference . . . . . 137
10.35.1 Detailed Description . . . . . 138
10.35.2 Macro Definition Documentation . . . . . 138
10.35.2.1 _XOPEN_SOURCE . . . . . 138
10.35.3 Function Documentation . . . . . 139
10.35.3.1 generate_salt . . . . . 139
10.36lib/pgutils/pgutils.h File Reference . . . . . 139
10.36.1 Detailed Description . . . . . 139
10.37lib/pgutils/stringhelp.cpp File Reference . . . . . 140
10.37.1 Detailed Description . . . . . 140
10.38lib/pgutils/stringhelp.h File Reference . . . . . 140
10.38.1 Detailed Description . . . . . 142
10.39progs/gl_db/gl_db_main.cpp File Reference . . . . . 142
10.39.1 Detailed Description . . . . . 143
10.40progs/gl_report/gl_report_main.cpp File Reference . . . . . 143
10.40.1 Detailed Description . . . . . 145
10.41progs/gl_user/gl_user_main.cpp File Reference . . . . . 145
10.41.1 Detailed Description . . . . . 146

# Chapter 1

# General Ledger.

General Ledger will be a fully-featured, multi-user, open-source general ledger system. The project is in the early stages of development.

# Chapter 2

# Todo List

**File gluser_pass.cpp**

Implement a better form of password encryption. In particular, these functions are not re-entrant, and only use the first 8 characters of the password.

# Chapter 3

# Bug List

**Member gldb::Table::Table (const Table &table)**

‘explicit‘ removed from here after failure to compile at end of MySQL query function.

# Chapter 4

# Module Index

## 4.1 Modules

Here is a list of all modules:


Program configuration module . . . . . 15
Database interaction module . . . . . 16
SQL statements module . . . . . 19
General Ledger database module. . . . . 20
General purpose utilities. . . . . 21
Database program. . . . . 25
Reporting program. . . . . 27
User administration program. . . . . 29

# Chapter 5

# Class Index

## 5.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:


genleg::Config . . . 33
genleg::ConfigException . . . 39
  genleg::ConfigBadConfigFile . . . 35
  genleg::ConfigBadOption . . . 37
  genleg::ConfigCouldNotOpenFile . . . 38
  genleg::ConfigOptionNotSet . . . 40
gldb::DBConn . . . 41
gldb::DBConnException . . . 48
  gldb::DBConnCouldNotConnect . . . 43
  gldb::DBConnCouldNotQuery . . . 44
gldb::DBConnImp . . . 49
  gldb::DBConnDummy . . . 46
  gldb::DBConnMySQL . . . 50
genleg::DBSQLStatements . . . 55
  genleg::DBSQLDummy . . . 53
  genleg::DBSQLMySQL . . . 54
genleg::GLDatabase . . . 58
genleg::GLDBException . . . 63
genleg::GLUser . . . 64
gldb::MySQLResult . . . 69
gldb::Table . . . 70
gldb::TableException . . . 78
  gldb::TableBadInputFile . . . 76
  gldb::TableCouldNotOpenInputFile . . . 77
  gldb::TableMismatchedRecordLength . . . 85
  gldb::TableNoSuchField . . . 86
  gldb::TableNoSuchRecord . . . 87
gldb::TableField . . . 79
gldb::TableRow . . . 88

# Chapter 6

# Class Index

## 6.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:


genleg::Config
Configuration options class . . . 33
genleg::ConfigBadConfigFile
Exception class for badly formed configuration file . . . 35
genleg::ConfigBadOption
Exception class for bad provided option . . . 37
genleg::ConfigCouldNotOpenFile
Exception class for when conf file cannot be opened . . . 38
genleg::ConfigException
Configuration module exception base class . . . 39
genleg::ConfigOptionNotSet
Exception class for option not set . . . 40
gldb::DBConn
Database connection class . . . 41
gldb::DBConnCouldNotConnect
Could not connect to database exception class . . . 43
gldb::DBConnCouldNotQuery
Could not execute database query exception class . . . 44
gldb::DBConnDummy
Dummy database implementation class . . . 46
gldb::DBConnException
Base database connection exception class . . . 48
gldb::DBConnImp
Abstract database implementation base class . . . 49
gldb::DBConnMySQL
MySQL database implementation class . . . 50
genleg::DBSQLDummy
Dummy SQL statements class . . . 53
genleg::DBSQLMySQL
MySQL SQL statements class . . . 54
genleg::DBSQLStatements
SQL statements class . . . 55
genleg::GLDatabase
General ledger database class . . . 58
genleg::GLDBException
Base general ledger database exceptionc class . . . 63
genleg::GLUser
General ledger user class . . . 64

gldb::MySQLResult
MySQL result structure class . . . . . 69
gldb::Table
Database table class . . . . . 70
gldb::TableBadInputFile
Could not connect to database exception class . . . . . 76
gldb::TableCouldNotOpenInputFile
Could not connect to database exception class . . . . . 77
gldb::TableException
Base database connection exception class . . . . . 78
gldb::TableField
Database table field class . . . . . 79
gldb::TableMismatchedRecordLength
Mismatched record length exception class . . . . . 85
gldb::TableNoSuchField
No such field exception class . . . . . 86
gldb::TableNoSuchRecord
No such record exception class . . . . . 87
gldb::TableRow
Database table row class . . . . . 88

# Chapter 7

# File Index

## 7.1 File List

Here is a list of all documented files with brief descriptions:


lib/config/config.cpp
Implementation of program configurations class . . . 95
lib/config/config.h
Interface to program configurations class . . . 96
lib/config/config_getopt.cpp
Implementation of command line functionality . . . 97
lib/database/data_structures.h
Main interface to database data structures . . . 98
lib/database/database.h
User interface to database functionality . . . 99
lib/database/dbconn.cpp
Implementation of database connection class . . . 101
lib/database/dbconn.h
Interface to database connection base class . . . 102
lib/database/dbconnimp.h
Interface to abstract database implementation base class . . . 103
lib/database/table.cpp
Implementation of database table data structure . . . 105
lib/database/table.h
Interface to database table data structure . . . 106
lib/database/tablefield.cpp
Implementation of database table field class . . . 107
lib/database/tablefield.h
Interface to database table field class . . . 108
lib/database/tablerow.cpp
Implementation of database table row data structure . . . 109
lib/database/tablerow.h
Interface to database table row data structure . . . 110
lib/database_imp/database_imp.h
Interface to database implementation factory function . . . 112
lib/database_imp/dummy/dbconn_dummy_imp.cpp
Implementation of Dummy database connection implementation class . . . 114
lib/database_imp/dummy/dbconn_dummy_imp.h
Interface to dummy database connection implementation class . . . 115
lib/database_imp/mysql/dbconn_mysql_functions.cpp
Implementation of MySQL implementation factory function . . . 117
lib/database_imp/mysql/dbconn_mysql_imp.cpp
Implementation of MySQL database connection implementation class . . . 118

lib/database_imp/mysql/dbconn_mysql_imp.h
Interface to MySQL database connection implementation class . . . 119
lib/database_imp/mysql/dbconn_mysql_result.cpp
Implementation of MySQL result structure resource handle class . . . 121
lib/database_imp/mysql/dbconn_mysql_result.h
Interface to MySQL result structure resource handle class . . . 122
lib/dbsql/dbsql.h
User interface to DBSQL module . . . 123
lib/dbsql/dbsql_dummy.h
Interface to dummy SQL statement class . . . 124
lib/dbsql/**dbsql_functions.h** . . . **??**
lib/dbsql/dbsql_implementations.h
Aggregation header for DBSqlStatements implementations . . . 126
lib/dbsql/dbsql_mysql.h
Interface to MySQL SQL statement class . . . 127
lib/dbsql/dbsqlstatements.cpp
Implementation of SQL statement class . . . 128
lib/dbsql/dbsqlstatements.h
Implementation of SQL module standalone functions . . . 129
lib/gldb/gldatabase.cpp
Implementation of General Ledger database class . . . 130
lib/gldb/gldatabase.h
Interface to General Ledger database class . . . 131
lib/gldb/gldb.h
User interface to General Ledger database module . . . 133
lib/gldb/glexception.h
Interface to General Ledger base exception class . . . 134
lib/gldb/gluser.cpp
Implementation of user class . . . 136
lib/gldb/gluser.h
Interface to user class . . . 136
lib/gldb/gluser_pass.cpp
Implementation of password functions for user class . . . 137
lib/pgutils/pgutils.h
Aggregate interface to general utility functions . . . 139
lib/pgutils/stringhelp.cpp
Implementation of string helper functions . . . 140
lib/pgutils/stringhelp.h
Interface to string helper functions . . . 140
progs/gl_db/gl_db_main.cpp
Main functionality for gl_db program . . . 142
progs/gl_report/gl_report_main.cpp
Main functionality for gl_report program . . . 143
progs/gl_user/gl_user_main.cpp
Main functionality for gl_user program . . . 145

# Chapter 8

# Module Documentation

## 8.1 Program configuration module

### Classes

- class genleg::ConfigException

  *Configuration module exception base class.*
- class genleg::ConfigOptionNotSet

  *Exception class for option not set.*
- class genleg::ConfigBadOption

  *Exception class for bad provided option.*
- class genleg::ConfigCouldNotOpenFile

  *Exception class for when conf file cannot be opened.*
- class genleg::ConfigBadConfigFile

  *Exception class for badly formed configuration file.*
- class genleg::Config

  *Configuration options class.*

### Enumerations

- enum genleg::Argument

  *Enumeration class for option argument specifications.*

### 8.1.1 Detailed Description

Module for getting options from the command line and configuration files.

## 8.2 Database interaction module

### Classes

- class gldb::DBConnException

  *Base database connection exception class.*
- class gldb::DBConnCouldNotConnect

  *Could not connect to database exception class.*
- class gldb::DBConnCouldNotQuery

  *Could not execute database query exception class.*
- class gldb::DBConn

  *Database connection class.*
- class gldb::DBConnImp

  *Abstract database implementation base class.*
- class gldb::TableException

  *Base database connection exception class.*
- class gldb::TableNoSuchField

  *No such field exception class.*
- class gldb::TableNoSuchRecord

  *No such record exception class.*
- class gldb::TableMismatchedRecordLength

  *Mismatched record length exception class.*
- class gldb::TableBadInputFile

  *Could not connect to database exception class.*
- class gldb::TableCouldNotOpenInputFile

  *Could not connect to database exception class.*
- class gldb::Table

  *Database table class.*
- class gldb::TableField

  *Database table field class.*
- class gldb::TableRow

  *Database table row class.*
- class gldb::DBConnDummy

  *Dummy database implementation class.*
- class gldb::DBConnMySQL

  *MySQL database implementation class.*
- class gldb::MySQLResult

  *MySQL result structure class.*

### Functions

- DBConnImp * gldb::get_connection (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)

  *Creates and returns a pointer to a database implementation.*
- std::string gldb::get_database_type ()

  *Returns the name of the compiled-in database type.*
- static TableRow get_field_names (MySQLResult &result)

  *Gets field names from a MySQL result structure.*
- static TableRow get_row (MySQLResult &result, MYSQL_ROW row)

  *Creates a TableRow from a MySQL result row.*

### 8.2.1 Detailed Description

Module for interacting with the database.

### 8.2.2 Function Documentation

#### 8.2.2.1 DBConnImp * gldb::get_connection ( const std::string & *database,* const std::string & *hostname,* const std::string & *username,* const std::string & *password* )

Creates and returns a pointer to a database implementation.

The implementation of this function is provided by the individual database implementations. One database implementation is compiled into the program at any one time. Multiple database systems are, or will be, supported, and not every system will possess the libraries and headers to compile every implementation. Therefore, only only implementation is compiled in at a time. The fact that each database implementation will implement this function to return the correct derived class prevents any attempt to compile unsupported library code. This would not be feasible if we were to simply provide each implementation as a subclass.

**Parameters**

| | |
|---|---|
| *database* | The name of the database to which to connect. |
| *hostname* | The hostname of the computer running the database. |
| *username* | The username with which to log into the database. |
| *password* | The password with which to log into the database. |

**Returns**

A pointer to the database implementation.

#### 8.2.2.2 std::string gldb::get_database_type ( )

Returns the name of the compiled-in database type.

**Returns**

The name of the compiled-in database type.

#### 8.2.2.3 static TableRow get_field_names ( MySQLResult & *result* ) [static]

Gets field names from a MySQL result structure.

**Parameters**

| | |
|---|---|
| *result* | The MySQL result structure. |

**Returns**

A TableRow containing the field names.

#### 8.2.2.4 static TableRow get_row ( MySQLResult & *result,* MYSQL_ROW *row* ) [static]

Creates a TableRow from a MySQL result row.

**Parameters**

| | |
|---|---|
| *result* | The MySQL result structure. |
| *row* | The MySQL row structure. |

**Returns**

A TableRow containing the row data.

## 8.3 SQL statements module

### Classes

- class genleg::DBSQLDummy

  *Dummy SQL statements class.*
- class genleg::DBSQLMySQL

  *MySQL SQL statements class.*
- class genleg::DBSQLStatements

  *SQL statements class.*

### 8.3.1 Detailed Description

Module for producing SQL statements used by program.

## 8.4 General Ledger database module.

### Classes

- class genleg::GLDatabase

  *General ledger database class.*
- class genleg::GLDBException

  *Base general ledger database exceptionc class.*
- class genleg::GLUser

  *General ledger user class.*

### 8.4.1 Detailed Description

Module for interacting with the general ledger database model.

## 8.5 General purpose utilities.

### Functions

- std::string & pgutils::trim_front (std::string &s)

  *Trims leading whitespace from a string.*
- std::string & pgutils::trim_back (std::string &s)

  *Trims trailing whitespace from a string.*
- std::string & pgutils::trim (std::string &s)

  *Trims leading and trailing whitespace from a string.*
- std::vector< std::string > pgutils::split (const std::string &s, const char delim)

  *Splits a delimited string into tokens.*
- std::vector< std::string > & pgutils::split (std::vector< std::string > &vec, const std::string &s, const char delim)

  *Splits a delimited string into tokens.*
- bool pgutils::next_content_line (std::istream &ifs, std::string &s)

  *Gets the next content line from a stream.*
- std::vector< std::string > & pgutils::content_lines (std::vector< std::string > &vec, std::istream &ifs)

  *Populates a vector of content lines from a stream.*
- std::vector< std::vector < std::string > > & pgutils::split_lines (std::vector< std::vector< std::string >> &vec, std::istream &ifs, const char delim)

  *Populates a vector of vectors of fields from a stream.*
- std::string & pgutils::join (const std::vector< std::string > &vec, std::string &s, const char delim)

  *Joins a vector of strings into a delimited line.*
- bool pgutils::replace (std::string &str, const std::string &from, const std::string &to)

  *Replaces a substring with another string.*

### 8.5.1 Detailed Description

General purpose utility classes and functions.

### 8.5.2 Function Documentation

#### 8.5.2.1 std::vector< std::string > & pgutils::content_lines ( std::vector< std::string > & *vec,* std::istream & *ifs* )

Populates a vector of content lines from a stream.

**Parameters**

| | |
|---|---|
| *vec* | The vector to populate. |
| *ifs* | The input stream. |

**Returns**

A reference to `vec`.

#### 8.5.2.2 std::string & pgutils::join ( const std::vector< std::string > & *vec,* std::string & *s,* const char *delim* )

Joins a vector of strings into a delimited line.

The function is the opposite of `split`.

**Parameters**

| | |
|---|---|
| *vec* | The vector containing the strings. |
| *s* | The string in which to store the line. |
| *delim* | The delimiter character to be used to delimit the fields. |

**Returns**

A reference to `s`.

#### 8.5.2.3 bool pgutils::next_content_line ( std::istream & *ifs,* std::string & *s* )

Gets the next content line from a stream.

A "content line" is defined as a line which, after being trimmed of trailing and leading whitespace, is not empty, and does not start with a '#' character (indicating a comment line).

**Parameters**

| | |
|---|---|
| *ifs* | The input stream. |
| *s* | The string in which to store the line. |

**Returns**

`true` if there is a next content line, `false` otherwise.

#### 8.5.2.4 bool pgutils::replace ( std::string & *str,* const std::string & *from,* const std::string & *to* )

Replaces a substring with another string.

**Parameters**

| | |
|---|---|
| *str* | The string containing the substring to replace. |
| *from* | The substring to replace. |
| *to* | The string with which to replace the substring. |

**Returns**

`true` if a replacement was made, `false` otherwise.

#### 8.5.2.5 std::vector< std::string > pgutils::split ( const std::string & *s,* const char *delim* )

Splits a delimited string into tokens.

**Parameters**

| | |
|---|---|
| *s* | The string to split. |
| *delim* | The delimiter character on which to split. |

**Returns**

A vector of tokens.

#### 8.5.2.6 std::vector< std::string > & pgutils::split ( std::vector< std::string > & *vec,* const std::string & *s,* const char *delim* )

Splits a delimited string into tokens.

**Parameters**

| | |
|---|---|
| *vec* | The vector into which to add the tokens. |
| *s* | The string to split. |
| *delim* | The delimiter character on which to split. |

**Returns**

A reference to `vec`.

#### 8.5.2.7 std::vector< std::vector< std::string > > & pgutils::split_lines ( std::vector< std::vector< std::string >> & *vec,* std::istream & *ifs,* const char *delim* )

Populates a vector of vectors of fields from a stream.

**Parameters**

| | |
|---|---|
| *vec* | The vector to populate. |
| *ifs* | The input stream. |
| *delim* | The delimiter character to split each content line. |

**Returns**

A reference to `vec`.

#### 8.5.2.8 std::string & pgutils::trim ( std::string & *s* )

Trims leading and trailing whitespace from a string.

**Parameters**

| | |
|---|---|
| *s* | The string to trim. |

**Returns**

The trimmed string.

#### 8.5.2.9 std::string & pgutils::trim_back ( std::string & *s* )

Trims trailing whitespace from a string.

**Parameters**

| | |
|---|---|
| *s* | The string to trim. |

**Returns**

The trimmed string.

#### 8.5.2.10 std::string & pgutils::trim_front ( std::string & *s* )

Trims leading whitespace from a string.

**Parameters**

| | |
|---|---|
| *s* | The string to trim. |

**Returns**

The trimmed string.

## 8.6 Database program.

### Functions

- static void set_configuration (Config &config, int argc, char ∗argv[])
  *Sets program configuration options.*
- static bool check_help_and_version (const Config &config)
  *Prints help or version messages if requested.*
- static bool check_db_parameters (const Config &config)
  *Checks if database, hostname and username were provided.*
- static void print_usage_message ()
  *Prints a program usage message.*
- static void print_version_message ()
  *Prints a program version message.*
- static void print_help_message ()
  *Prints a program help message.*
- static std::string login (void)
  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])
  *Main function.*

### Variables

- static const char ∗ progname = "gl_db"
  *Static variable for program name.*

### 8.6.1 Detailed Description

Administrative database management program.

### 8.6.2 Function Documentation

#### 8.6.2.1 static bool check_db_parameters ( const Config & *config* ) [static]

Checks if database, hostname and username were provided.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |

**Returns**

`true` if the information was provided, `false` otherwise.

#### 8.6.2.2 static bool check_help_and_version ( const Config & *config* ) [static]

Prints help or version messages if requested.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |

**Returns**

`true` if the help or version message was requested, `false` otherwise.

#### 8.6.2.3 static std::string login ( void ) [static]

Gets a password from the terminal.

**Returns**

The password.

#### 8.6.2.4 int main ( int *argc*, char * *argv[]* )

Main function.

**Parameters**

| | |
|---|---|
| *argc* | Number of command line arguments. |
| *argv* | Command line arguments. |

**Returns**

Exit status code.

#### 8.6.2.5 static void set_configuration ( Config & *config*, int *argc*, char * *argv[]* ) [static]

Sets program configuration options.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |
| *argc* | `argc` passed to `main()`. |
| *argv* | `argv` passed to `main()`. |

## 8.7 Reporting program.

### Functions

- static void set_configuration (genleg::Config &config, int argc, char ∗argv[])

  *Sets program configuration options.*
- static void print_usage_message ()

  *Prints a program usage message.*
- static void print_version_message ()

  *Prints a program version message.*
- static void print_help_message ()

  *Prints a program help message.*
- static std::string login (void)

  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])

  *Main function.*

### Variables

- static const char ∗ progname = "gl_report"

  *Static variable for program name.*

### 8.7.1 Detailed Description

Administrative reporting program.

### 8.7.2 Function Documentation

#### 8.7.2.1 static std::string login ( void ) [static]

Gets a password from the terminal.

**Returns**

The password.

#### 8.7.2.2 int main ( int *argc*, char ∗ *argv[]* )

Main function.

**Parameters**

| | |
|---|---|
| *argc* | Number of command line arguments. |
| *argv* | Command line arguments. |

**Returns**

Exit status code.

#### 8.7.2.3 static void set_configuration ( genleg::Config & *config*, int *argc*, char * *argv[]* ) [static]

Sets program configuration options.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |
| *argc* | `argc` passed to `main()`. |
| *argv* | `argv` passed to `main()`. |

## 8.8 User administration program.

### Functions

- static void set_configuration (Config &config, int argc, char ∗argv[])

  *Sets program configuration options.*
- static bool check_help_and_version (const Config &config)

  *Prints help or version messages if requested.*
- static bool check_db_parameters (const Config &config)

  *Checks if database, hostname and username were provided.*
- GLUser get_user (Config &config, GLDatabase &gdb)

  *Returns a user from either an ID or a name.*
- static void show_user_details (const GLUser &user)

  *Outputs details for a user.*
- static void enable_user (GLUser &user, Config &config, GLDatabase &gdb)

  *Enables or disables a user.*
- static void set_user_password (GLUser &user, Config &config, GLDatabase &gdb)

  *Sets a user's password.*
- static void check_user_password (GLUser &user, Config &config)

  *Checks a user's password.*
- static void print_usage_message ()

  *Prints a program usage message.*
- static void print_version_message ()

  *Prints a program version message.*
- static void print_help_message ()

  *Prints a program help message.*
- static std::string login (void)

  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])

  *Main function.*

### Variables

- static const char ∗ progname = "gl_user"

  *Static variable for program name.*

### 8.8.1 Detailed Description

User administration program.

### 8.8.2 Function Documentation

#### 8.8.2.1 static bool check_db_parameters ( const Config & *config* ) [static]

Checks if database, hostname and username were provided.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |

**Returns**

`true` if the information was provided, `false` otherwise.

#### 8.8.2.2 static bool check_help_and_version ( const Config & *config* ) [static]

Prints help or version messages if requested.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |

**Returns**

`true` if the help or version message was requested, `false` otherwise.

#### 8.8.2.3 static void check_user_password ( GLUser & *user,* Config & *config* ) [static]

Checks a user's password.

**Parameters**

| | |
|---|---|
| *user* | Reference to user. |
| *config* | Reference to program configuration options. |

#### 8.8.2.4 static void enable_user ( GLUser & *user,* Config & *config,* GLDatabase & *gdb* ) [static]

Enables or disables a user.

**Parameters**

| | |
|---|---|
| *user* | Reference to user. |
| *config* | Reference to program configuration. |
| *gdb* | Reference to database object. |

#### 8.8.2.5 GLUser get_user ( Config & *config,* GLDatabase & *gdb* )

Returns a user from either an ID or a name.

**Parameters**

| | |
|---|---|
| *config* | Program configurations object. |
| *gdb* | Database object. |

**Returns**

The user.

#### 8.8.2.6 static std::string login ( void ) [static]

Gets a password from the terminal.

**Returns**

The password.

#### 8.8.2.7 int main ( int *argc*, char * *argv[]* )

Main function.

**Parameters**

| | |
|---|---|
| *argc* | Number of command line arguments. |
| *argv* | Command line arguments. |

**Returns**

Exit status code.

#### 8.8.2.8 static void set_configuration ( Config & *config*, int *argc*, char * *argv[]* ) [static]

Sets program configuration options.

**Parameters**

| | |
|---|---|
| *config* | Reference to a Config object. |
| *argc* | `argc` passed to `main()`. |
| *argv* | `argv` passed to `main()`. |

#### 8.8.2.9 static void set_user_password ( GLUser & *user*, Config & *config*, GLDatabase & *gdb* ) [static]

Sets a user's password.

**Parameters**

| | |
|---|---|
| *user* | Reference to user. |
| *config* | Reference to program configuration. |
| *gdb* | Reference to database object. |

#### 8.8.2.10 static void show_user_details ( const GLUser & *user* ) [static]

Outputs details for a user.

**Parameters**

| | |
|---|---|
| *user* | Reference to user. |

# Chapter 9

# Class Documentation

## 9.1 genleg::Config Class Reference

Configuration options class.

`#include <config.h>`

### Public Member Functions

- Config ()
- ~Config ()
- void add_cmdline_option (const std::string option, const enum Argument arg)

  *Adds a supported command line option.*
- void populate_from_cmdline (const int argc, char *const *argv)

  *Populates options from the command line.*
- void populate_from_file (const std::string filename)

  *Populates options from a configuration file.*
- bool is_set (const std::string option) const

  *Checks is an option is set.*
- const std::string & operator[] (const std::string &option) const

  *operator[] overload.*

### Private Attributes

- std::map< std::string, std::string > m_opts_set
- std::list< std::pair < std::string, enum Argument > > m_opts_supp

### 9.1.1 Detailed Description

Configuration options class.

### 9.1.2 Constructor & Destructor Documentation

#### 9.1.2.1 Config::Config ( )

Constructor

#### 9.1.2.2 Config::~Config ( )

Destructor

### 9.1.3 Member Function Documentation

#### 9.1.3.1 void Config::add_cmdline_option ( const std::string *option*, const enum Argument *arg* )

Adds a supported command line option.

**Parameters**

| | |
|---|---|
| *option* | The name of the option. |
| *arg* | The argument specification for the option. |

#### 9.1.3.2 bool Config::is_set ( const std::string *option* ) const

Checks is an option is set.

**Parameters**

| | |
|---|---|
| *option* | The name of the option to check. |

**Returns**

`true` if the option has been set, `false` if it has not.

#### 9.1.3.3 const std::string & Config::operator[] ( const std::string & *option* ) const

operator[] overload.

Retrieves the value of a set option.

**Parameters**

| | |
|---|---|
| *option* | The name of the option. |

**Returns**

The value of the option.

**Exceptions**

| | |
|---|---|
| *ConfigOptionNotSet* | If the named option has not been set. |

#### 9.1.3.4 void Config::populate_from_cmdline ( const int *argc*, char *const * *argv* )

Populates options from the command line.

**Parameters**

| | |
|---|---|
| *argc* | `argc` supplied to `main()`. |
| *argv* | `argv` supplied to `main()`. |

**Exceptions**

| | |
|---|---|
| *ConfigBadOption* | If an unsupported option is specified, or if a required argument is missing, or if an unexpected argument is found. |

#### 9.1.3.5 void Config::populate_from_file ( const std::string *filename* )

Populates options from a configuration file.

**Parameters**

| | |
|---|---|
| *filename* | The name of the configuration file. |

**Exceptions**

| | |
|---|---|
| *ConfigCouldNotOpenFile* | If the configuration file cannot be opened. |
| *ConfigBadConfigFile* | If the configuration file is badly formed. |

### 9.1.4 Member Data Documentation

#### 9.1.4.1 std::map<std::string, std::string> genleg::Config::m_opts_set [private]

Map of options which have been set

#### 9.1.4.2 std::list<std::pair<std::string, enum Argument> > genleg::Config::m_opts_supp [private]

List of options which are supported

The documentation for this class was generated from the following files:

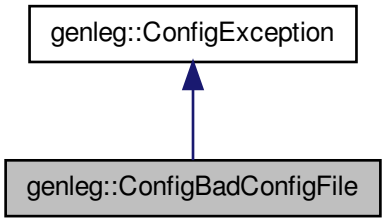
- lib/config/config.h
- lib/config/config.cpp
- lib/config/config_getopt.cpp

## 9.2 genleg::ConfigBadConfigFile Class Reference

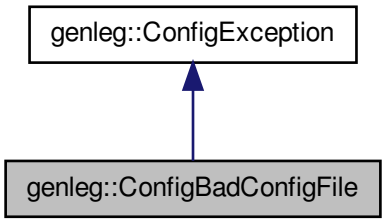
Exception class for badly formed configuration file.

`#include <config.h>`

Inheritance diagram for genleg::ConfigBadConfigFile:




Collaboration diagram for genleg::ConfigBadConfigFile:




## Public Member Functions

- ConfigBadConfigFile (const std::string &msg)
  *Constructor.*

### 9.2.1 Detailed Description

Exception class for badly formed configuration file.

### 9.2.2 Constructor & Destructor Documentation

#### 9.2.2.1 genleg::ConfigBadConfigFile::ConfigBadConfigFile ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

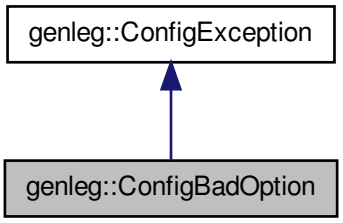
- lib/config/config.h

## 9.3 genleg::ConfigBadOption Class Reference

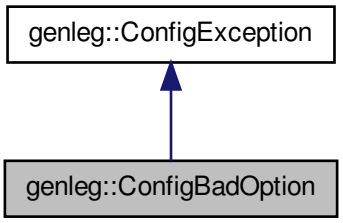
Exception class for bad provided option.

`#include <config.h>`

Inheritance diagram for genleg::ConfigBadOption:




Collaboration diagram for genleg::ConfigBadOption:




### Public Member Functions

- ConfigBadOption (const std::string &msg)

  *Constructor.*

### 9.3.1 Detailed Description

Exception class for bad provided option.

### 9.3.2 Constructor & Destructor Documentation

#### 9.3.2.1 genleg::ConfigBadOption::ConfigBadOption ( const std::string & *msg* ) [inline],[explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

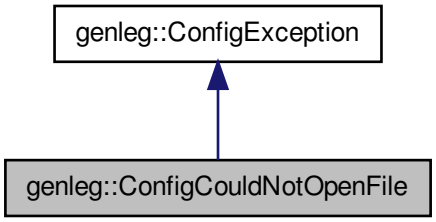
- lib/config/config.h

## 9.4 genleg::ConfigCouldNotOpenFile Class Reference

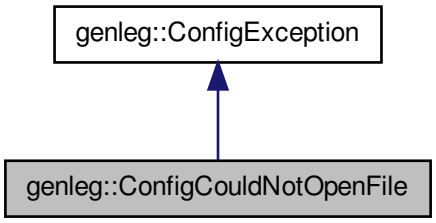
Exception class for when conf file cannot be opened.

`#include <config.h>`

Inheritance diagram for genleg::ConfigCouldNotOpenFile:




Collaboration diagram for genleg::ConfigCouldNotOpenFile:




### Public Member Functions

- ConfigCouldNotOpenFile (const std::string &msg)

  *Constructor.*

### 9.4.1 Detailed Description

Exception class for when conf file cannot be opened.

### 9.4.2 Constructor & Destructor Documentation

#### 9.4.2.1 genleg::ConfigCouldNotOpenFile::ConfigCouldNotOpenFile ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

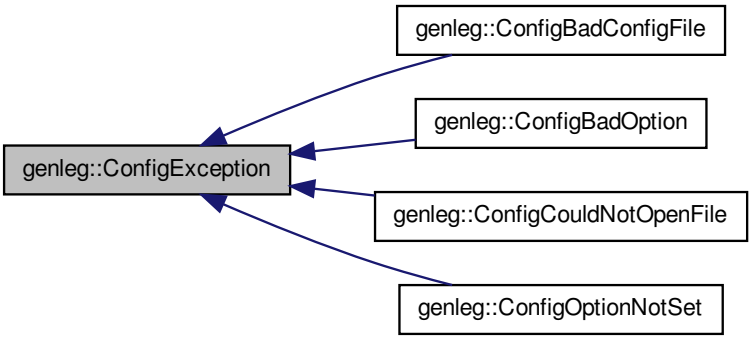
- lib/config/config.h

## 9.5 genleg::ConfigException Class Reference

Configuration module exception base class.

`#include <config.h>`

Inheritance diagram for genleg::ConfigException:




### Public Member Functions

- ConfigException (const std::string &msg)

  *Constructor.*

### 9.5.1 Detailed Description

Configuration module exception base class.

### 9.5.2 Constructor & Destructor Documentation

#### 9.5.2.1 genleg::ConfigException::ConfigException ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

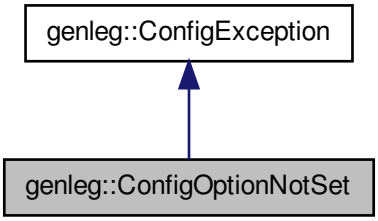
- lib/config/config.h

## 9.6 genleg::ConfigOptionNotSet Class Reference

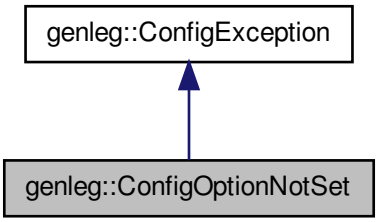
Exception class for option not set.

`#include <config.h>`

Inheritance diagram for genleg::ConfigOptionNotSet:




Collaboration diagram for genleg::ConfigOptionNotSet:




### Public Member Functions

- ConfigOptionNotSet (const std::string &msg)

*Constructor.*

### 9.6.1 Detailed Description

Exception class for option not set.

### 9.6.2 Constructor & Destructor Documentation

#### 9.6.2.1 genleg::ConfigOptionNotSet::ConfigOptionNotSet ( const std::string & *msg* ) [inline],[explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

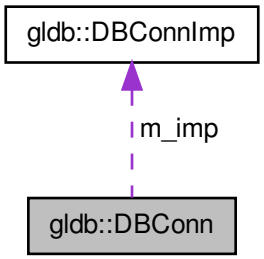
- lib/config/config.h

## 9.7 gldb::DBConn Class Reference

Database connection class.

`#include <dbconn.h>`

Collaboration diagram for gldb::DBConn:




### Public Member Functions

- DBConn (DBConnImp *imp)

  *Constructor.*
- ~DBConn ()

  *Destructor..*
- void query (const std::string &sql_query)

  *Runs an SQL query.*
- Table select (const std::string &query)

*Runs an SQL SELECT query.*

- DBConn (const DBConn &)
- DBConn (const DBConn &&)
- DBConn & operator= (const DBConn &)
- DBConn & operator= (const DBConn &&)

### Private Attributes

- DBConnImp * m_imp

## 9.7.1 Detailed Description

Database connection class.

## 9.7.2 Constructor & Destructor Documentation

### 9.7.2.1 DBConn::DBConn ( DBConnImp * *imp* ) [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *imp* | Pointer to database implementation object. |

### 9.7.2.2 gldb::DBConn::DBConn ( const DBConn & )

Deleted copy constructor

### 9.7.2.3 gldb::DBConn::DBConn ( const DBConn && )

Deleted move constructor

## 9.7.3 Member Function Documentation

### 9.7.3.1 DBConn& gldb::DBConn::operator= ( const DBConn & )

Deleted copy assignment operator

### 9.7.3.2 DBConn& gldb::DBConn::operator= ( const DBConn && )

Deleted move assignment operator

### 9.7.3.3 void DBConn::query ( const std::string & *sql_query* )

Runs an SQL query.

**Parameters**

| | |
|---|---|
| *sql_query* | The query. |

**Returns**

A Table object containing the results.

#### 9.7.3.4 Table DBConn::select ( const std::string & *query* )

Runs an SQL SELECT query.

**Parameters**

| | |
|---|---|
| *query* | The query. |

**Returns**

A Table object containing the results.

### 9.7.4 Member Data Documentation

#### 9.7.4.1 DBConnImp* gldb::DBConn::m_imp [private]

Pointer to database implementation object.

The documentation for this class was generated from the following files:

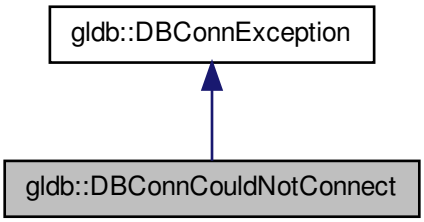
- lib/database/dbconn.h
- lib/database/dbconn.cpp

## 9.8 gldb::DBConnCouldNotConnect Class Reference

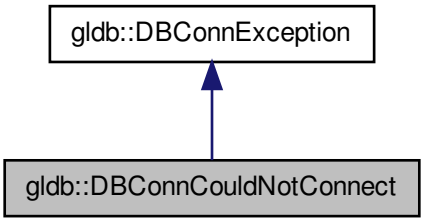
Could not connect to database exception class.

`#include <dbconn.h>`

Inheritance diagram for gldb::DBConnCouldNotConnect:



Collaboration diagram for gldb::DBConnCouldNotConnect:




### Public Member Functions

- DBConnCouldNotConnect (const std::string &msg)

  *Constructor.*

### 9.8.1 Detailed Description

Could not connect to database exception class.

### 9.8.2 Constructor & Destructor Documentation

#### 9.8.2.1 gldb::DBConnCouldNotConnect::DBConnCouldNotConnect ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

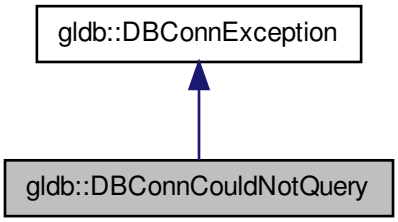
- lib/database/dbconn.h

## 9.9 gldb::DBConnCouldNotQuery Class Reference

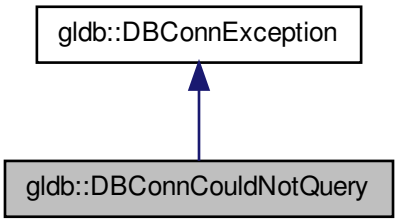
Could not execute database query exception class.

`#include <dbconn.h>`

Inheritance diagram for gldb::DBConnCouldNotQuery:




Collaboration diagram for gldb::DBConnCouldNotQuery:




## Public Member Functions

- DBConnCouldNotQuery (const std::string &msg)
  *Constructor.*

### 9.9.1 Detailed Description

Could not execute database query exception class.

### 9.9.2 Constructor & Destructor Documentation

#### 9.9.2.1 gldb::DBConnCouldNotQuery::DBConnCouldNotQuery ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

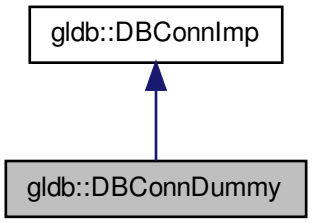
- lib/database/dbconn.h

## 9.10 gldb::DBConnDummy Class Reference

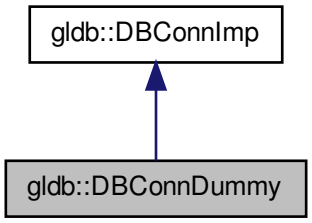
Dummy database implementation class.

`#include <dbconn_dummy_imp.h>`

Inheritance diagram for gldb::DBConnDummy:




Collaboration diagram for gldb::DBConnDummy:




### Public Member Functions

- DBConnDummy (const std::string database, const std::string hostname, const std::string username, const std::string password)

  *Constructor.*
- DBConnDummy (const DBConnDummy &)
- virtual ~DBConnDummy ()
- DBConnDummy & operator= (const DBConnDummy &)
- virtual void query (const std::string &sql_query)

  *Runs an SQL query.*
- Table select (const std::string &query)

  *Fakes running of an SQL SELECT query.*

### 9.10.1 Detailed Description

Dummy database implementation class.

### 9.10.2 Constructor & Destructor Documentation

#### 9.10.2.1 DBConnDummy::DBConnDummy ( const std::string *database,* const std::string *hostname,* const std::string *username,* const std::string *password* )

Constructor.

**Parameters**

| | |
|---|---|
| *database* | The name of the Dummy database. |
| *hostname* | The hostname of the server. |
| *username* | The username to log into the database. |
| *password* | The password to log into the database. |

#### 9.10.2.2 gldb::DBConnDummy::DBConnDummy ( const DBConnDummy & )

Deleted copy constructor

#### 9.10.2.3 DBConnDummy::~DBConnDummy ( ) [virtual]

Destructor

### 9.10.3 Member Function Documentation

#### 9.10.3.1 DBConnDummy& gldb::DBConnDummy::operator= ( const DBConnDummy & )

Deleted assignment operator

#### 9.10.3.2 void DBConnDummy::query ( const std::string & *sql_query* ) [virtual]

Runs an SQL query.

**Parameters**

| | |
|---|---|
| *sql_query* | The query. |

**Exceptions**

| | |
|---|---|
| *DBConnCouldNotQuery* | If could not successfully execute query. |

Implements gldb::DBConnImp.

#### 9.10.3.3 Table DBConnDummy::select ( const std::string & *query* ) [virtual]

Fakes running of an SQL SELECT query.

**Parameters**

| | |
|---|---|
| *query* | Any query. |

**Returns**

A Table object containing dummy results.

Implements gldb::DBConnImp.

The documentation for this class was generated from the following files:

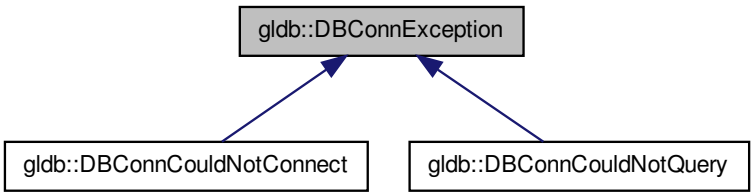
- lib/database_imp/dummy/dbconn_dummy_imp.h
- lib/database_imp/dummy/dbconn_dummy_imp.cpp

## 9.11 gldb::DBConnException Class Reference

Base database connection exception class.

`#include <dbconn.h>`

Inheritance diagram for gldb::DBConnException:




### Public Member Functions

- DBConnException (const std::string &msg)

  *Constructor.*

### 9.11.1 Detailed Description

Base database connection exception class.

### 9.11.2 Constructor & Destructor Documentation

#### 9.11.2.1 gldb::DBConnException::DBConnException ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

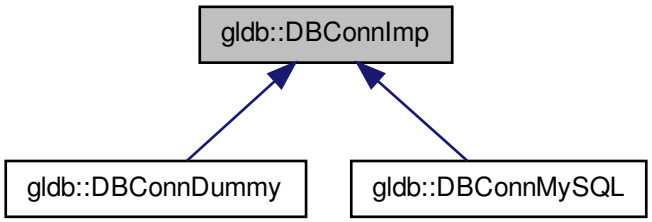
- lib/database/dbconn.h

## 9.12 gldb::DBConnImp Class Reference

Abstract database implementation base class.

`#include <dbconnimp.h>`

Inheritance diagram for gldb::DBConnImp:




### Public Member Functions

- DBConnImp ()
- virtual ~DBConnImp ()
- virtual void query (const std::string &sql_query)=0

  *Runs an SQL query.*
- virtual Table select (const std::string &query)=0

  *Runs an SQL SELECT query.*

### 9.12.1 Detailed Description

Abstract database implementation base class.

### 9.12.2 Constructor & Destructor Documentation

#### 9.12.2.1 gldb::DBConnImp::DBConnImp ( ) [inline]

Constructor

#### 9.12.2.2 virtual gldb::DBConnImp::~DBConnImp ( ) [inline], [virtual]

Destructor

### 9.12.3 Member Function Documentation

#### 9.12.3.1 virtual void gldb::DBConnImp::query ( const std::string & *sql_query* ) [pure virtual]

Runs an SQL query.

**Parameters**

| | |
|---|---|
| *sql_query* | The query. |

Implemented in gldb::DBConnMySQL, and gldb::DBConnDummy.

#### 9.12.3.2 virtual Table gldb::DBConnImp::select ( const std::string & *query* ) [pure virtual]

Runs an SQL SELECT query.

**Parameters**

| | |
|---|---|
| *query* | The query. |

**Returns**

A Table object containing the results.

Implemented in gldb::DBConnMySQL, and gldb::DBConnDummy.

The documentation for this class was generated from the following file:

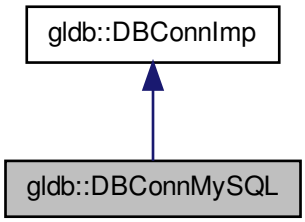
- lib/database/dbconnimp.h

## 9.13 gldb::DBConnMySQL Class Reference

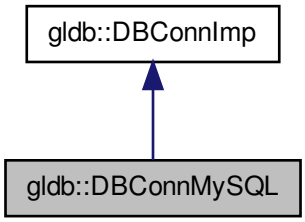
MySQL database implementation class.

`#include <dbconn_mysql_imp.h>`

Inheritance diagram for gldb::DBConnMySQL:



Collaboration diagram for gldb::DBConnMySQL:




## Public Member Functions

- DBConnMySQL (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)

  *Constructor.*
- DBConnMySQL (const DBConnMySQL &)
- DBConnMySQL (const DBConnMySQL &&)
- virtual ~DBConnMySQL ()
- DBConnMySQL & operator= (const DBConnMySQL &)
- DBConnMySQL & operator= (const DBConnMySQL &&)
- virtual void query (const std::string &sql_query)

  *Runs an SQL query.*
- virtual Table select (const std::string &sql_query)

  *Runs an SQL SELECT query.*

## Private Attributes

- MYSQL * m_conn

### 9.13.1 Detailed Description

MySQL database implementation class.

### 9.13.2 Constructor & Destructor Documentation

#### 9.13.2.1 DBConnMySQL::DBConnMySQL ( const std::string & *database*, const std::string & *hostname*, const std::string & *username*, const std::string & *password* )

Constructor.

**Parameters**

| | |
|---|---|
| *database* | The name of the MySQL database. |
| *hostname* | The hostname of the server. |
| *username* | The username to log into the database. |
| *password* | The password to log into the database. |

**Exceptions**

| | |
|---|---|
| *DBConnCouldNotConnect* | If could not connect to database. |

#### 9.13.2.2 gldb::DBConnMySQL::DBConnMySQL ( const DBConnMySQL & )

Deleted copy constructor

#### 9.13.2.3 gldb::DBConnMySQL::DBConnMySQL ( const DBConnMySQL && )

Delete move constructor

#### 9.13.2.4 virtual gldb::DBConnMySQL::~DBConnMySQL ( ) [virtual]

Destructor

### 9.13.3 Member Function Documentation

#### 9.13.3.1 DBConnMySQL& gldb::DBConnMySQL::operator= ( const DBConnMySQL & )

Deleted assignment operator

#### 9.13.3.2 DBConnMySQL& gldb::DBConnMySQL::operator= ( const DBConnMySQL && )

Deleted move assignment operator

#### 9.13.3.3 virtual void gldb::DBConnMySQL::query ( const std::string & *sql_query* ) [virtual]

Runs an SQL query.

**Parameters**

| | |
|---|---|
| *sql_query* | The SQL query. |

**Exceptions**

| | |
|---|---|
| *DBConnCouldNotQuery* | If could not successfully execute query. |

Implements gldb::DBConnImp.

#### 9.13.3.4 virtual Table gldb::DBConnMySQL::select ( const std::string & *sql_query* ) [virtual]

Runs an SQL SELECT query.

**Parameters**

| | |
|---|---|
| *sql_query* | The SQL query. |

**Returns**

A Table object containing the results.

**Exceptions**

| | |
|---|---|
| *DBConnCouldNotQuery* | If could not successfully execute query. |

Implements gldb::DBConnImp.

### 9.13.4 Member Data Documentation

#### 9.13.4.1 MYSQL* gldb::DBConnMySQL::m_conn [private]

The initialized MySQL handle.

The documentation for this class was generated from the following files:

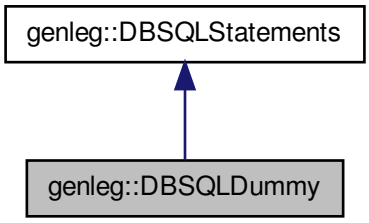
- lib/database_imp/mysql/dbconn_mysql_imp.h
- lib/database_imp/mysql/dbconn_mysql_imp.cpp

## 9.14 genleg::DBSQLDummy Class Reference

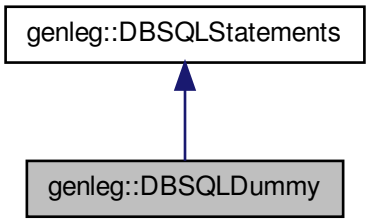
Dummy SQL statements class.

`#include <dbsql_dummy.h>`

Inheritance diagram for genleg::DBSQLDummy:




Collaboration diagram for genleg::DBSQLDummy:



### Additional Inherited Members

#### 9.14.1 Detailed Description

Dummy SQL statements class.

The documentation for this class was generated from the following file:

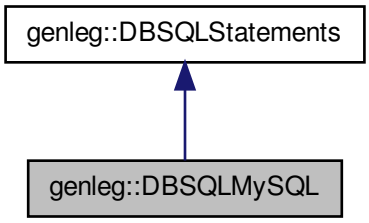
- lib/dbsql/dbsql_dummy.h

## 9.15 genleg::DBSQLMySQL Class Reference

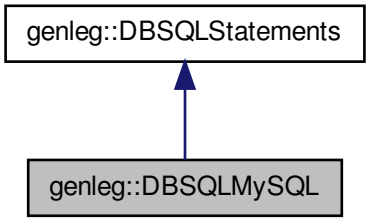
MySQL SQL statements class.

`#include <dbsql_mysql.h>`

Inheritance diagram for genleg::DBSQLMySQL:




Collaboration diagram for genleg::DBSQLMySQL:




### Additional Inherited Members

#### 9.15.1 Detailed Description

MySQL SQL statements class.

The documentation for this class was generated from the following file:

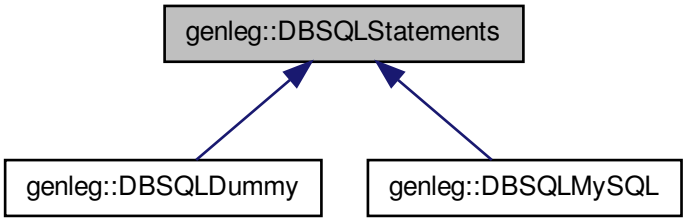
- lib/dbsql/dbsql_mysql.h

## 9.16 genleg::DBSQLStatements Class Reference

SQL statements class.

`#include <dbsqlstatements.h>`

Inheritance diagram for genleg::DBSQLStatements:




### Public Member Functions

- DBSQLStatements ()
- virtual ~DBSQLStatements ()
- virtual std::string create_table (const std::string &table_name) const

  *Returns a SQL statement for creating a table.*
- virtual std::string drop_table (const std::string &table_name) const

  *Returns a SQL statement for dropping a table.*
- virtual std::string create_view (const std::string &view_name) const

  *Returns a SQL statement for creating a view.*
- virtual std::string drop_view (const std::string &view_name) const

  *Returns a SQL statement for dropping a view.*
- virtual std::string user_by_id (const std::string &user_id) const

  *Returns a SQL statement to select a user by ID.*
- virtual std::string user_by_username (const std::string &user_name) const

  *Returns a SQL statement to select a user by username.*
- virtual std::string update_user (const GLUser &user) const

  *Returns a SQL UPDATE statement to update a user.*
- virtual std::string grant (const std::string &user_id, const std::string &perm) const

  *Returns a SQL statement to grant a user a permission.*
- virtual std::string revoke (const std::string &user_id, const std::string &perm) const

  *Returns a SQL UPDATE statement to revoke a permission from a user.*
- virtual std::string get_perms (const std::string &user_id) const

  *Returns a SQL UPDATE statement to list a user's permissions.*

### 9.16.1 Detailed Description

SQL statements class.

### 9.16.2 Constructor & Destructor Documentation

#### 9.16.2.1 DBSQLStatements::DBSQLStatements ( )

Constructor

#### 9.16.2.2 DBSQLStatements::~DBSQLStatements ( ) [virtual]

Destructor

### 9.16.3 Member Function Documentation

#### 9.16.3.1 std::string DBSQLStatements::create_table ( const std::string & *table_name* ) const [virtual]

Returns a SQL statement for creating a table.

**Parameters**

| | |
|---|---|
| *table_name* | The table to create. |

**Returns**

The SQL statement to create the table.

#### 9.16.3.2 std::string DBSQLStatements::create_view ( const std::string & *view_name* ) const [virtual]

Returns a SQL statement for creating a view.

**Parameters**

| | |
|---|---|
| *view_name* | The view to create. |

**Returns**

The SQL statement to create the view.

#### 9.16.3.3 std::string DBSQLStatements::drop_table ( const std::string & *table_name* ) const [virtual]

Returns a SQL statement for dropping a table.

**Parameters**

| | |
|---|---|
| *table_name* | The table to drop. |

**Returns**

The SQL statement to drop the table.

#### 9.16.3.4 std::string DBSQLStatements::drop_view ( const std::string & *view_name* ) const [virtual]

Returns a SQL statement for dropping a view.

**Parameters**

| | |
|---|---|
| *view_name* | The view to drop. |

**Returns**

The SQL statement to drop the view.

#### 9.16.3.5 std::string DBSQLStatements::get_perms ( const std::string & *user_id* ) const [virtual]

Returns a SQL UPDATE statement to list a user's permissions.

**Parameters**

| | |
|---|---|
| *user_id* | The user ID for which to list. |

**Returns**

The SQL statement.

#### 9.16.3.6 std::string DBSQLStatements::grant ( const std::string & *user_id,* const std::string & *perm* ) const [virtual]

Returns a SQL statement to grant a user a permission.

**Attention**

This function always sets the user granting the permission to user 1. This will need to be updated to support the recording of which user has granted the permission, when support for others to be able to do so is implemented.

**Parameters**

| | |
|---|---|
| *user_id* | The user ID for which to grant the permission. |
| *perm* | A string containing the name of the permission. |

**Returns**

The SQL statement.

#### 9.16.3.7 std::string DBSQLStatements::revoke ( const std::string & *user_id,* const std::string & *perm* ) const [virtual]

Returns a SQL UPDATE statement to revoke a permission from a user.

**Parameters**

| | |
|---|---|
| *user_id* | The user ID from which to revoke. |
| *perm* | The permission to revoke. |

**Returns**

The SQL statement.

#### 9.16.3.8 std::string DBSQLStatements::update_user ( const GLUser & *user* ) const [virtual]

Returns a SQL UPDATE statement to update a user.

**Parameters**

| | |
|---|---|
| *user* | A user object. |

**Returns**

The SQL statement.

#### 9.16.3.9 std::string DBSQLStatements::user_by_id ( const std::string & *user_id* ) const [virtual]

Returns a SQL statement to select a user by ID.

**Parameters**

| | |
|---|---|
| *user_id* | The user_id |

**Returns**

The SQL statement.

#### 9.16.3.10 std::string DBSQLStatements::user_by_username ( const std::string & *user_name* ) const [virtual]

Returns a SQL statement to select a user by username.

**Parameters**

| | |
|---|---|
| *user_name* | The username. |

**Returns**

The SQL statement.

The documentation for this class was generated from the following files:

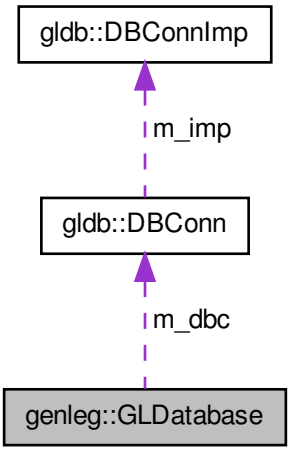
- lib/dbsql/dbsqlstatements.h
- lib/dbsql/dbsqlstatements.cpp

## 9.17 genleg::GLDatabase Class Reference

General ledger database class.

`#include <gldatabase.h>`

Collaboration diagram for genleg::GLDatabase:




## Public Member Functions

- GLDatabase (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)

  *Constructor.*
- ~GLDatabase ()
- void create_structure ()

  *Creates the database structure.*
- void destroy_structure ()

  *Destroys the database structure.*
- void load_sample_data (const std::string &dir)

  *Loads sample data into the database.*
- GLUser get_user_by_id (const std::string &user_id)

  *Returns a user from an ID.*
- GLUser get_user_by_username (const std::string &user_name)

  *Returns a user from a user name.*
- void update_user (const GLUser &user)

  *Updates a user's details.*
- void grant (const GLUser &user, const std::string &perm)

  *Grants a user a permission.*
- void revoke (const GLUser &user, const std::string &perm)

  *Revokes a permission from a user.*

## Static Public Member Functions

- static std::string backend ()

  *Returns the backend database implementation.*

### Private Member Functions

- GLUser create_user (gldb::Table &table)

  *Creates a user from a query table.*

### Private Attributes

- gldb::DBConn m_dbc
- const std::shared_ptr< const DBSQLStatements > m_sql
- const std::vector< std::string > m_tables
- const std::vector< std::string > m_views

## 9.17.1 Detailed Description

General ledger database class.

## 9.17.2 Constructor & Destructor Documentation

### 9.17.2.1 GLDatabase::GLDatabase ( const std::string & *database*, const std::string & *hostname*, const std::string & *username*, const std::string & *password* )

Constructor.

**Parameters**

| | |
|---|---|
| *database* | Database name. |
| *hostname* | Hostname of database machine. |
| *username* | Username to log into database. |
| *password* | Password to log into database. |

**Exceptions**

| | |
|---|---|
| *GLDBException* | on error. |

### 9.17.2.2 GLDatabase::~GLDatabase ( )

Destructor

## 9.17.3 Member Function Documentation

### 9.17.3.1 std::string GLDatabase::backend ( ) [static]

Returns the backend database implementation.

This may be called to discover which database platform support has been compiled into the application.

**Returns**

A string containing the database platform name.

#### 9.17.3.2 void GLDatabase::create_structure ( )

Creates the database structure.

**Exceptions**

| | |
|---|---|
| *GLDBException* | on error. |

#### 9.17.3.3 GLUser GLDatabase::create_user ( gldb::Table & *table* ) [private]

Creates a user from a query table.

Provided because the public functions can get a user either from an ID or a name, this function contains the common functionality.

**Parameters**

| | |
|---|---|
| *table* | A table from the appropriate query. |

**Returns**

The new user.

#### 9.17.3.4 void GLDatabase::destroy_structure ( )

Destroys the database structure.

**Exceptions**

| | |
|---|---|
| *GLDBException* | on error. |

#### 9.17.3.5 GLUser GLDatabase::get_user_by_id ( const std::string & *user_id* )

Returns a user from an ID.

**Parameters**

| | |
|---|---|
| *user_id* | The user ID. |

**Returns**

The user.

**Exceptions**

| | |
|---|---|
| *GLDBException* | if the user cannot be found. |

#### 9.17.3.6 GLUser GLDatabase::get_user_by_username ( const std::string & *user_name* )

Returns a user from a user name.

**Parameters**

| | |
|---|---|
| *user_name* | The user name. |

**Returns**

The user.

**Exceptions**

| | |
|---|---|
| *GLDBException* | if the user cannot be found. |

#### 9.17.3.7 void GLDatabase::grant ( const GLUser & *user,* const std::string & *perm* )

Grants a user a permission.

**Parameters**

| | |
|---|---|
| *user* | The user for which to grant. |
| *perm* | A string containing the permission to grant. |

#### 9.17.3.8 void GLDatabase::load_sample_data ( const std::string & *dir* )

Loads sample data into the database.

**Parameters**

| | |
|---|---|
| *dir* | The directory containing the sample data. Individual files in that directory should be named after the table they are intended to poplate. |

**Exceptions**

| | |
|---|---|
| *GLDBException* | on error. |

#### 9.17.3.9 void GLDatabase::revoke ( const GLUser & *user,* const std::string & *perm* )

Revokes a permission from a user.

**Parameters**

| | |
|---|---|
| *user* | The user for which to revoke. |
| *perm* | A string containing the permission to revoke. |

#### 9.17.3.10 void GLDatabase::update_user ( const GLUser & *user* )

Updates a user's details.

**Parameters**

| | |
|---|---|
| *user* | The user object. |

### 9.17.4 Member Data Documentation

#### 9.17.4.1 gldb::DBConn genleg::GLDatabase::m_dbc [private]

Database connection

#### 9.17.4.2 const std::shared_ptr<const DBSQLStatements> genleg::GLDatabase::m_sql [private]

SQL statements object

#### 9.17.4.3 const std::vector<std::string> genleg::GLDatabase::m_tables [private]

Vector containing database table names

#### 9.17.4.4 const std::vector<std::string> genleg::GLDatabase::m_views [private]

Vector containing database view names

The documentation for this class was generated from the following files:

- lib/gldb/gldatabase.h
- lib/gldb/gldatabase.cpp

## 9.18 genleg::GLDBException Class Reference

Base general ledger database exceptionc class.

`#include <glexception.h>`

### Public Member Functions

- GLDBException (const std::string &msg)

  *Constructor.*

### 9.18.1 Detailed Description

Base general ledger database exceptionc class.

### 9.18.2 Constructor & Destructor Documentation

#### 9.18.2.1 genleg::GLDBException::GLDBException ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

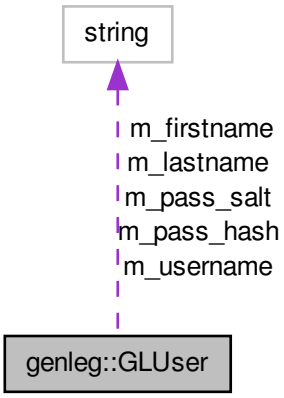
- lib/gldb/glexception.h

## 9.19 genleg::GLUser Class Reference

General ledger user class.

`#include <gluser.h>`

Collaboration diagram for genleg::GLUser:




### Public Member Functions

- GLUser (const std::string &id, const std::string &username, const std::string &firstname, const std::string &lastname, const std::string &pass_hash, const std::string &pass_salt, std::vector< std::string > &&perms, const bool enabled)

  *Constructor.*
- ~GLUser ()
- const std::string & id () const

  *Returns the user ID.*
- const std::string & username () const

  *Returns the username.*
- const std::string & firstname () const

  *Returns the user's first name.*
- const std::string & lastname () const

  *Returns the user's last name.*
- const std::string & pass_hash () const

  *Returns the user's hashed password.*
- const std::string & pass_salt () const

  *Returns the user's password salt.*
- const std::vector< std::string > & permissions () const

  *Returns the permissions for a user.*
- bool enabled () const

  *Returns the user's enabled status.*
- void set_username (const std::string &new_username)

  *Sets a user's username.*
- void set_firstname (const std::string &new_firstname)

*Sets a user's first name.*

- void set_lastname (const std::string &new_lastname)

  *Sets a user's last name.*

- void set_enabled (const bool new_enabled)

  *Sets a user's enabled status.*

- void set_password (const std::string &new_pass)

  *Sets a user's password hash and salt.*

- bool check_password (const std::string &check_pass)

  *Checks a password against the user's hash.*

### Private Attributes

- const std::string m_id
- std::string m_username
- std::string m_firstname
- std::string m_lastname
- std::string m_pass_hash
- std::string m_pass_salt
- const std::vector< std::string > m_perms
- bool m_enabled

## 9.19.1 Detailed Description

General ledger user class.

## 9.19.2 Constructor & Destructor Documentation

### 9.19.2.1 GLUser::GLUser ( const std::string & *id*, const std::string & *username*, const std::string & *firstname*, const std::string & *lastname*, const std::string & *pass_hash*, const std::string & *pass_salt*, std::vector< std::string > && *perms*, const bool *enabled* )

Constructor.

**Parameters**

| | |
|---|---|
| *id* | User ID |
| *username* | Username |
| *firstname* | First name |
| *lastname* | Last name |
| *pass_hash* | The hashed password |
| *pass_salt* | The salt for the hashed password |
| *perms* | Vector of user permissions |
| *enabled* | `true` if user is enabled, `false` otherwise. |

### 9.19.2.2 GLUser::~GLUser ( )

Destructor

## 9.19.3 Member Function Documentation

#### 9.19.3.1 bool GLUser::check_password ( const std::string & *check_pass* )

Checks a password against the user's hash.

**Parameters**

| | |
|---|---|
| *check_pass* | The password to check, must be $>$ 8 characters. |

**Returns**

`true` is the password matches, `false` otherwise.

#### 9.19.3.2 bool GLUser::enabled ( ) const

Returns the user's enabled status.

**Returns**

The user's enabled status.

#### 9.19.3.3 const std::string & GLUser::firstname ( ) const

Returns the user's first name.

**Returns**

The user's first name.

#### 9.19.3.4 const std::string & GLUser::id ( ) const

Returns the user ID.

**Returns**

The user ID.

#### 9.19.3.5 const std::string & GLUser::lastname ( ) const

Returns the user's last name.

**Returns**

The user's last name.

#### 9.19.3.6 const std::string & GLUser::pass_hash ( ) const

Returns the user's hashed password.

**Returns**

The user's hashed password.

#### 9.19.3.7 const std::string & GLUser::pass_salt ( ) const

Returns the user's password salt.

**Returns**

The user's password salt.

#### 9.19.3.8 const std::vector< std::string > & GLUser::permissions ( ) const

Returns the permissions for a user.

**Returns**

A vector of strings containing the names of the permissions held by the user.

#### 9.19.3.9 void GLUser::set_enabled ( const bool *new_enabled* )

Sets a user's enabled status.

**Parameters**

| | |
|---|---|
| *new_enabled* | The user's new enabled status. |

#### 9.19.3.10 void GLUser::set_firstname ( const std::string & *new_firstname* )

Sets a user's first name.

**Parameters**

| | |
|---|---|
| *new_firstname* | The user's new first name. |

#### 9.19.3.11 void GLUser::set_lastname ( const std::string & *new_lastname* )

Sets a user's last name.

**Parameters**

| | |
|---|---|
| *new_lastname* | The user's new last name. |

#### 9.19.3.12 void GLUser::set_password ( const std::string & *new_pass* )

Sets a user's password hash and salt.

**Parameters**

| | |
|---|---|
| *new_pass* | The new password, must be > 8 characters. |

#### 9.19.3.13 void GLUser::set_username ( const std::string & *new_username* )

Sets a user's username.

**Parameters**

| | |
|---|---|
| *new_username* | The user's new username. |

#### 9.19.3.14 const std::string & GLUser::username ( ) const

Returns the username.

**Returns**

The username.

### 9.19.4 Member Data Documentation

#### 9.19.4.1 bool genleg::GLUser::m_enabled [private]

User's enabled status

#### 9.19.4.2 std::string genleg::GLUser::m_firstname [private]

User's first name

#### 9.19.4.3 const std::string genleg::GLUser::m_id [private]

User ID

#### 9.19.4.4 std::string genleg::GLUser::m_lastname [private]

User's last name

#### 9.19.4.5 std::string genleg::GLUser::m_pass_hash [private]

User's hashed password

#### 9.19.4.6 std::string genleg::GLUser::m_pass_salt [private]

User's password salt

#### 9.19.4.7 const std::vector<std::string> genleg::GLUser::m_perms [private]

List of permissions

#### 9.19.4.8 std::string genleg::GLUser::m_username [private]

Username

The documentation for this class was generated from the following files:

- lib/gldb/gluser.h
- lib/gldb/gluser.cpp
- lib/gldb/gluser_pass.cpp

## 9.20 gldb::MySQLResult Class Reference

MySQL result structure class.

`#include <dbconn_mysql_result.h>`

### Public Member Functions

- MySQLResult (MYSQL *conn)

  *Constructor.*
- ~MySQLResult ()
- MySQLResult (const MySQLResult &result)
- MySQLResult (MySQLResult &&result)
- MySQLResult & operator= (const MySQLResult &result)
- MySQLResult & operator= (MySQLResult &&result)
- MYSQL_RES * result ()

  *Returns the MYSQL_RES pointer.*
- unsigned int num_fields () const

  *Returns the number of fields in the result set.*

### Private Attributes

- MYSQL_RES * m_result
- unsigned int m_num_fields

### 9.20.1 Detailed Description

MySQL result structure class.

### 9.20.2 Constructor & Destructor Documentation

#### 9.20.2.1 MySQLResult::MySQLResult ( MYSQL * *conn* ) [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *conn* | MySQL connection |

**Exceptions**

| | |
|---|---|
| *DBConnCouldNotQuery* | on failure |

#### 9.20.2.2 gldb::MySQLResult::~MySQLResult ( )

Destructor

#### 9.20.2.3 gldb::MySQLResult::MySQLResult ( const MySQLResult & *result* )

Deleted copy constructor

#### 9.20.2.4 gldb::MySQLResult::MySQLResult ( MySQLResult && *result* )

Deleted move constructor

### 9.20.3 Member Function Documentation

#### 9.20.3.1 unsigned int gldb::MySQLResult::num_fields ( ) const [inline]

Returns the number of fields in the result set.

**Returns**

The number of fields in the result set.

#### 9.20.3.2 MySQLResult& gldb::MySQLResult::operator= ( const MySQLResult & *result* )

Deleted copy assignment operator

#### 9.20.3.3 MySQLResult& gldb::MySQLResult::operator= ( MySQLResult && *result* )

Deleted move assignment operator

#### 9.20.3.4 MYSQL_RES* gldb::MySQLResult::result ( ) [inline]

Returns the MYSQL_RES pointer.

**Returns**

The MYSQL_RES pointer.

### 9.20.4 Member Data Documentation

#### 9.20.4.1 unsigned int gldb::MySQLResult::m_num_fields [private]

The number of fields in the result set

#### 9.20.4.2 MYSQL_RES* gldb::MySQLResult::m_result [private]

The MYSQL_RES pointer

The documentation for this class was generated from the following files:

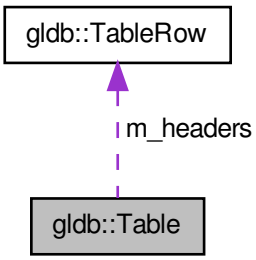
- lib/database_imp/mysql/dbconn_mysql_result.h
- lib/database_imp/mysql/dbconn_mysql_result.cpp

## 9.21 gldb::Table Class Reference

Database table class.

`#include <table.h>`

Collaboration diagram for gldb::Table:




## Public Member Functions

- Table (const TableRow &headers)
  *Constructor.*
- Table (TableRow &&headers)
  *Constructor with move semantics.*
- Table (const Table &table)
  *Copy constructor.*
- Table (Table &&table)
  *Move constructor.*
- Table & operator= (const Table &table)
  *Copy assignment operator.*
- Table & operator= (Table &&table)
  *Move assignment operator.*
- ~Table ()
- size_t num_fields () const
  *Returns the number of fields in each row.*
- size_t num_records () const
  *Returns the number of record in the table.*
- void set_quoted (const std::vector< bool > &vec)
  *Sets the quote flags for the records.*
- void set_quoted (std::vector< bool > &&vec)
  *Sets the quote flags for the records with move semantics.*
- const TableRow & get_headers () const
  *Returns the field names.*
- const TableRow & operator[] (const size_t idx) const
  *Overloaded index operator.*
- void append_record (const TableRow &new_record)
  *Appends a record to the table.*
- void append_record (TableRow &&new_record)
  *Appends a record to the table with move semantics.*
- std::string insert_query (const std::string &table_name, const size_t idx)
  *Creates an SQL INSERT query from a table record.*
- std::string get_field (const std::string &field_name, const size_t row_index)
  *Gets a field from a record by field name.*

### Static Public Member Functions

- static Table create_from_file (const std::string &filename, const char delim)

  *Creates a table from an input file.*

### Private Attributes

- TableRow m_headers
- std::vector< TableRow > m_records
- std::vector< bool > m_quoted

## 9.21.1 Detailed Description

Database table class.

## 9.21.2 Constructor & Destructor Documentation

### 9.21.2.1 Table::Table ( const TableRow & *headers* ) [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *headers* | Table row containing field names. |

### 9.21.2.2 Table::Table ( TableRow && *headers* ) [explicit]

Constructor with move semantics.

**Parameters**

| | |
|---|---|
| *headers* | Table row containing field names. |

### 9.21.2.3 Table::Table ( const Table & *table* )

Copy constructor.

**Bug** 'explicit' removed from here after failure to compile at end of MySQL query function.

**Parameters**

| | |
|---|---|
| *table* | Table to copy. |

### 9.21.2.4 Table::Table ( Table && *table* )

Move constructor.

**Parameters**

| | |
|---|---|
| *table* | Table to move. |

#### 9.21.2.5 Table::~Table ( )

Destructor

### 9.21.3 Member Function Documentation

#### 9.21.3.1 void Table::append_record ( const TableRow & *new_record* )

Appends a record to the table.

**Parameters**

| | |
|---|---|
| *new_record* | The record to append. |

#### 9.21.3.2 void Table::append_record ( TableRow && *new_record* )

Appends a record to the table with move semantics.

**Parameters**

| | |
|---|---|
| *new_record* | The record to append. |

#### 9.21.3.3 Table Table::create_from_file ( const std::string & *filename,* const char *delim* ) [static]

Creates a table from an input file.

**Parameters**

| | |
|---|---|
| *filename* | The name of the input file. |
| *delim* | The delimiting character. |

**Returns**

The table.

**Exceptions**

| | |
|---|---|
| *TableBadInputFile* | on badly formed input file. |
| *TableCouldNotOpenInput-File* | on bad filename. |

#### 9.21.3.4 std::string Table::get_field ( const std::string & *field_name,* const size_t *row_index* )

Gets a field from a record by field name.

**Parameters**

| | |
|---|---|
| *field_name* | The name of the field. |
| *row_index* | The index of the row. |

**Returns**

The contents ofthe field.

**Exceptions**

| | |
|---|---|
| *TableNoSuchField* | if `field_name` is not a valid field name. |
| *TableNoSuchRecord* | if there is no record at index `row_index`. |

#### 9.21.3.5 const TableRow& gldb::Table::get_headers ( ) const [inline]

Returns the field names.

**Returns**

The field names.

#### 9.21.3.6 std::string Table::insert_query ( const std::string & *table_name*, const size_t *idx* )

Creates an SQL INSERT query from a table record.

**Parameters**

| | |
|---|---|
| *table_name* | The name of the table into which to INSERT. |
| *idx* | The index of the record. |

**Returns**

A string containing the query.

#### 9.21.3.7 size_t gldb::Table::num_fields ( ) const [inline]

Returns the number of fields in each row.

**Returns**

The number of fields in each row.

#### 9.21.3.8 size_t gldb::Table::num_records ( ) const [inline]

Returns the number of record in the table.

**Returns**

The number of records in the table.

#### 9.21.3.9 Table & Table::operator= ( const Table & *table* )

Copy assignment operator.

**Parameters**

| | |
|---|---|
| *table* | Table to copy. |

**Returns**

Reference to the assigned-to table.

#### 9.21.3.10 Table & Table::operator= ( Table && *table* )

Move assignment operator.

**Parameters**

| | |
|---|---|
| *table* | Table to move. |

**Returns**

Reference to the assigned-to table.

#### 9.21.3.11 const TableRow & Table::operator[] ( const size_t *idx* ) const

Overloaded index operator.

**Parameters**

| | |
|---|---|
| *idx* | The zero-based index of the record. |

**Returns**

The selected record.

#### 9.21.3.12 void Table::set_quoted ( const std::vector< bool > & *vec* )

Sets the quote flags for the records.

**Parameters**

| | |
|---|---|
| *vec* | A vector of bools. The size must match the size of the records. |

#### 9.21.3.13 void Table::set_quoted ( std::vector< bool > && *vec* )

Sets the quote flags for the records with move semantics.

**Parameters**

| | |
|---|---|
| *vec* | A vector of bools. The size must match the size of the records. |

### 9.21.4 Member Data Documentation

#### 9.21.4.1 TableRow gldb::Table::m_headers [private]

The names of the fields

#### 9.21.4.2 std::vector<bool> gldb::Table::m_quoted [private]

A vector to show if fields should be quoted for INSERT

#### 9.21.4.3 std::vector<TableRow> gldb::Table::m_records [private]

A vector of the records

The documentation for this class was generated from the following files:

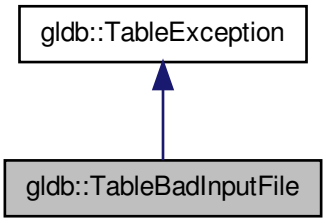
- lib/database/table.h
- lib/database/table.cpp

## 9.22 gldb::TableBadInputFile Class Reference

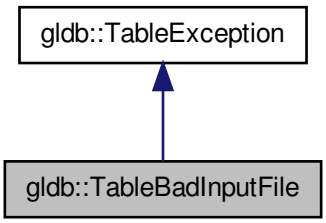
Could not connect to database exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableBadInputFile:




Collaboration diagram for gldb::TableBadInputFile:




### Public Member Functions

- TableBadInputFile (const std::string &msg)

*Constructor.*

### 9.22.1 Detailed Description

Could not connect to database exception class.

### 9.22.2 Constructor & Destructor Documentation

#### 9.22.2.1 gldb::TableBadInputFile::TableBadInputFile ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

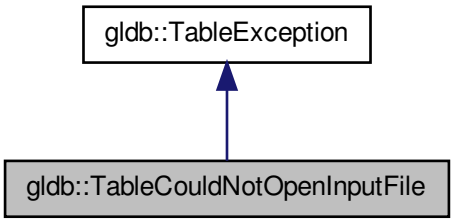
- lib/database/table.h

## 9.23 gldb::TableCouldNotOpenInputFile Class Reference

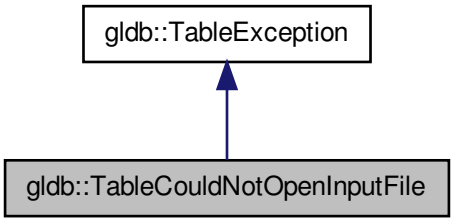
Could not connect to database exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableCouldNotOpenInputFile:



Collaboration diagram for gldb::TableCouldNotOpenInputFile:




### Public Member Functions

- TableCouldNotOpenInputFile (const std::string &msg)
  *Constructor.*

### 9.23.1 Detailed Description

Could not connect to database exception class.

### 9.23.2 Constructor & Destructor Documentation

#### 9.23.2.1 gldb::TableCouldNotOpenInputFile::TableCouldNotOpenInputFile ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

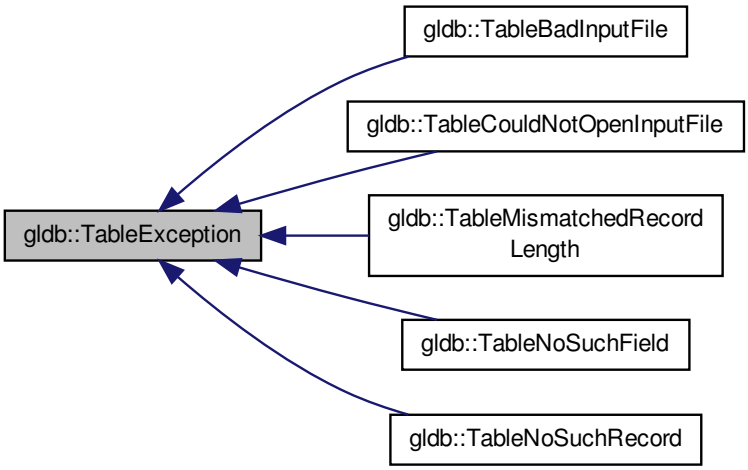
- lib/database/table.h

## 9.24 gldb::TableException Class Reference

Base database connection exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableException:




### Public Member Functions

- TableException (const std::string &msg)
  *Constructor.*

### 9.24.1 Detailed Description

Base database connection exception class.

### 9.24.2 Constructor & Destructor Documentation

#### 9.24.2.1 gldb::TableException::TableException ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

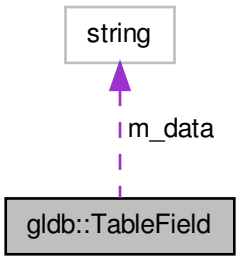
- lib/database/table.h

## 9.25 gldb::TableField Class Reference

Database table field class.

`#include <tablefield.h>`

Collaboration diagram for gldb::TableField:




## Public Member Functions

- TableField (const char *data)
  *Constructor accepting `const char *` data.*
- TableField (const std::string &data)
  *Constructor accepting `std:string` data.*
- TableField (std::string &&data)
  *Constructor accepting `std:string` data with move semantics.*
- TableField (const TableField &field)
  *Copy constructor.*
- TableField (TableField &&field)
  *Move constructor.*
- ~TableField ()
- size_t length () const
  *Returns the length of the field.*
- operator std::string () const
  *Overridden conversion operator.*
- TableField & operator= (const char *data)
  *Overridden assignment operator for `const char *`.*
- TableField & operator= (const std::string &data)
  *Overridden assignment operator for `std::string`.*
- TableField & operator= (std::string &&data)
  *Overridden assignment operator for `std::string` with move semantics.*
- TableField & operator= (const TableField &field)
  *Overridden copy assignment operator.*
- TableField & operator= (TableField &&field)
  *Overridden move assignment operator.*
- char & operator[] (const size_t idx)
  *Overridden index operator.*
- const char & operator[] (const size_t idx) const
  *Overridden index operator.*
- TableField & operator+= (const char c)
  *Overridden compound assignment operator.*
- TableField & operator+= (const std::string &data)
  *Overridden compound assignment operator.*

### Private Attributes

- std::string m_data

### Friends

- std::ostream & operator<< (std::ostream &out, const TableField &field)

  *Overridden << operator for printing a field.*

## 9.25.1 Detailed Description

Database table field class.

## 9.25.2 Constructor & Destructor Documentation

#### 9.25.2.1 TableField::TableField ( const char * *data* ) [explicit]

Constructor accepting `const char *` data.

**Parameters**

| | |
|---|---|
| *data* | The initial contents of the field. |

#### 9.25.2.2 TableField::TableField ( const std::string & *data* )

Constructor accepting `std:string` data.

**Parameters**

| | |
|---|---|
| *data* | The initial contents of the field. |

#### 9.25.2.3 TableField::TableField ( std::string && *data* )

Constructor accepting `std:string` data with move semantics.

**Parameters**

| | |
|---|---|
| *data* | The initial contents of the field. |

#### 9.25.2.4 TableField::TableField ( const TableField & *field* )

Copy constructor.

**Parameters**

| | |
|---|---|
| *field* | The field from which to copy. |

#### 9.25.2.5 TableField::TableField ( TableField && *field* )

Move constructor.

**Parameters**

| | |
|---|---|
| *field* | The field from which to move. |

#### 9.25.2.6 TableField::~TableField ( )

Destructor

### 9.25.3 Member Function Documentation

#### 9.25.3.1 size_t gldb::TableField::length ( ) const [inline]

Returns the length of the field.

**Returns**

The length of the field.

#### 9.25.3.2 gldb::TableField::operator std::string ( ) const [inline]

Overridden conversion operator.

Returns the field contents as a string.

#### 9.25.3.3 TableField & TableField::operator+= ( const char *c* )

Overridden compound assignment operator.

**Parameters**

| | |
|---|---|
| *c* | The character to append to the field. |

**Returns**

A reference to the same field.

#### 9.25.3.4 TableField & TableField::operator+= ( const std::string & *data* )

Overridden compound assignment operator.

**Parameters**

| | |
|---|---|
| *data* | The string to append to the field. |

**Returns**

A reference to the same field.

#### 9.25.3.5 TableField & TableField::operator= ( const char * *data* )

Overridden assignment operator for `const char *`.

**Parameters**

| | |
|---|---|
| *data* | The new contents of the field. |

**Returns**

A reference to the same field.

#### 9.25.3.6 TableField & TableField::operator= ( const std::string & *data* )

Overridden assignment operator for `std::string`.

**Parameters**

| | |
|---|---|
| *data* | The new contents of the field. |

**Returns**

A reference to the same field.

#### 9.25.3.7 TableField & TableField::operator= ( std::string && *data* )

Overridden assignment operator for `std::string` with move semantics.

**Parameters**

| | |
|---|---|
| *data* | The new contents of the field. |

**Returns**

A reference to the same field.

#### 9.25.3.8 TableField & TableField::operator= ( const TableField & *field* )

Overridden copy assignment operator.

**Parameters**

| | |
|---|---|
| *field* | The field to copy. |

**Returns**

A reference to the same field.

#### 9.25.3.9 TableField & TableField::operator= ( TableField && *field* )

Overridden move assignment operator.

**Parameters**

| | |
|---|---|
| *field* | The field to move. |

**Returns**

A reference to the same field.

#### 9.25.3.10 char& gldb::TableField::operator[] ( const size_t *idx* ) [inline]

Overridden index operator.

**Parameters**

| | |
|---|---|
| *idx* | The desired index. |

**Returns**

A reference to the character at the specified index.

#### 9.25.3.11 const char& gldb::TableField::operator[] ( const size_t *idx* ) const [inline]

Overridden index operator.

**Parameters**

| | |
|---|---|
| *idx* | The desired index. |

**Returns**

A const reference to the character at the specified index.

### 9.25.4 Friends And Related Function Documentation

#### 9.25.4.1 std::ostream& operator<< ( std::ostream & *out*, const TableField & *field* ) [friend]

Overridden << operator for printing a field.

**Parameters**

| | |
|---|---|
| *out* | The ostream to which to print. |
| *field* | A reference to the field. |

**Returns**

A reference to `out`.

### 9.25.5 Member Data Documentation

#### 9.25.5.1 std::string gldb::TableField::m_data [private]

The field contents

The documentation for this class was generated from the following files:

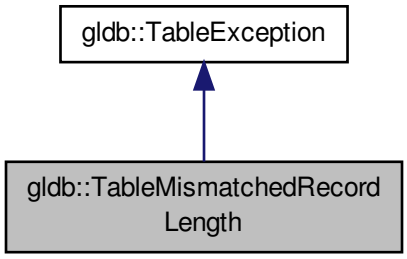
- lib/database/tablefield.h
- lib/database/tablefield.cpp

## 9.26 gldb::TableMismatchedRecordLength Class Reference

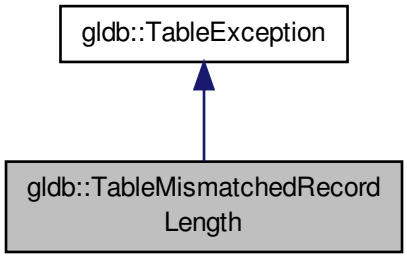
Mismatched record length exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableMismatchedRecordLength:




Collaboration diagram for gldb::TableMismatchedRecordLength:




### Public Member Functions

- TableMismatchedRecordLength (const std::string &msg)

  *Constructor.*

### 9.26.1 Detailed Description

Mismatched record length exception class.

### 9.26.2 Constructor & Destructor Documentation

#### 9.26.2.1 gldb::TableMismatchedRecordLength::TableMismatchedRecordLength ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

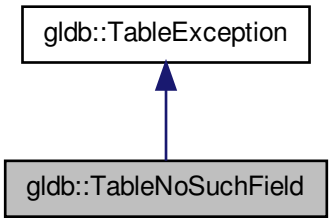
- lib/database/table.h

## 9.27 gldb::TableNoSuchField Class Reference

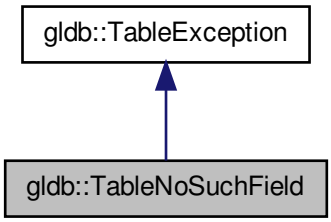
No such field exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableNoSuchField:




Collaboration diagram for gldb::TableNoSuchField:




### Public Member Functions

- TableNoSuchField (const std::string &msg)

  *Constructor.*

### 9.27.1 Detailed Description

No such field exception class.

### 9.27.2 Constructor & Destructor Documentation

#### 9.27.2.1 gldb::TableNoSuchField::TableNoSuchField ( const std::string & *msg* ) [inline], [explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

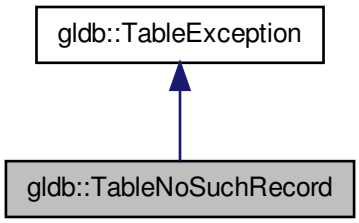
- lib/database/table.h

## 9.28 gldb::TableNoSuchRecord Class Reference

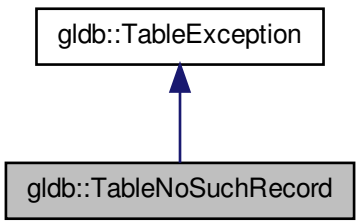
No such record exception class.

`#include <table.h>`

Inheritance diagram for gldb::TableNoSuchRecord:




Collaboration diagram for gldb::TableNoSuchRecord:



### Public Member Functions

- TableNoSuchRecord (const std::string &msg)

  *Constructor.*

### 9.28.1 Detailed Description

No such record exception class.

### 9.28.2 Constructor & Destructor Documentation

#### 9.28.2.1 gldb::TableNoSuchRecord::TableNoSuchRecord ( const std::string & *msg* ) [inline],[explicit]

Constructor.

**Parameters**

| | |
|---|---|
| *msg* | Database error message |

The documentation for this class was generated from the following file:

- lib/database/table.h

## 9.29 gldb::TableRow Class Reference

Database table row class.

`#include <tablerow.h>`

### Public Member Functions

- TableRow ()
- TableRow (const size_t size)

  *Constructor with initial number of fields.*
- TableRow (const std::vector< std::string > &vec)

  *Constructor with string vector.*
- TableRow (std::vector< std::string > &&vec)

  *Constructor with string vector and move semantics.*
- TableRow (std::initializer_list< std::string > i)

  *Constructor with std::string initializer list.*
- TableRow (const TableRow &row)

  *Copy constructor.*
- TableRow (TableRow &&row)

  *Move constructor.*
- TableRow & operator= (const TableRow &row)

  *Copy assignment operator.*
- TableRow & operator= (TableRow &&row)

  *Move assignment operator.*
- ~TableRow ()
- size_t size () const

  *Returns the number of fields.*

- iterator begin ()

  *Returns iterator for beginning.*
- iterator end ()

  *Returns iterator for end plus one.*
- const_iterator begin () const

  *Returns const iterator for beginning.*
- const_iterator end () const

  *Returns const iterator for end plus one.*
- TableField & operator[] (const size_t idx)

  *Overridden index operator.*
- const TableField & operator[] (const size_t idx) const

  *Overridden index operator.*
- void append_field (const char *new_field)

  *Appends a field to the row.*
- void append_field (const std::string &new_field)

  *Appends a field to the row.*
- void append_field (std::string &&new_field)

  *Appends a field to the row with move semantics.*
- void append_field (const TableField &new_field)

  *Appends a field to the row.*
- void append_field (TableField &&new_field)

  *Appends a field to the row with move semantics.*
- void print (std::ostream &stream) const

  *Prints a row.*
- std::string record_string (const std::vector< bool > &quoted) const

  *Creates a comma separated string of fields.*
- std::string record_string () const

  *Creates an unquoted comma separated string of fields.*

## Private Attributes

- std::vector< TableField > m_fields

### 9.29.1 Detailed Description

Database table row class.

### 9.29.2 Constructor & Destructor Documentation

#### 9.29.2.1 TableRow::TableRow ( )

Default constructor

#### 9.29.2.2 TableRow::TableRow ( const size_t *size* ) [explicit]

Constructor with initial number of fields.

**Parameters**

| | |
|---|---|
| *size* | The initial number of fields. |

#### 9.29.2.3 TableRow::TableRow ( const std::vector< std::string > & *vec* ) [explicit]

Constructor with string vector.

**Parameters**

| | |
|---|---|
| *vec* | The vector. |

#### 9.29.2.4 TableRow::TableRow ( std::vector< std::string > && *vec* ) [explicit]

Constructor with string vector and move semantics.

**Parameters**

| | |
|---|---|
| *vec* | The vector. |

#### 9.29.2.5 TableRow::TableRow ( std::initializer_list< std::string > *i* ) [explicit]

Constructor with std::string initializer list.

**Parameters**

| | |
|---|---|
| *i* | The initializer list. |

#### 9.29.2.6 TableRow::TableRow ( const TableRow & *row* )

Copy constructor.

**Parameters**

| | |
|---|---|
| *row* | The row to copy. |

#### 9.29.2.7 TableRow::TableRow ( TableRow && *row* )

Move constructor.

**Parameters**

| | |
|---|---|
| *row* | The row to move. |

#### 9.29.2.8 TableRow::~TableRow ( )

Destructor

### 9.29.3 Member Function Documentation

#### 9.29.3.1 void TableRow::append_field ( const char * *new_field* )

Appends a field to the row.

**Parameters**

| | |
|---|---|
| *new_field* | The contents of the new field. |

#### 9.29.3.2 void TableRow::append_field ( const std::string & *new_field* )

Appends a field to the row.

**Parameters**

| | |
|---|---|
| *new_field* | The contents of the new field. |

#### 9.29.3.3 void TableRow::append_field ( std::string && *new_field* )

Appends a field to the row with move semantics.

**Parameters**

| | |
|---|---|
| *new_field* | The contents of the new field. |

#### 9.29.3.4 void TableRow::append_field ( const TableField & *new_field* )

Appends a field to the row.

**Parameters**

| | |
|---|---|
| *new_field* | A field from which to copy. |

#### 9.29.3.5 void TableRow::append_field ( TableField && *new_field* )

Appends a field to the row with move semantics.

**Parameters**

| | |
|---|---|
| *new_field* | A field from which to copy. |

#### 9.29.3.6 iterator gldb::TableRow::begin ( ) [inline]

Returns iterator for beginning.

**Returns**

Iterator for beginning.

#### 9.29.3.7 const_iterator gldb::TableRow::begin ( ) const [inline]

Returns const iterator for beginning.

**Returns**

Const iterator for beginning.

#### 9.29.3.8 iterator gldb::TableRow::end ( ) [inline]

Returns iterator for end plus one.

**Returns**

Iterator for end plus one.

#### 9.29.3.9 const_iterator gldb::TableRow::end ( ) const [inline]

Returns const iterator for end plus one.

**Returns**

Const iterator for end plus one.

#### 9.29.3.10 TableRow & TableRow::operator= ( const TableRow & *row* )

Copy assignment operator.

**Parameters**

| | |
|---|---|
| *row* | The row to copy. |

**Returns**

A reference to the assigned-to row.

#### 9.29.3.11 TableRow & TableRow::operator= ( TableRow && *row* )

Move assignment operator.

**Parameters**

| | |
|---|---|
| *row* | The row to move. |

**Returns**

A reference to the assigned-to row.

#### 9.29.3.12 TableField& gldb::TableRow::operator[] ( const size_t *idx* ) [inline]

Overridden index operator.

**Parameters**

| | |
|---|---|
| *idx* | The zero-based index of the field. |

**Returns**

A reference to the field at the specified index.

#### 9.29.3.13 const TableField& gldb::TableRow::operator[] ( const size_t *idx* ) const [inline]

Overridden index operator.

**Parameters**

| | |
|---|---|
| *idx* | The zero-based index of the field. |

**Returns**

A const reference to the field at the specified index.

#### 9.29.3.14 void TableRow::print ( std::ostream & *stream* ) const

Prints a row.

**Parameters**

| | |
|---|---|
| *stream* | The ostream to which to print. |

#### 9.29.3.15 std::string TableRow::record_string ( const std::vector< bool > & *quoted* ) const

Creates a comma separated string of fields.

**Parameters**

| | |
|---|---|
| *quoted* | A vector of `bool`, for each field `true` means that field will be enclosed in single quotes in the comma separated string, `false` means it will not be. |

**Returns**

The comma separated string.

#### 9.29.3.16 std::string TableRow::record_string ( ) const

Creates an unquoted comma separated string of fields.

**Returns**

The unquoted comma separated string.

#### 9.29.3.17 size_t gldb::TableRow::size ( ) const [inline]

Returns the number of fields.

**Returns**

The number of fields.

### 9.29.4 Member Data Documentation

#### 9.29.4.1 std::vector<TableField> gldb::TableRow::m_fields [private]

A vector of fields

The documentation for this class was generated from the following files:

- lib/database/tablerow.h
- lib/database/tablerow.cpp

# Chapter 10

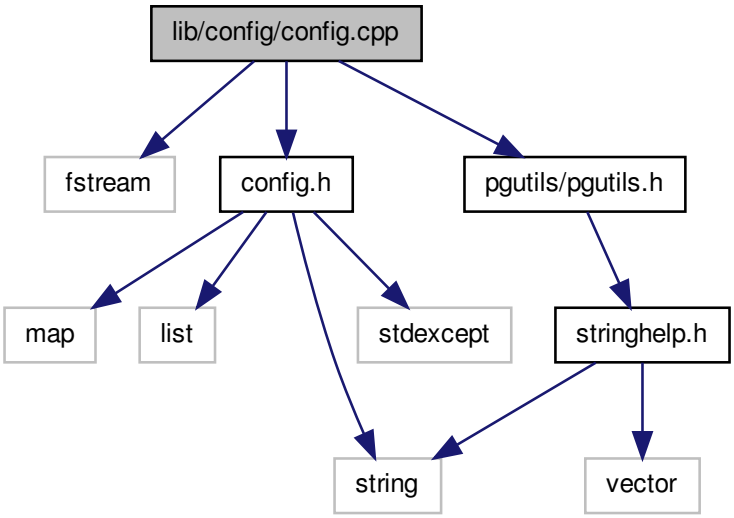
# File Documentation

## 10.1 lib/config/config.cpp File Reference

Implementation of program configurations class.

```
#include <fstream>
#include "config.h"
#include "pgutils/pgutils.h"
```

Include dependency graph for config.cpp:




### 10.1.1 Detailed Description

Implementation of program configurations class.

**Author**

Paul Griffiths

### Copyright

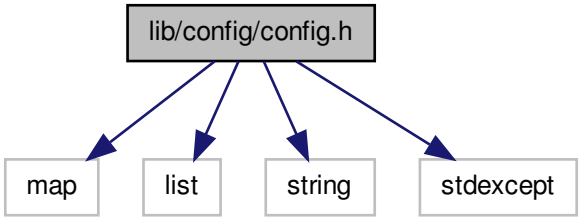
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.2 lib/config/config.h File Reference

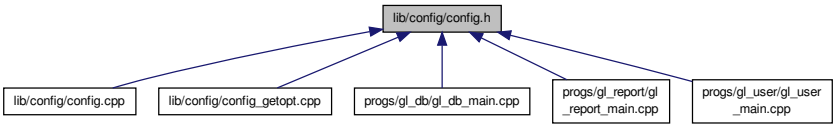
Interface to program configurations class.

```
#include <map>
#include <list>
#include <string>
#include <stdexcept>
```

Include dependency graph for config.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class genleg::ConfigException

  *Configuration module exception base class.*
- class genleg::ConfigOptionNotSet

  *Exception class for option not set.*
- class genleg::ConfigBadOption

  *Exception class for bad provided option.*
- class genleg::ConfigCouldNotOpenFile

  *Exception class for when conf file cannot be opened.*
- class genleg::ConfigBadConfigFile

  *Exception class for badly formed configuration file.*
- class genleg::Config

  *Configuration options class.*

### Enumerations

- enum genleg::Argument

  *Enumeration class for option argument specifications.*

### 10.2.1 Detailed Description

Interface to program configurations class.

#### Author

Paul Griffiths

#### Copyright

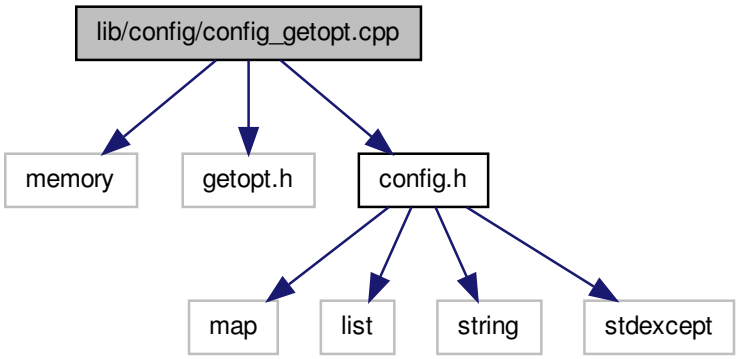
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.3 lib/config/config_getopt.cpp File Reference

Implementation of command line functionality.

```
#include <memory>
#include <getopt.h>
#include "config.h"
```

Include dependency graph for config_getopt.cpp:




### Macros

- #define _XOPEN_SOURCE 600

### 10.3.1 Detailed Description

Implementation of command line functionality. Included in separate file to isolate usage of non-standard getopt library.

**Author**

Paul Griffiths

**Copyright**

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

### 10.3.2 Macro Definition Documentation

#### 10.3.2.1 #define _XOPEN_SOURCE 600

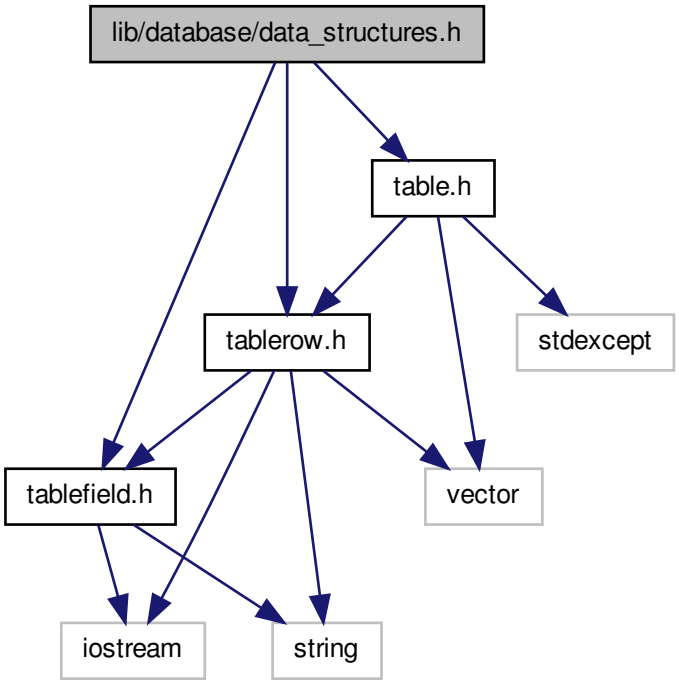
UNIX feature test macro for getopt library

## 10.4 lib/database/data_structures.h File Reference

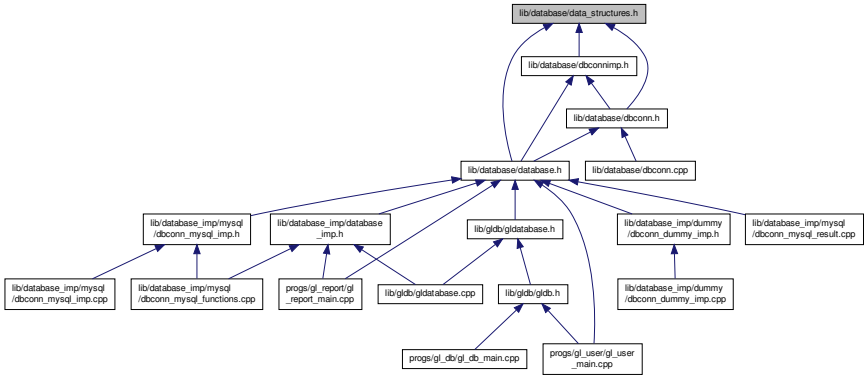
Main interface to database data structures.

```
#include "tablefield.h"
#include "tablerow.h"
#include "table.h"
```

Include dependency graph for data_structures.h:



This graph shows which files directly or indirectly include this file:




### 10.4.1 Detailed Description

Main interface to database data structures.

#### Author

Paul Griffiths

#### Copyright

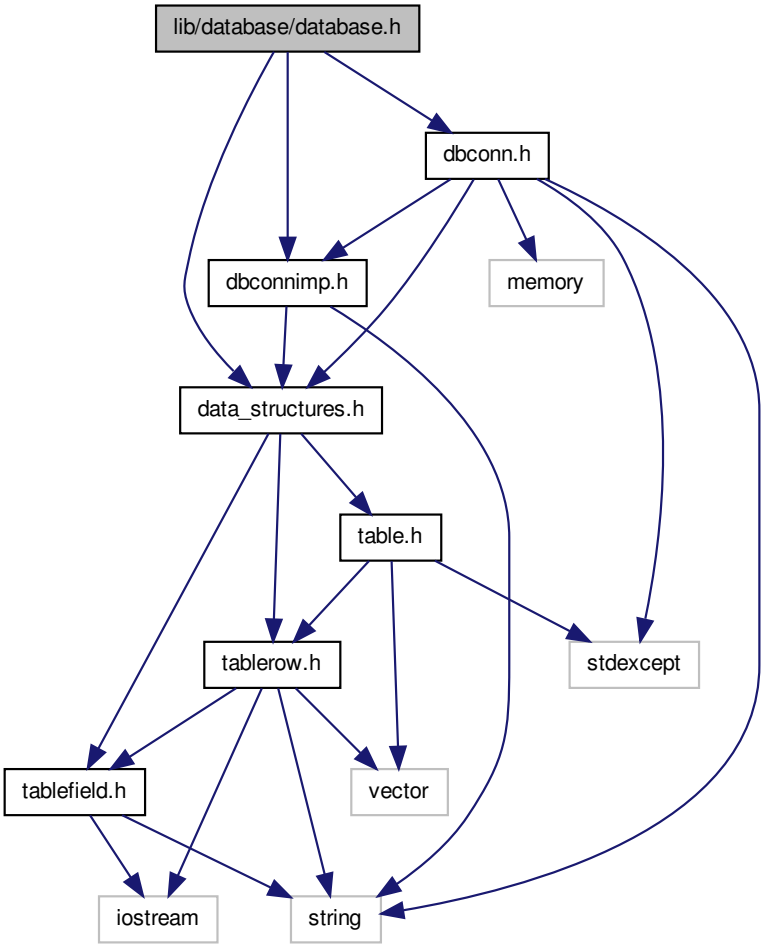
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.5 lib/database/database.h File Reference

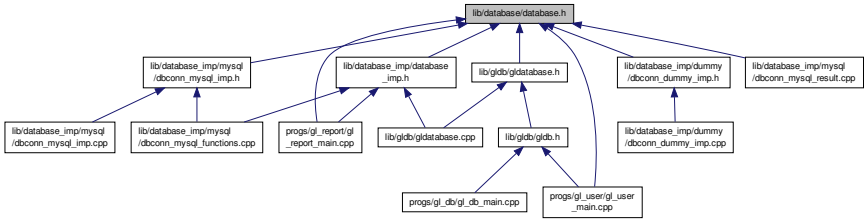
User interface to database functionality.

```
#include "data_structures.h"
#include "dbconnimp.h"
#include "dbconn.h"
```

Include dependency graph for database.h:




This graph shows which files directly or indirectly include this file:




### 10.5.1 Detailed Description

User interface to database functionality.

**Author**

Paul Griffiths

**Copyright**

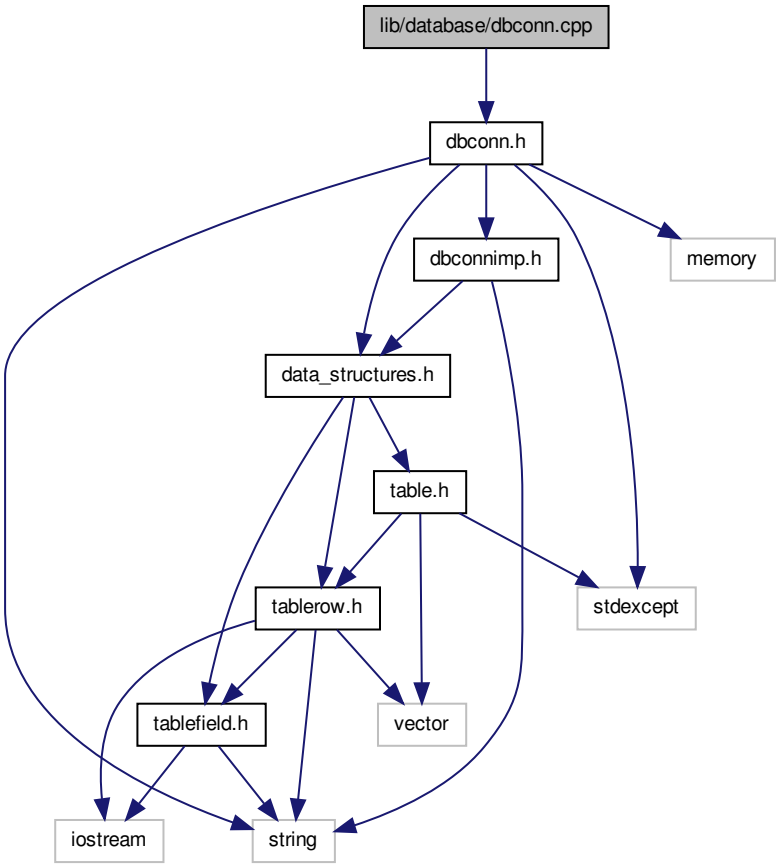
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.6 lib/database/dbconn.cpp File Reference

Implementation of database connection class.

```
#include "dbconn.h"
```

Include dependency graph for dbconn.cpp:




### 10.6.1 Detailed Description

Implementation of database connection class.

**Author**

Paul Griffiths

### Copyright

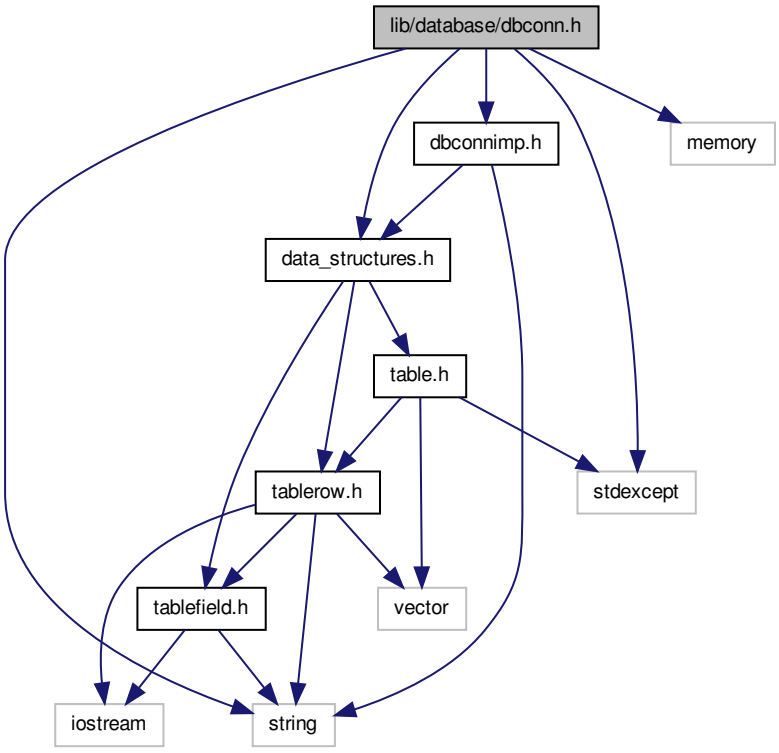
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.7 lib/database/dbconn.h File Reference

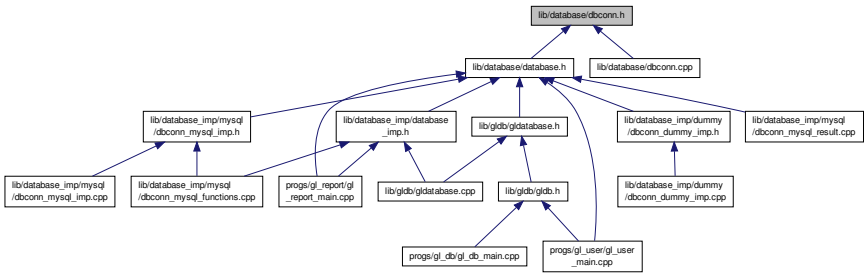
Interface to database connection base class.

```
#include <string>
#include <memory>
#include <stdexcept>
#include "data_structures.h"
#include "dbconnimp.h"
```

Include dependency graph for dbconn.h:



This graph shows which files directly or indirectly include this file:




## Classes

- class gldb::DBConnException

  *Base database connection exception class.*

- class gldb::DBConnCouldNotConnect

  *Could not connect to database exception class.*

- class gldb::DBConnCouldNotQuery

  *Could not execute database query exception class.*

- class gldb::DBConn

  *Database connection class.*

### 10.7.1 Detailed Description

Interface to database connection base class.

#### Author

Paul Griffiths

#### Copyright

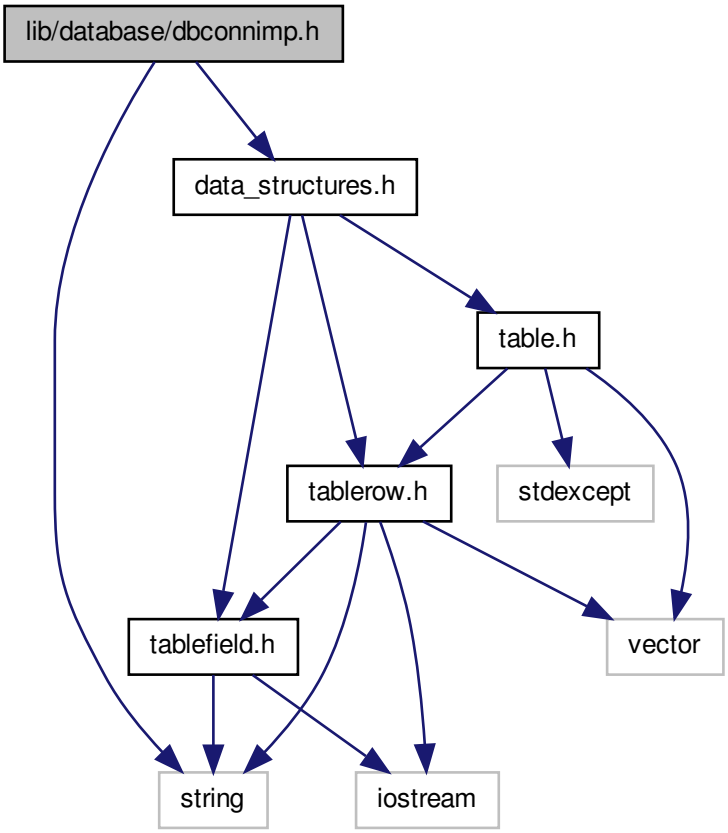
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.8 lib/database/dbconnimp.h File Reference

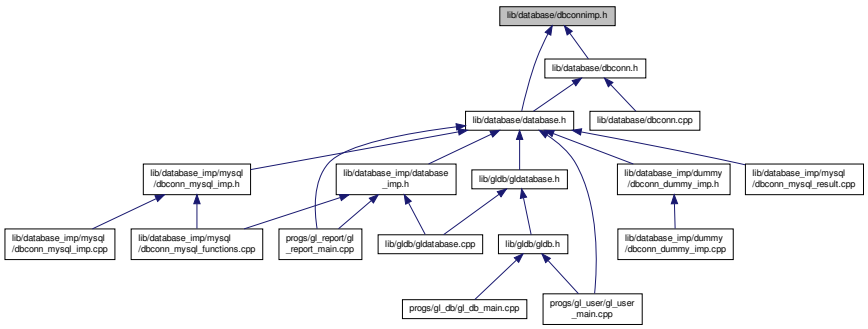
Interface to abstract database implementation base class.

```
#include <string>
#include "data_structures.h"
```

Include dependency graph for dbconnimp.h:




This graph shows which files directly or indirectly include this file:




## Classes

- class gldb::DBConnImp

  *Abstract database implementation base class.*

### 10.8.1 Detailed Description

Interface to abstract database implementation base class.

**Author**

Paul Griffiths

**Copyright**

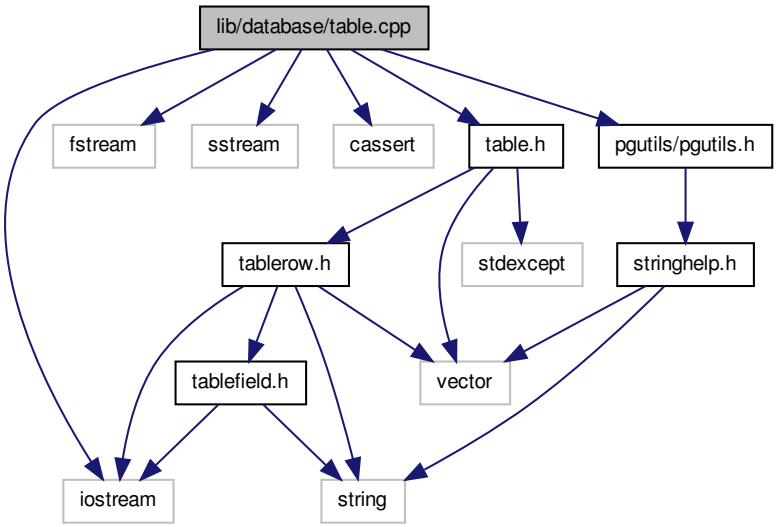
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.9 lib/database/table.cpp File Reference

Implementation of database table data structure.

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <cassert>
#include "table.h"
#include "pgutils/pgutils.h"
```

Include dependency graph for table.cpp:




### 10.9.1 Detailed Description

Implementation of database table data structure.

**Author**

Paul Griffiths

### Copyright

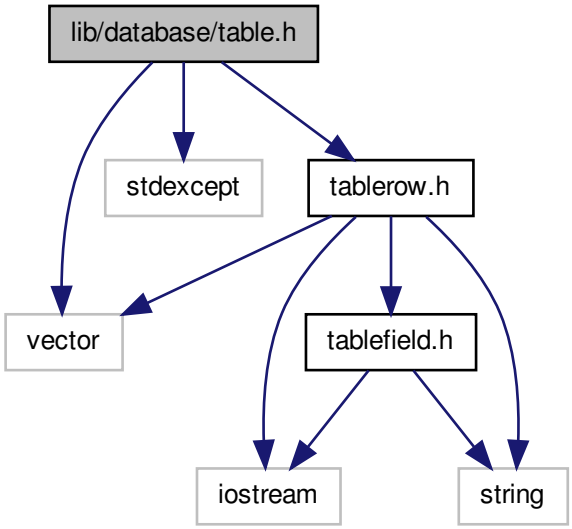
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.10 lib/database/table.h File Reference

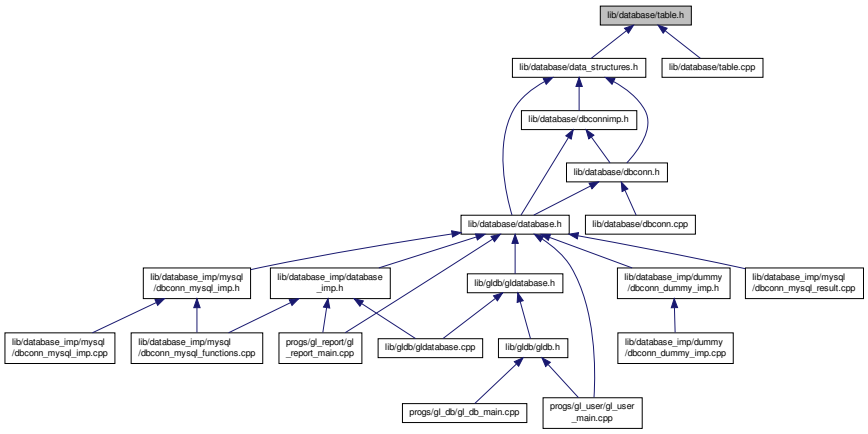
Interface to database table data structure.

```
#include <vector>
#include <stdexcept>
#include "tablerow.h"
```

Include dependency graph for table.h:




This graph shows which files directly or indirectly include this file:



### Classes

- class gldb::TableException
  *Base database connection exception class.*
- class gldb::TableNoSuchField
  *No such field exception class.*
- class gldb::TableNoSuchRecord
  *No such record exception class.*
- class gldb::TableMismatchedRecordLength
  *Mismatched record length exception class.*
- class gldb::TableBadInputFile
  *Could not connect to database exception class.*
- class gldb::TableCouldNotOpenInputFile
  *Could not connect to database exception class.*
- class gldb::Table
  *Database table class.*

### 10.10.1 Detailed Description

Interface to database table data structure.

**Author**

Paul Griffiths

**Copyright**

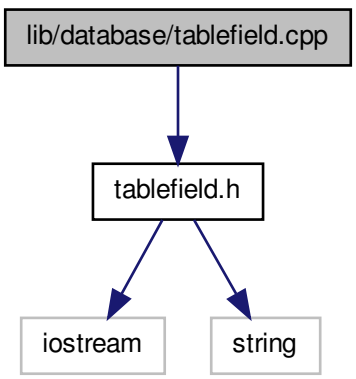
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.11 lib/database/tablefield.cpp File Reference

Implementation of database table field class.

```
#include "tablefield.h"
```

Include dependency graph for tablefield.cpp:



### 10.11.1 Detailed Description

Implementation of database table field class.

**Author**

Paul Griffiths

**Copyright**

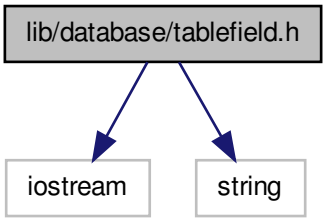
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.12 lib/database/tablefield.h File Reference

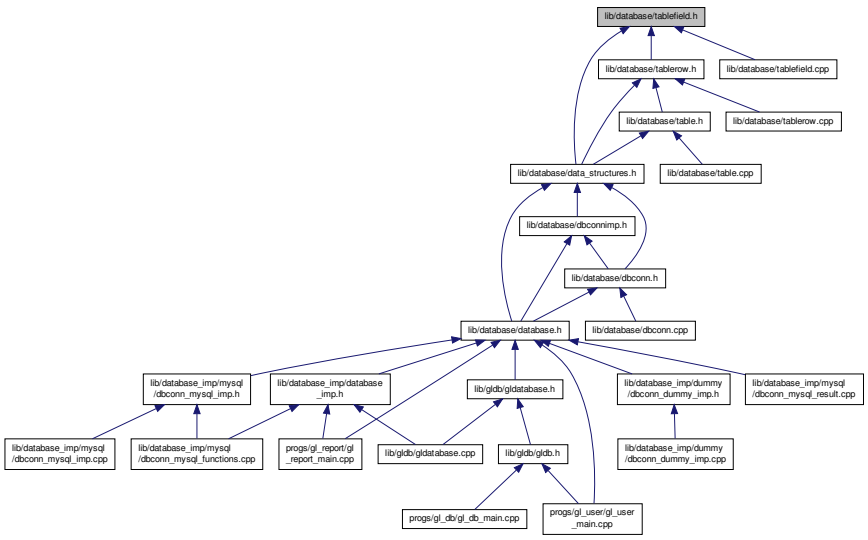
Interface to database table field class.

```
#include <iostream>
#include <string>
```

Include dependency graph for tablefield.h:




This graph shows which files directly or indirectly include this file:



### Classes

- class gldb::TableField

  *Database table field class.*

### Functions

- std::ostream & **gldb::operator**<< (std::ostream &out, const TableField &field)

  *Overridden << operator for printing a field.*

#### 10.12.1 Detailed Description

Interface to database table field class.

**Author**

Paul Griffiths

**Copyright**

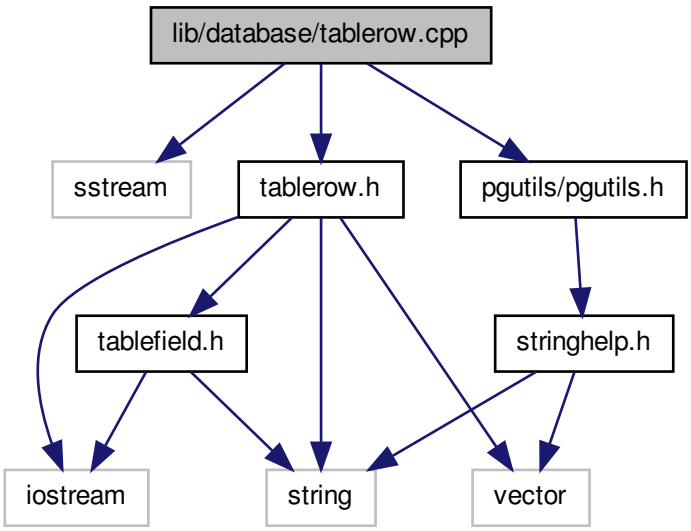
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.13 lib/database/tablerow.cpp File Reference

Implementation of database table row data structure.

```
#include <sstream>
#include "tablerow.h"
#include "pgutils/pgutils.h"
```

Include dependency graph for tablerow.cpp:



### 10.13.1 Detailed Description

Implementation of database table row data structure.

**Author**

Paul Griffiths

**Copyright**

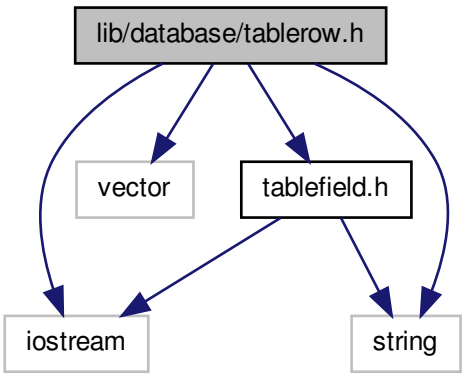
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.14 lib/database/tablerow.h File Reference

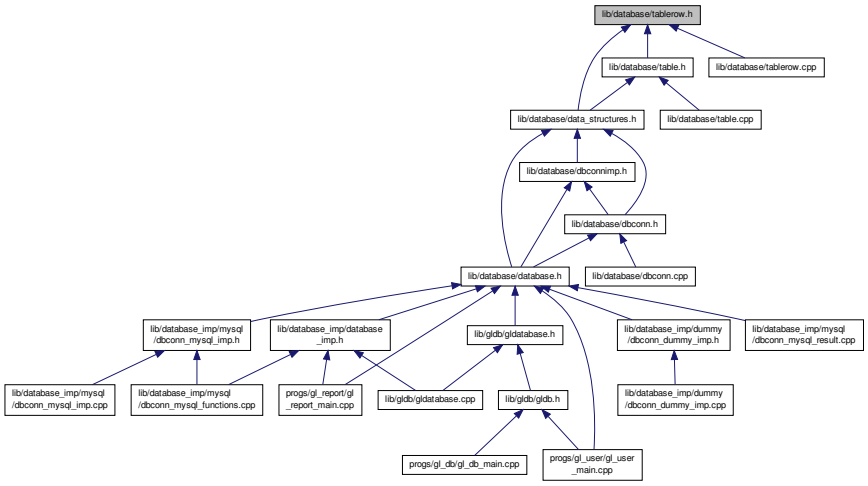
Interface to database table row data structure.

```
#include <iostream>
#include <vector>
#include <string>
#include "tablefield.h"
```

Include dependency graph for tablerow.h:



This graph shows which files directly or indirectly include this file:




## Classes

- class gldb::TableRow

  *Database table row class.*

### 10.14.1 Detailed Description

Interface to database table row data structure.

#### Author

Paul Griffiths

### Copyright

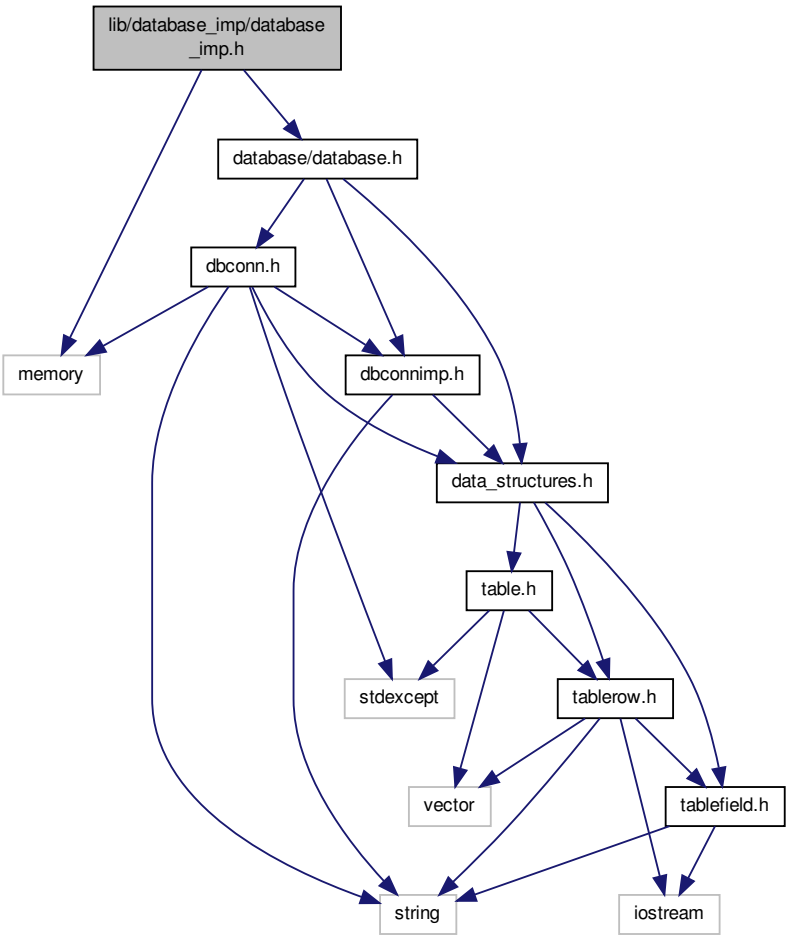
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.15 lib/database_imp/database_imp.h File Reference

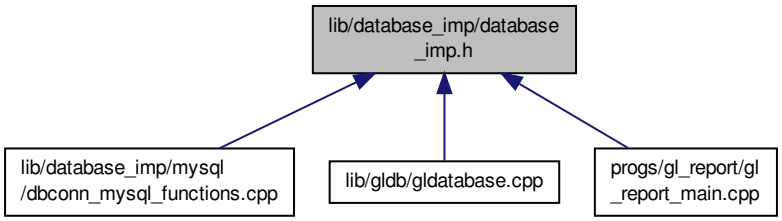
Interface to database implementation factory function.

```
#include <memory>
#include "database/database.h"
```

Include dependency graph for database_imp.h:



This graph shows which files directly or indirectly include this file:




## Functions

- DBConnImp * gldb::get_connection (const std::string &database, const std::string &hostname, const std::string &username, const std::string &password)

  *Creates and returns a pointer to a database implementation.*

- std::string gldb::get_database_type ()

  *Returns the name of the compiled-in database type.*

### 10.15.1 Detailed Description

Interface to database implementation factory function.

**Author**

Paul Griffiths

**Copyright**

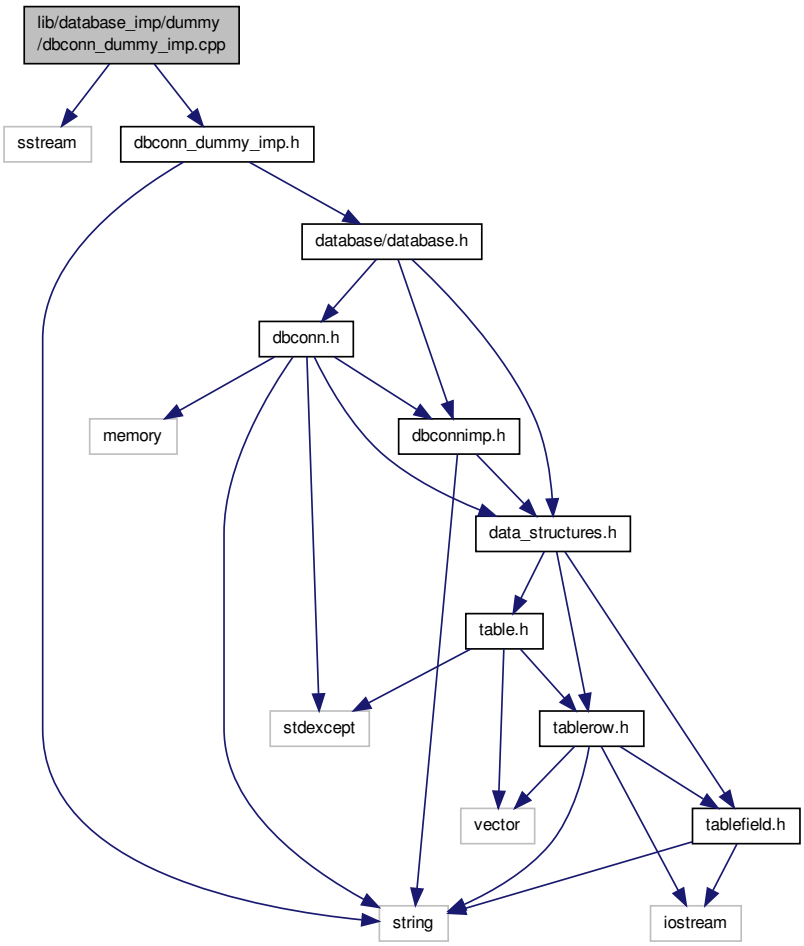
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.16 lib/database_imp/dummy/dbconn_dummy_imp.cpp File Reference

Implementation of Dummy database connection implementation class.

```
#include <sstream>
#include "dbconn_dummy_imp.h"
```

Include dependency graph for dbconn_dummy_imp.cpp:




### 10.16.1 Detailed Description

Implementation of Dummy database connection implementation class.

**Author**

Paul Griffiths

### Copyright

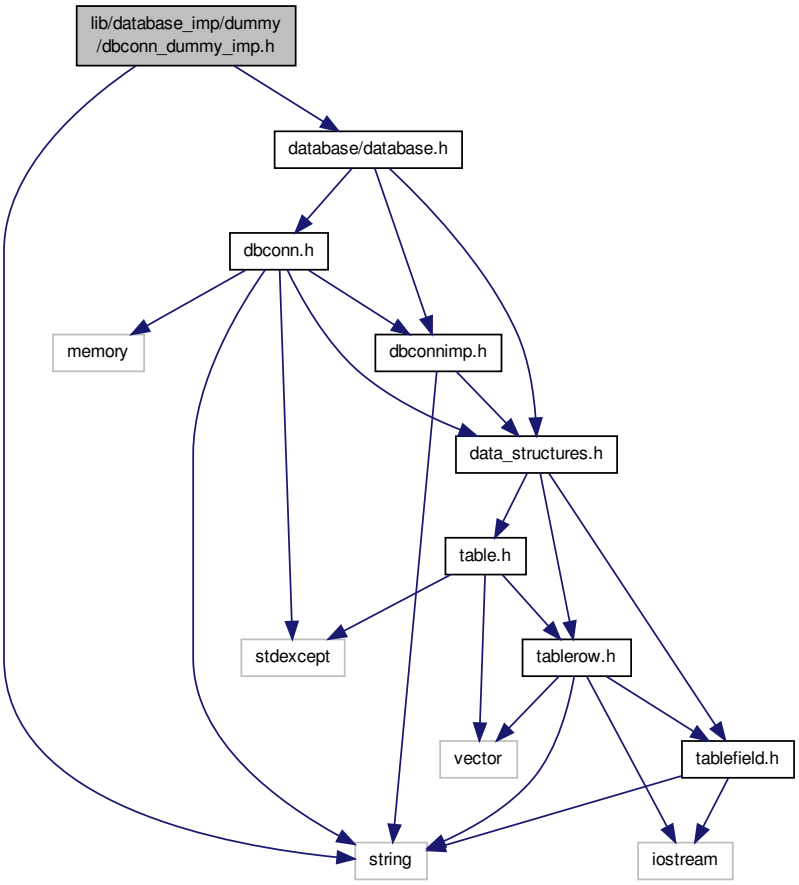
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.17 lib/database_imp/dummy/dbconn_dummy_imp.h File Reference

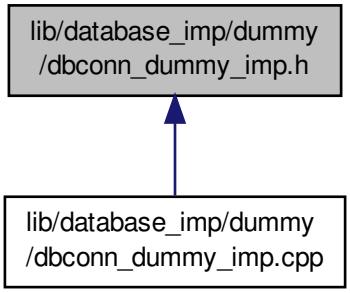
Interface to dummy database connection implementation class.

```
#include <string>
#include "database/database.h"
```

Include dependency graph for dbconn_dummy_imp.h:



This graph shows which files directly or indirectly include this file:




## Classes

- class gldb::DBConnDummy

  *Dummy database implementation class.*

### 10.17.1 Detailed Description

Interface to dummy database connection implementation class.

**Author**

Paul Griffiths

**Copyright**

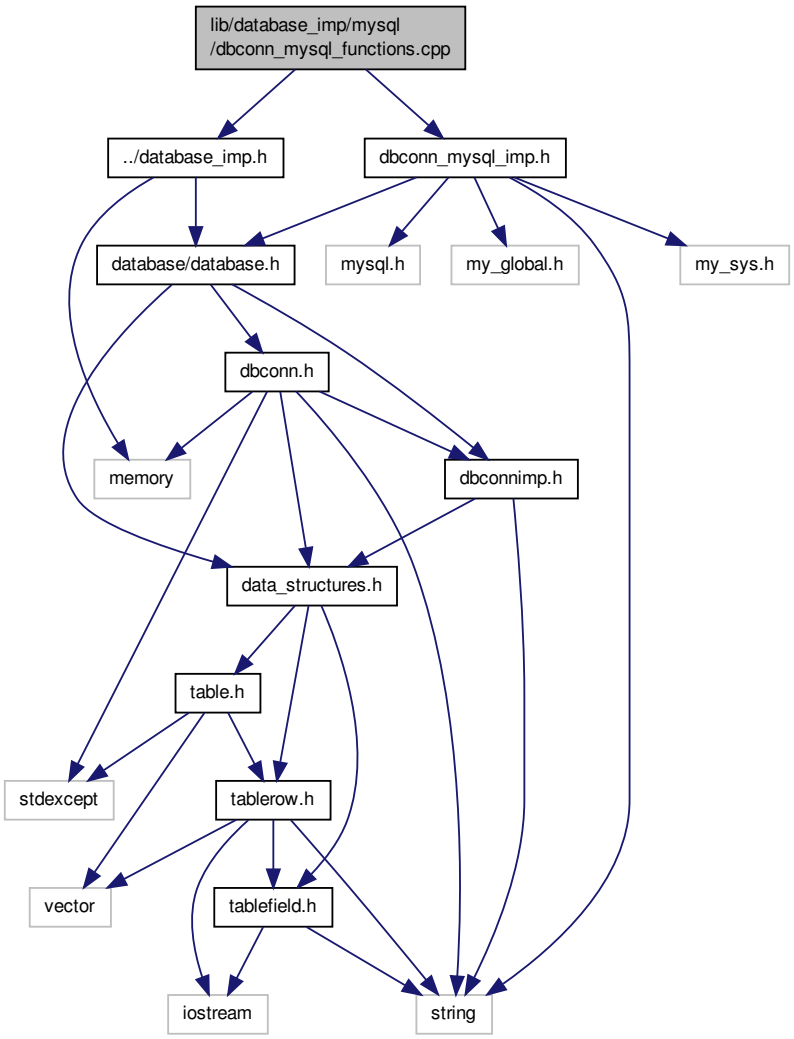
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.18 lib/database_imp/mysql/dbconn_mysql_functions.cpp File Reference

Implementation of MySQL implementation factory function.

```
#include "../database_imp.h"
#include "dbconn_mysql_imp.h"
```

Include dependency graph for dbconn_mysql_functions.cpp:




### 10.18.1 Detailed Description

Implementation of MySQL implementation factory function.

**Author**

Paul Griffiths

**Copyright**

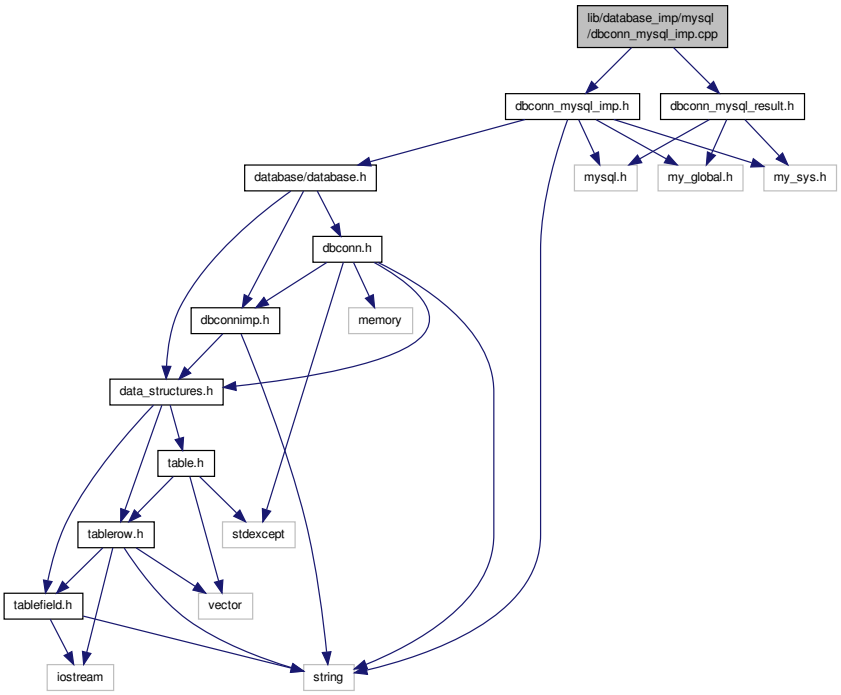
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.19 lib/database_imp/mysql/dbconn_mysql_imp.cpp File Reference

Implementation of MySQL database connection implementation class.

```
#include "dbconn_mysql_imp.h"
#include "dbconn_mysql_result.h"
```

Include dependency graph for dbconn_mysql_imp.cpp:




### Functions

- static TableRow get_field_names (MySQLResult &result)

  *Gets field names from a MySQL result structure.*
- static TableRow get_row (MySQLResult &result, MYSQL_ROW row)

  *Creates a TableRow from a MySQL result row.*

### 10.19.1 Detailed Description

Implementation of MySQL database connection implementation class.

**Author**

Paul Griffiths

**Copyright**

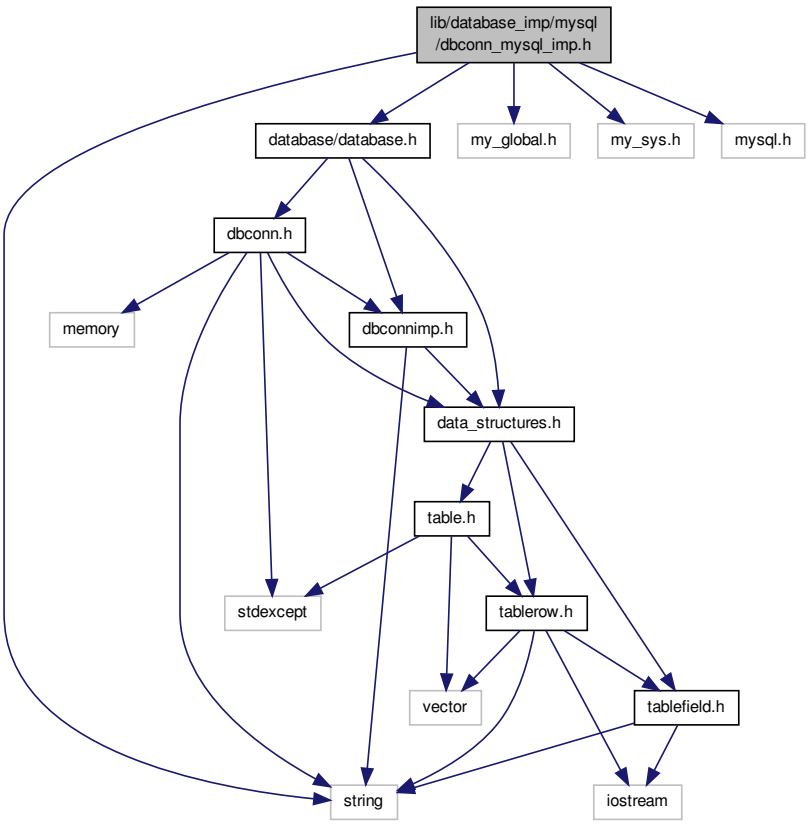
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.20 lib/database_imp/mysql/dbconn_mysql_imp.h File Reference

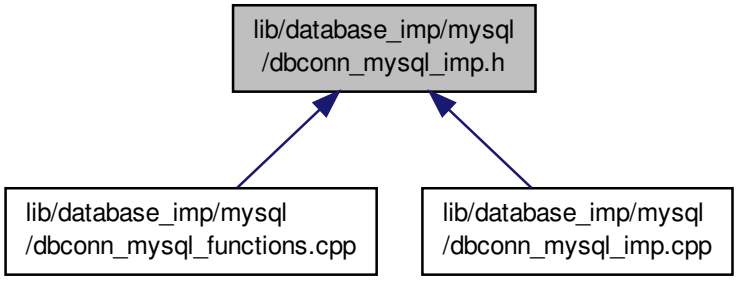
Interface to MySQL database connection implementation class.

```
#include <string>
#include "database/database.h"
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
```

Include dependency graph for dbconn_mysql_imp.h:



This graph shows which files directly or indirectly include this file:




## Classes

- class gldb::DBConnMySQL

  *MySQL database implementation class.*

### 10.20.1 Detailed Description

Interface to MySQL database connection implementation class.

**Author**

Paul Griffiths

**Copyright**

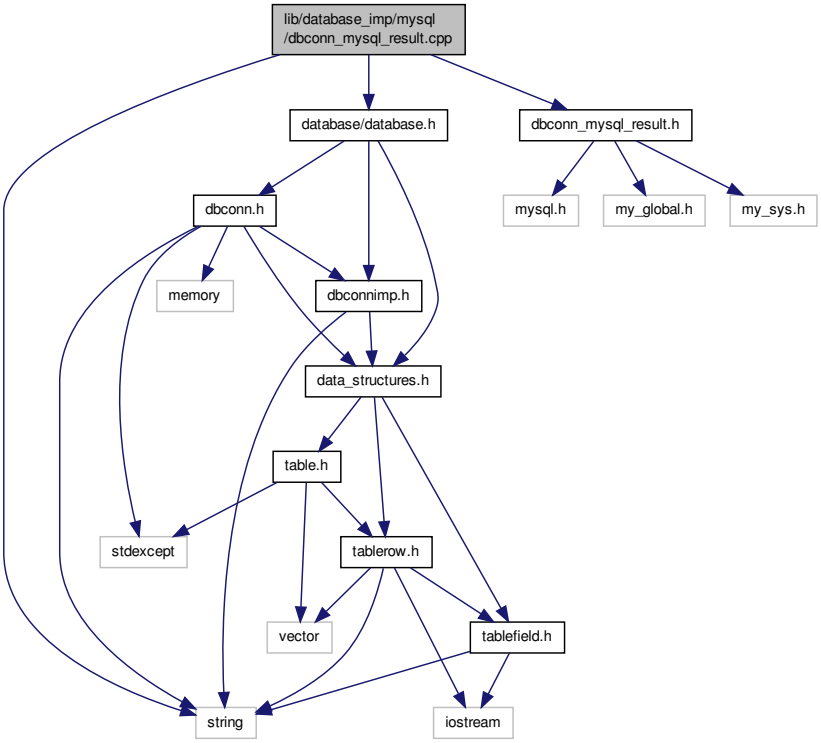
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.21 lib/database_imp/mysql/dbconn_mysql_result.cpp File Reference

Implementation of MySQL result structure resource handle class.

```
#include <string>
#include "database/database.h"
#include "dbconn_mysql_result.h"
```

Include dependency graph for dbconn_mysql_result.cpp:




### 10.21.1 Detailed Description

Implementation of MySQL result structure resource handle class.

**Author**

Paul Griffiths

**Copyright**

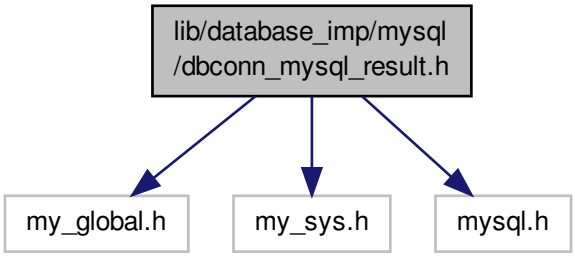
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.22 lib/database_imp/mysql/dbconn_mysql_result.h File Reference

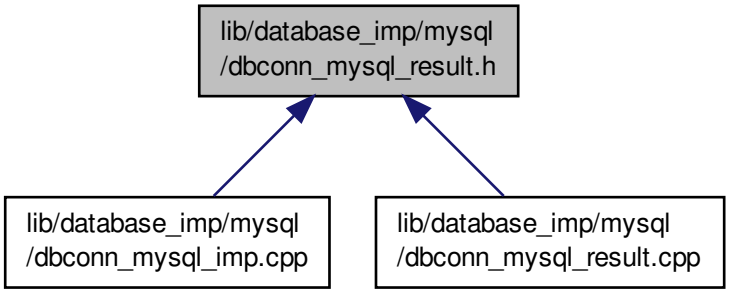
Interface to MySQL result structure resource handle class.

```
#include <my_global.h>
#include <my_sys.h>
#include <mysql.h>
```

Include dependency graph for dbconn_mysql_result.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class gldb::MySQLResult

  *MySQL result structure class.*

### 10.22.1 Detailed Description

Interface to MySQL result structure resource handle class.

**Author**

Paul Griffiths

**Copyright**

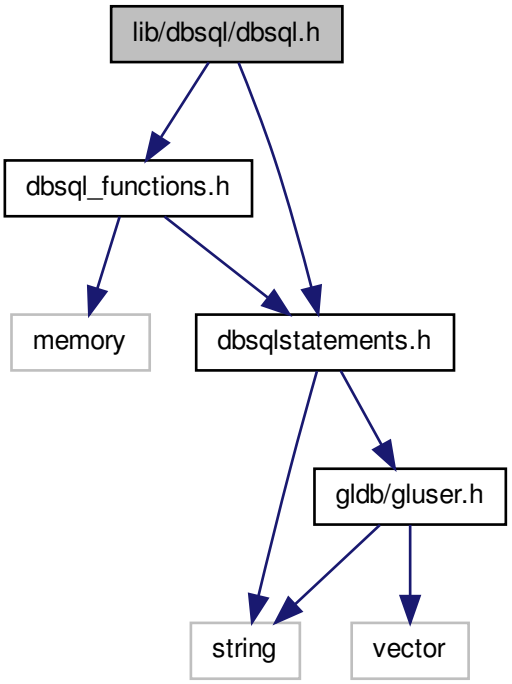
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.23 lib/dbsql/dbsql.h File Reference

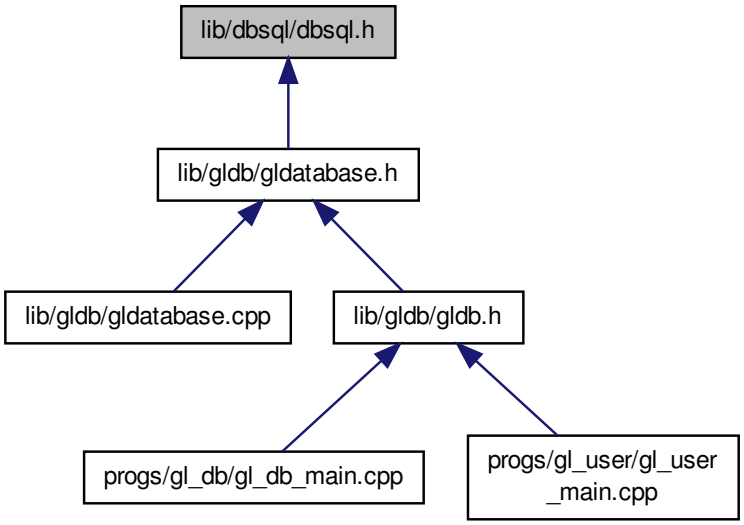
User interface to DBSQL module.

```
#include "dbsql_functions.h"
#include "dbsqlstatements.h"
```

Include dependency graph for dbsql.h:



This graph shows which files directly or indirectly include this file:




### 10.23.1 Detailed Description

User interface to DBSQL module.

**Author**

Paul Griffiths

**Copyright**

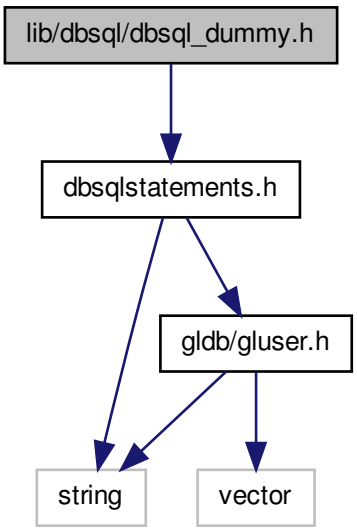
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.24 lib/dbsql/dbsql_dummy.h File Reference

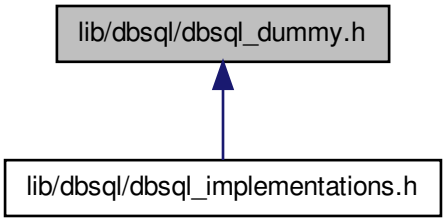
Interface to dummy SQL statement class.

```
#include "dbsqlstatements.h"
```

Include dependency graph for dbsql_dummy.h:




This graph shows which files directly or indirectly include this file:




## Classes

- class genleg::DBSQLDummy

  *Dummy SQL statements class.*

### 10.24.1 Detailed Description

Interface to dummy SQL statement class.

**Author**

Paul Griffiths

**Copyright**

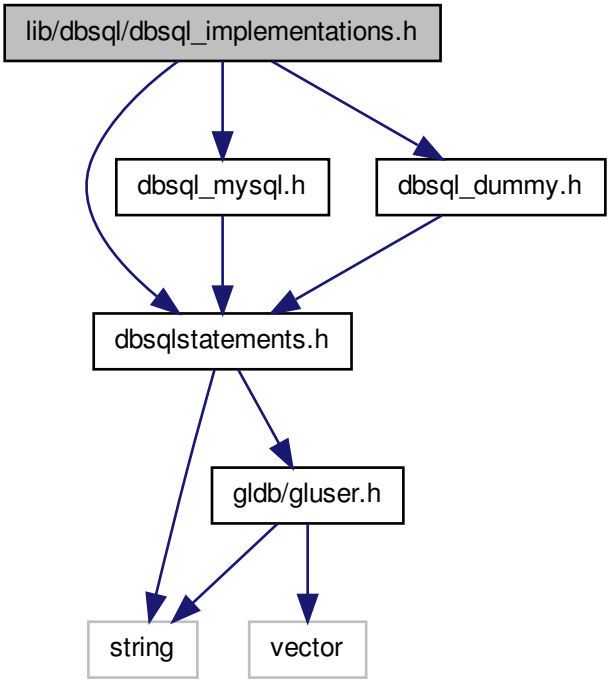
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.25 lib/dbsql/dbsql_implementations.h File Reference

Aggregation header for DBSqlStatements implementations.

```
#include "dbsqlstatements.h"
#include "dbsql_mysql.h"
#include "dbsql_dummy.h"
```

Include dependency graph for dbsql_implementations.h:




### 10.25.1 Detailed Description

Aggregation header for DBSqlStatements implementations.

**Author**

Paul Griffiths

**Copyright**

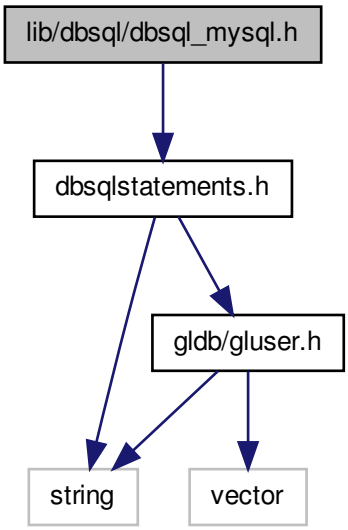
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.26 lib/dbsql/dbsql_mysql.h File Reference

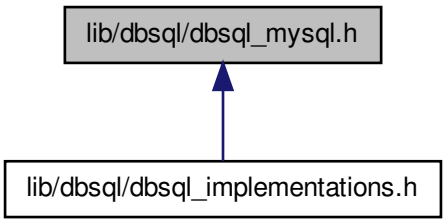
Interface to MySQL SQL statement class.

```
#include "dbsqlstatements.h"
```

Include dependency graph for dbsql_mysql.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class genleg::DBSQLMySQL

*MySQL SQL statements class.*

### 10.26.1 Detailed Description

Interface to MySQL SQL statement class.

**Author**

Paul Griffiths

**Copyright**

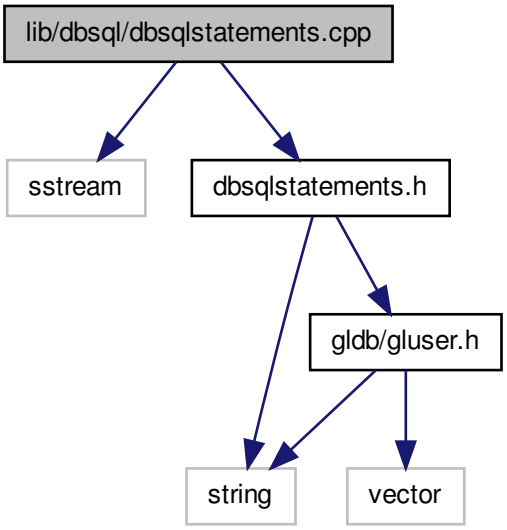
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.27 lib/dbsql/dbsqlstatements.cpp File Reference

Implementation of SQL statement class.

```
#include <sstream>
#include "dbsqlstatements.h"
```

Include dependency graph for dbsqlstatements.cpp:




### 10.27.1 Detailed Description

Implementation of SQL statement class.

**Author**

Paul Griffiths

### Copyright

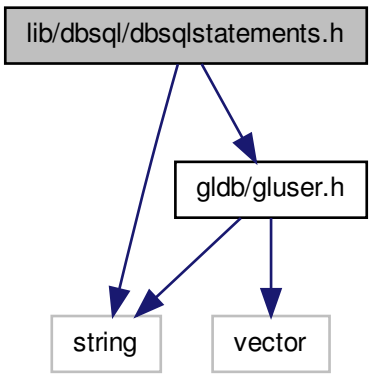
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.28 lib/dbsql/dbsqlstatements.h File Reference

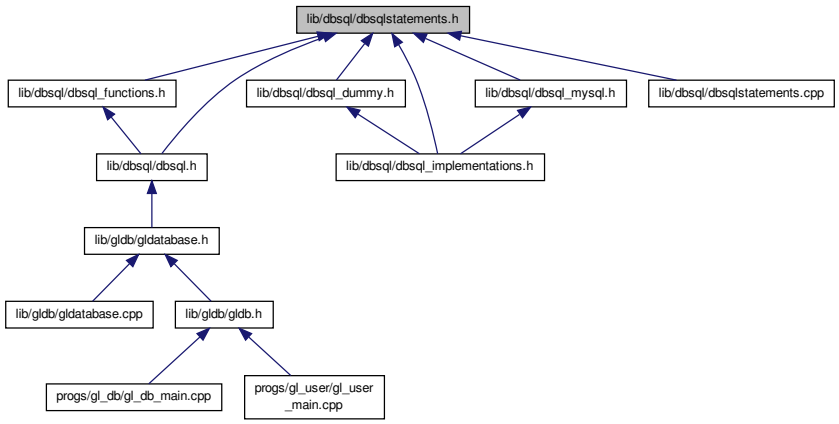
Implementation of SQL module standalone functions.

```
#include <string>
#include "gldb/gluser.h"
```

Include dependency graph for dbsqlstatements.h:




This graph shows which files directly or indirectly include this file:




### Classes

- class genleg::DBSQLStatements

*SQL statements class.*

### 10.28.1 Detailed Description

Implementation of SQL module standalone functions. Interface to SQL statements class.

Interface to SQL module standalone functions.

#### Author

Paul Griffiths

#### Copyright

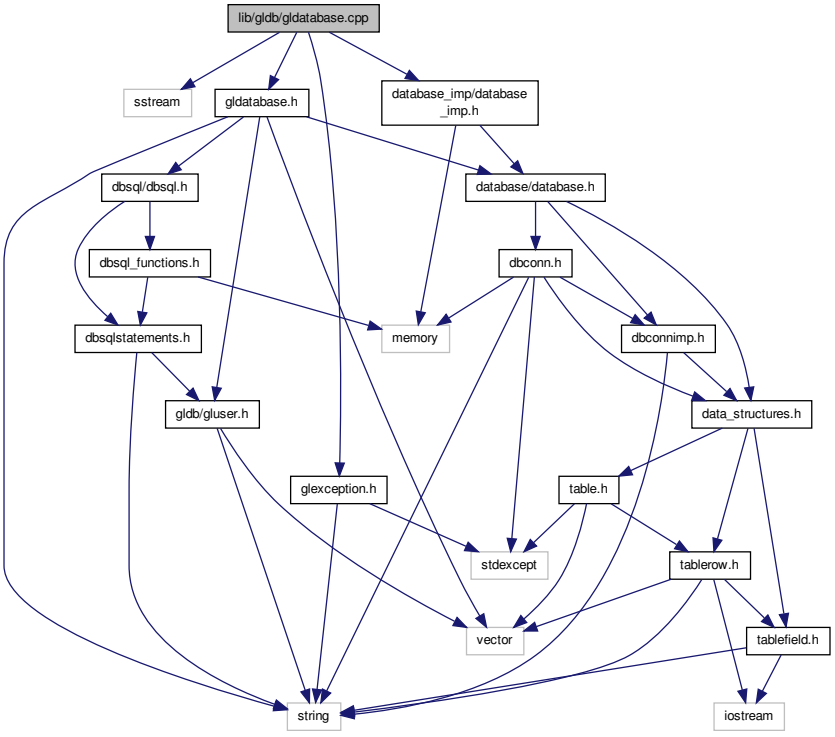
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.29 lib/gldb/gldatabase.cpp File Reference

Implementation of General Ledger database class.

```
#include <sstream>
#include "gldatabase.h"
#include "glexception.h"
#include "database_imp/database_imp.h"
```

Include dependency graph for gldatabase.cpp:



### Functions

- static bool boolstring_to_bool (const std::string &bs)

  *Converts a string representation of a bool to a bool.*
- **m_views** ({"current_trial_balance","check_total","all_jes"})

### 10.29.1 Detailed Description

Implementation of General Ledger database class.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

### 10.29.2 Function Documentation

#### 10.29.2.1 static bool boolstring_to_bool ( const std::string & *bs* ) [static]

Converts a string representation of a bool to a bool.

#### Parameters

| | |
|---|---|
| *bs* | The bool string. |

#### Returns

`true` if `bs` contains "1" or "TRUE", `false` if `bs` contains "0" or "FALSE".

#### Exceptions

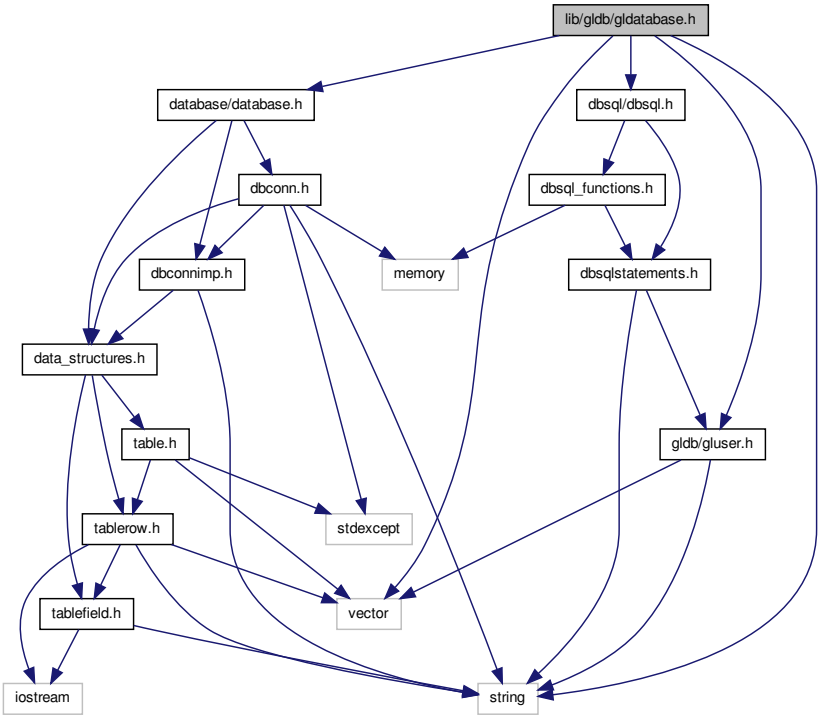
| | |
|---|---|
| *GLDBException* | if `bs` contains any other value. |

## 10.30 lib/gldb/gldatabase.h File Reference

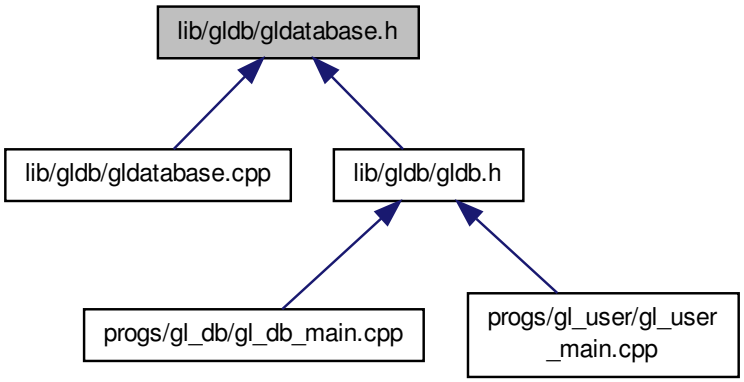
Interface to General Ledger database class.

```
#include <vector>
#include <string>
#include "database/database.h"
#include "dbsql/dbsql.h"
#include "gluser.h"
```

Include dependency graph for gldatabase.h:




This graph shows which files directly or indirectly include this file:




## Classes

- class genleg::GLDatabase

  *General ledger database class.*

### 10.30.1 Detailed Description

Interface to General Ledger database class.

**Author**

Paul Griffiths

**Copyright**

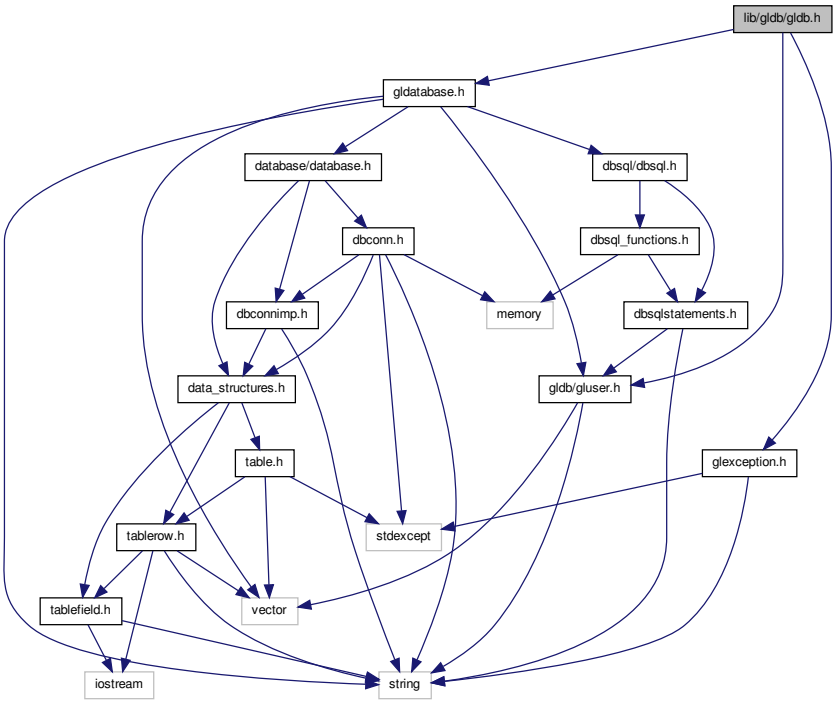
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.31 lib/gldb/gldb.h File Reference

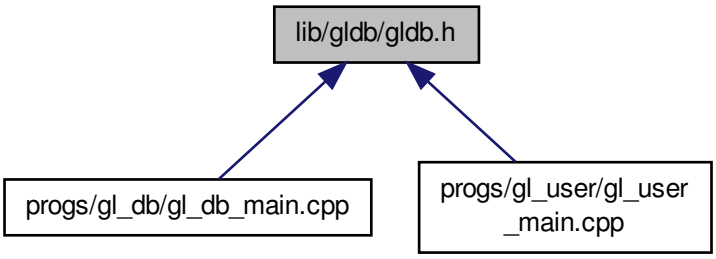
User interface to General Ledger database module.

```
#include "glexception.h"
#include "gldatabase.h"
#include "gluser.h"
```

Include dependency graph for gldb.h:



This graph shows which files directly or indirectly include this file:




### 10.31.1 Detailed Description

User interface to General Ledger database module.

**Author**

Paul Griffiths

**Copyright**

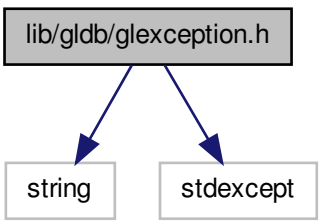
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.32 lib/gldb/glexception.h File Reference

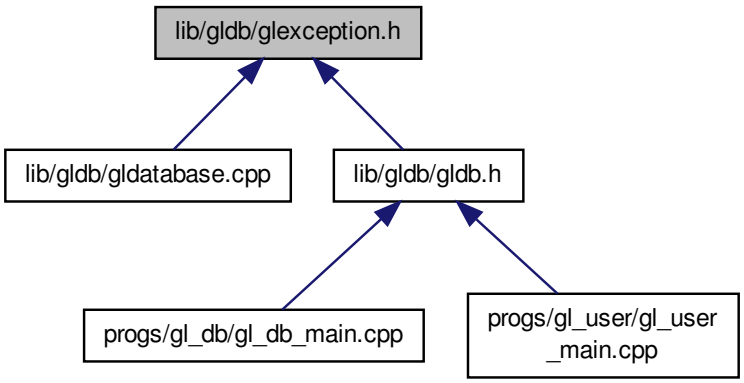
Interface to General Ledger base exception class.

```
#include <string>
#include <stdexcept>
```

Include dependency graph for glexception.h:



This graph shows which files directly or indirectly include this file:




## Classes

- class genleg::GLDBException

  *Base general ledger database exceptionc class.*

### 10.32.1 Detailed Description

Interface to General Ledger base exception class.

**Author**

Paul Griffiths

**Copyright**

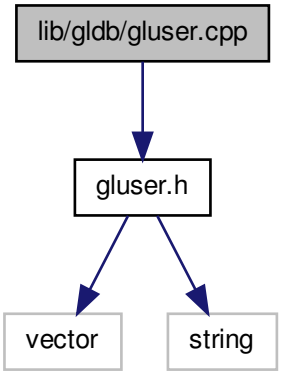
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.33 lib/gldb/gluser.cpp File Reference

Implementation of user class.

```
#include "gluser.h"
```

Include dependency graph for gluser.cpp:




### 10.33.1 Detailed Description

Implementation of user class.

**Author**

Paul Griffiths

**Copyright**

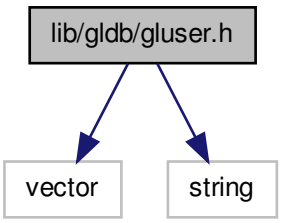
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.34 lib/gldb/gluser.h File Reference

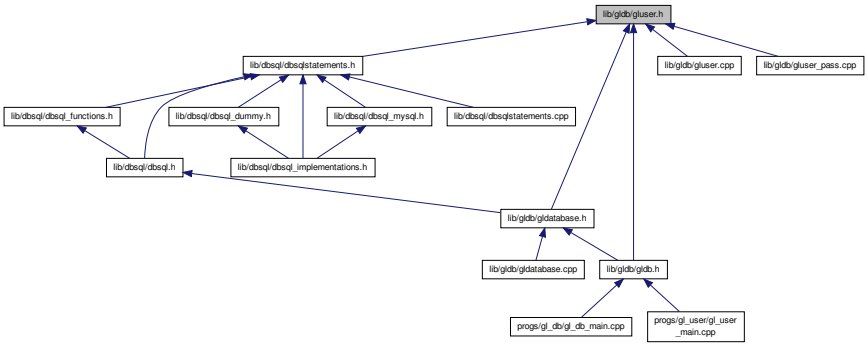
Interface to user class.

```
#include <vector>
#include <string>
```

Include dependency graph for gluser.h:




This graph shows which files directly or indirectly include this file:




## Classes

- class genleg::GLUser

  *General ledger user class.*

### 10.34.1 Detailed Description

Interface to user class.

#### Author

Paul Griffiths

#### Copyright

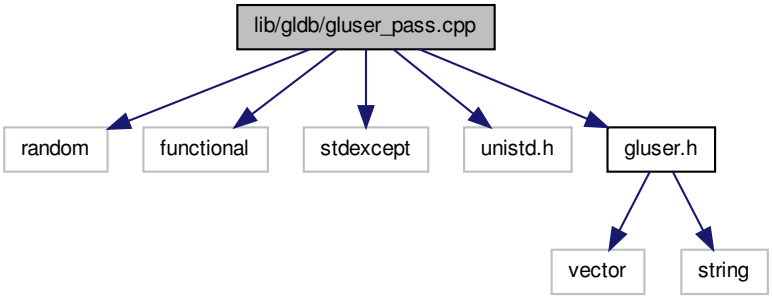
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.35 lib/gldb/gluser_pass.cpp File Reference

Implementation of password functions for user class.

```
#include <random>
#include <functional>
#include <stdexcept>
#include <unistd.h>
#include "gluser.h"
```

Include dependency graph for gluser_pass.cpp:




## Macros

- #define _XOPEN_SOURCE 600

## Functions

- static std::string generate_salt ()

  *Generates a random two-character salt for crypt()*

### 10.35.1 Detailed Description

Implementation of password functions for user class.

**Todo** Implement a better form of password encryption. In particular, these functions are not re-entrant, and only use the first 8 characters of the password.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

### 10.35.2 Macro Definition Documentation

#### 10.35.2.1 #define _XOPEN_SOURCE 600

UNIX feature test macro

### 10.35.3 Function Documentation

#### 10.35.3.1 static std::string generate_salt ( ) [static]

Generates a random two-character salt for crypt()

**Returns**

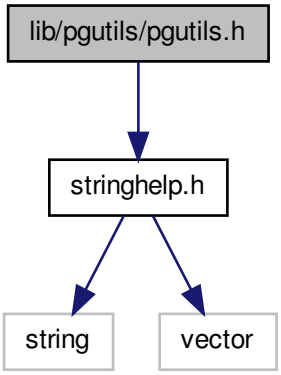
The two-character salt.

## 10.36 lib/pgutils/pgutils.h File Reference

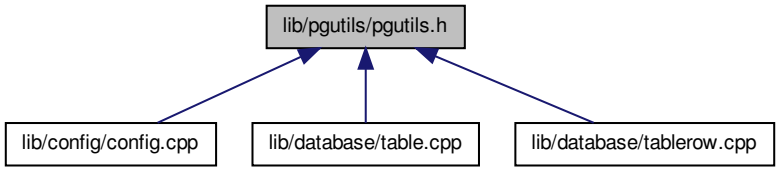
Aggregate interface to general utility functions.

```
#include "stringhelp.h"
```

Include dependency graph for pgutils.h:




This graph shows which files directly or indirectly include this file:




### 10.36.1 Detailed Description

Aggregate interface to general utility functions.

**Author**

Paul Griffiths

**Copyright**

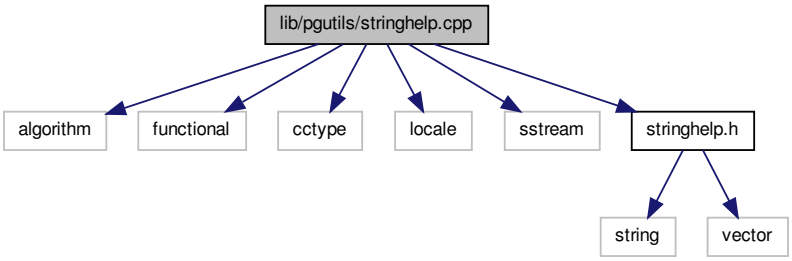
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.37 lib/pgutils/stringhelp.cpp File Reference

Implementation of string helper functions.

```
#include <algorithm>
#include <functional>
#include <cctype>
#include <locale>
#include <sstream>
#include "stringhelp.h"
```

Include dependency graph for stringhelp.cpp:




### 10.37.1 Detailed Description

Implementation of string helper functions.

**Author**

Paul Griffiths

**Copyright**

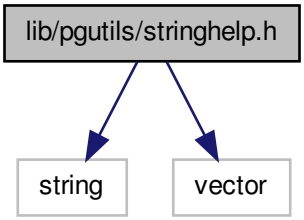
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.38 lib/pgutils/stringhelp.h File Reference

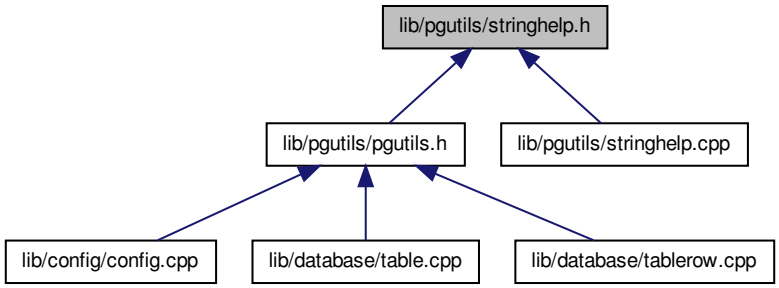
Interface to string helper functions.

```
#include <string>
#include <vector>
```

Include dependency graph for stringhelp.h:




This graph shows which files directly or indirectly include this file:




## Functions

- std::string & pgutils::trim_front (std::string &s)

  *Trims leading whitespace from a string.*
- std::string & pgutils::trim_back (std::string &s)

  *Trims trailing whitespace from a string.*
- std::string & pgutils::trim (std::string &s)

  *Trims leading and trailing whitespace from a string.*
- std::vector< std::string > pgutils::split (const std::string &s, const char delim)

  *Splits a delimited string into tokens.*
- std::vector< std::string > & pgutils::split (std::vector< std::string > &vec, const std::string &s, const char delim)

  *Splits a delimited string into tokens.*
- bool pgutils::next_content_line (std::istream &ifs, std::string &s)

  *Gets the next content line from a stream.*
- std::vector< std::string > & pgutils::content_lines (std::vector< std::string > &vec, std::istream &ifs)

  *Populates a vector of content lines from a stream.*
- std::vector< std::vector < std::string > > & pgutils::split_lines (std::vector< std::vector< std::string >> &vec, std::istream &ifs, const char delim)

*Populates a vector of vectors of fields from a stream.*

- std::string & pgutils::join (const std::vector< std::string > &vec, std::string &s, const char delim)

  *Joins a vector of strings into a delimited line.*

- bool pgutils::replace (std::string &str, const std::string &from, const std::string &to)

  *Replaces a substring with another string.*

### 10.38.1 Detailed Description

Interface to string helper functions.

#### Author

Paul Griffiths

#### Copyright

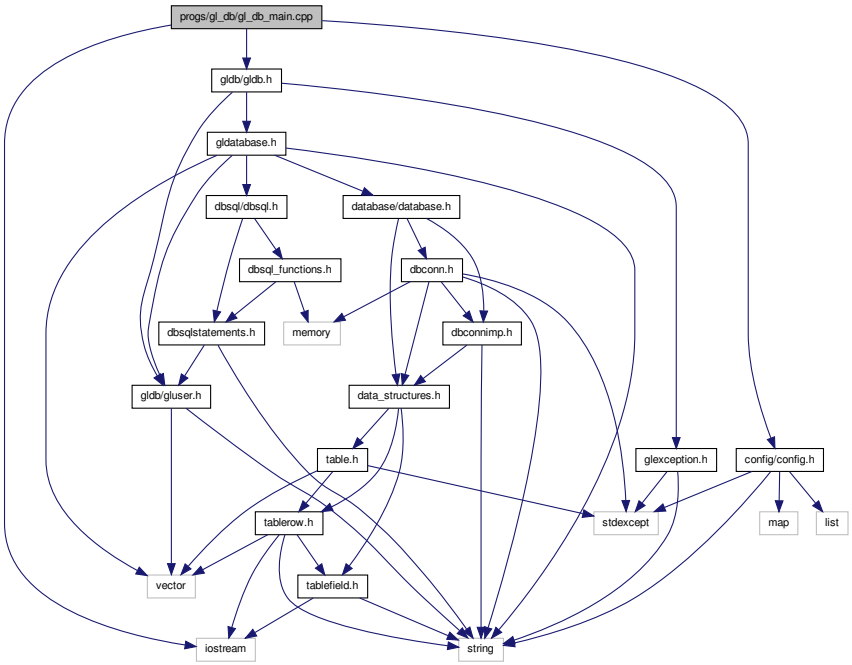
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.39 progs/gl_db/gl_db_main.cpp File Reference

Main functionality for gl_db program.

```
#include <iostream>
#include "gldb/gldb.h"
#include "config/config.h"
```

Include dependency graph for gl_db_main.cpp:



### Functions

- static void set_configuration (Config &config, int argc, char ∗argv[])

  *Sets program configuration options.*
- static bool check_help_and_version (const Config &config)

  *Prints help or version messages if requested.*
- static bool check_db_parameters (const Config &config)

  *Checks if database, hostname and username were provided.*
- static void print_usage_message ()

  *Prints a program usage message.*
- static void print_version_message ()

  *Prints a program version message.*
- static void print_help_message ()

  *Prints a program help message.*
- static std::string login (void)

  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])

  *Main function.*

### Variables

- static const char ∗ progname = "gl_db"

  *Static variable for program name.*

#### 10.39.1 Detailed Description

Main functionality for gl_db program.

**Author**

Paul Griffiths

**Copyright**

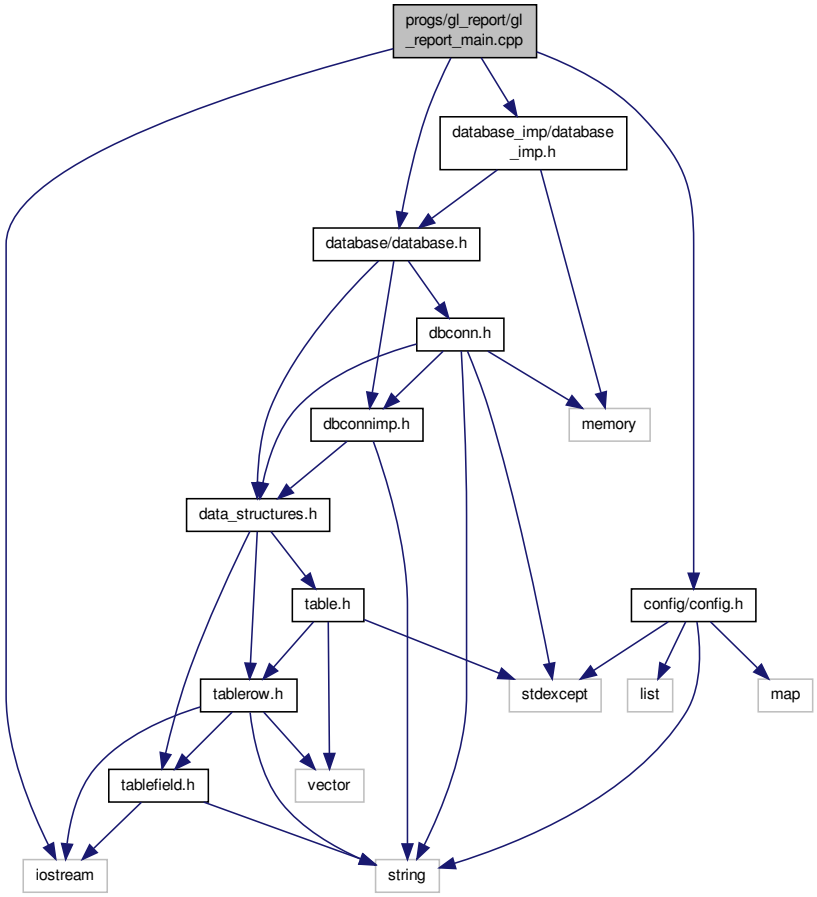
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.40 progs/gl_report/gl_report_main.cpp File Reference

Main functionality for gl_report program.

```
#include <iostream>
#include "database/database.h"
#include "database_imp/database_imp.h"
#include "config/config.h"
```

Include dependency graph for gl_report_main.cpp:




## Functions

- static void set_configuration (genleg::Config &config, int argc, char ∗argv[])

  *Sets program configuration options.*
- static void print_usage_message ()

  *Prints a program usage message.*
- static void print_version_message ()

  *Prints a program version message.*
- static void print_help_message ()

  *Prints a program help message.*
- static std::string login (void)

  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])

  *Main function.*

## Variables

- static const char ∗ progname = "gl_report"

  *Static variable for program name.*

### 10.40.1 Detailed Description

Main functionality for gl_report program.

#### Author

Paul Griffiths

#### Copyright

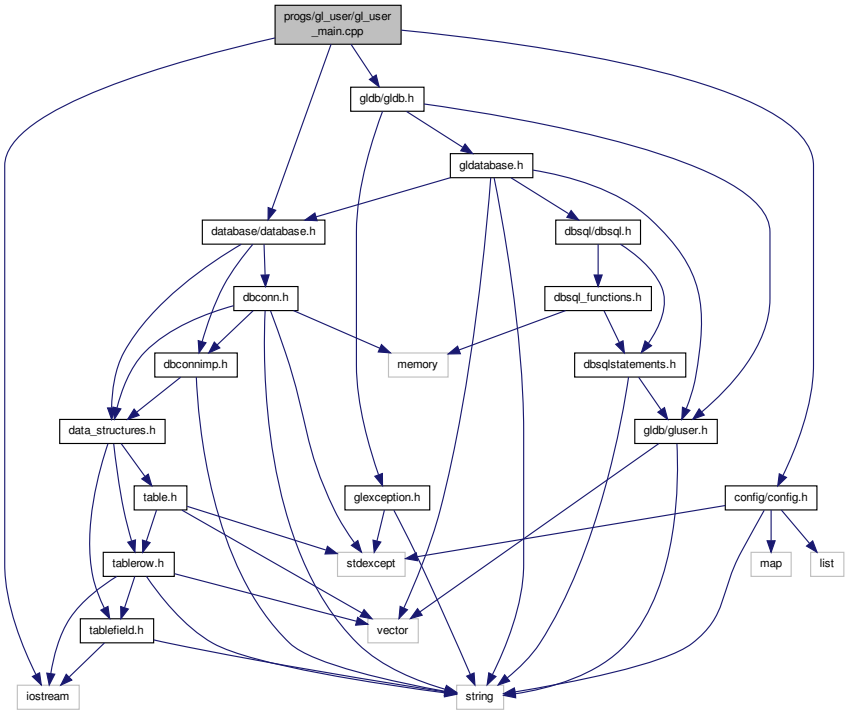
Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

## 10.41 progs/gl_user/gl_user_main.cpp File Reference

Main functionality for gl_user program.

```
#include <iostream>
#include "gldb/gldb.h"
#include "database/database.h"
#include "config/config.h"
```

Include dependency graph for gl_user_main.cpp:




### Functions

- static void set_configuration (Config &config, int argc, char *argv[])

  *Sets program configuration options.*
- static bool check_help_and_version (const Config &config)

  *Prints help or version messages if requested.*

- static bool check_db_parameters (const Config &config)

  *Checks if database, hostname and username were provided.*
- GLUser get_user (Config &config, GLDatabase &gdb)

  *Returns a user from either an ID or a name.*
- static void show_user_details (const GLUser &user)

  *Outputs details for a user.*
- static void enable_user (GLUser &user, Config &config, GLDatabase &gdb)

  *Enables or disables a user.*
- static void set_user_password (GLUser &user, Config &config, GLDatabase &gdb)

  *Sets a user's password.*
- static void check_user_password (GLUser &user, Config &config)

  *Checks a user's password.*
- static void print_usage_message ()

  *Prints a program usage message.*
- static void print_version_message ()

  *Prints a program version message.*
- static void print_help_message ()

  *Prints a program help message.*
- static std::string login (void)

  *Gets a password from the terminal.*
- int main (int argc, char ∗argv[])

  *Main function.*

## Variables

- static const char ∗ progname = "gl_user"

  *Static variable for program name.*

### 10.41.1 Detailed Description

Main functionality for gl_user program.

#### Author

Paul Griffiths

#### Copyright

Copyright 2014 Paul Griffiths. Distributed under the terms of the GNU General Public License. http://www.gnu.org/licenses/

# Index


~Config
genleg::Config, 33
~DBConnDummy
gldb::DBConnDummy, 47
~DBConnImp
gldb::DBConnImp, 49
~DBConnMySQL
gldb::DBConnMySQL, 52
~DBSQLStatements
genleg::DBSQLStatements, 56
~GLDatabase
genleg::GLDatabase, 60
~GLUser
genleg::GLUser, 65
~MySQLResult
gldb::MySQLResult, 69
~Table
gldb::Table, 72
~TableField
gldb::TableField, 82
~TableRow
gldb::TableRow, 90
_XOPEN_SOURCE
config_getopt.cpp, 98
gluser_pass.cpp, 138

add_cmdline_option
genleg::Config, 34
append_field
gldb::TableRow, 90, 91
append_record
gldb::Table, 73

backend
genleg::GLDatabase, 60
begin
gldb::TableRow, 91
boolstring_to_bool
gldatabase.cpp, 131

check_db_parameters
Database program., 25
User administration program., 29
check_help_and_version
Database program., 25
User administration program., 30
check_password
genleg::GLUser, 65
check_user_password
User administration program., 30

Config
genleg::Config, 33
config_getopt.cpp
_XOPEN_SOURCE, 98
ConfigBadConfigFile
genleg::ConfigBadConfigFile, 36
ConfigBadOption
genleg::ConfigBadOption, 37
ConfigCouldNotOpenFile
genleg::ConfigCouldNotOpenFile, 39
ConfigException
genleg::ConfigException, 40
ConfigOptionNotSet
genleg::ConfigOptionNotSet, 41
content_lines
General purpose utilities., 21
create_from_file
gldb::Table, 73
create_structure
genleg::GLDatabase, 60
create_table
genleg::DBSQLStatements, 56
create_user
genleg::GLDatabase, 61
create_view
genleg::DBSQLStatements, 56

DBConn
gldb::DBConn, 42
DBConnCouldNotConnect
gldb::DBConnCouldNotConnect, 44
DBConnCouldNotQuery
gldb::DBConnCouldNotQuery, 45
DBConnDummy
gldb::DBConnDummy, 47
DBConnException
gldb::DBConnException, 48
DBConnImp
gldb::DBConnImp, 49
DBConnMySQL
gldb::DBConnMySQL, 51, 52
DBSQLStatements
genleg::DBSQLStatements, 56
Database interaction module, 16
get_connection, 17
get_database_type, 17
get_field_names, 17
get_row, 17
Database program., 25
check_db_parameters, 25

check_help_and_version, 25
login, 26
main, 26
set_configuration, 26
destroy_structure
genleg::GLDatabase, 61
drop_table
genleg::DBSQLStatements, 56
drop_view
genleg::DBSQLStatements, 56

enable_user
User administration program., 30
enabled
genleg::GLUser, 66
end
gldb::TableRow, 91, 92

firstname
genleg::GLUser, 66

GLDBException
genleg::GLDBException, 63
GLDatabase
genleg::GLDatabase, 60
GLUser
genleg::GLUser, 65
General Ledger database module., 20
General purpose utilities., 21
content_lines, 21
join, 21
next_content_line, 22
replace, 22
split, 22
split_lines, 23
trim, 23
trim_back, 23
trim_front, 23
generate_salt
gluser_pass.cpp, 139
genleg::Config, 33
∼Config, 33
add_cmdline_option, 34
Config, 33
is_set, 34
m_opts_set, 35
m_opts_supp, 35
populate_from_cmdline, 34
populate_from_file, 35
genleg::ConfigBadConfigFile, 35
ConfigBadConfigFile, 36
genleg::ConfigBadOption, 37
ConfigBadOption, 37
genleg::ConfigCouldNotOpenFile, 38
ConfigCouldNotOpenFile, 39
genleg::ConfigException, 39
ConfigException, 40
genleg::ConfigOptionNotSet, 40
ConfigOptionNotSet, 41
genleg::DBSQLDummy, 53
genleg::DBSQLMySQL, 54
genleg::DBSQLStatements, 55
∼DBSQLStatements, 56
create_table, 56
create_view, 56
DBSQLStatements, 56
drop_table, 56
drop_view, 56
get_perms, 57
grant, 57
revoke, 57
update_user, 58
user_by_id, 58
user_by_username, 58
genleg::GLDBException, 63
GLDBException, 63
genleg::GLDatabase, 58
∼GLDatabase, 60
backend, 60
create_structure, 60
create_user, 61
destroy_structure, 61
GLDatabase, 60
get_user_by_id, 61
get_user_by_username, 61
grant, 62
load_sample_data, 62
m_dbc, 62
m_sql, 63
m_tables, 63
m_views, 63
revoke, 62
update_user, 62
genleg::GLUser, 64
∼GLUser, 65
check_password, 65
enabled, 66
firstname, 66
GLUser, 65
id, 66
lastname, 66
m_enabled, 68
m_firstname, 68
m_id, 68
m_lastname, 68
m_pass_hash, 68
m_pass_salt, 68
m_perms, 68
m_username, 68
pass_hash, 66
pass_salt, 66
permissions, 67
set_enabled, 67
set_firstname, 67
set_lastname, 67
set_password, 67
set_username, 67

username, 68
get_connection
Database interaction module, 17
get_database_type
Database interaction module, 17
get_field
gldb::Table, 73
get_field_names
Database interaction module, 17
get_headers
gldb::Table, 74
get_perms
genleg::DBSQLStatements, 57
get_row
Database interaction module, 17
get_user
User administration program., 30
get_user_by_id
genleg::GLDatabase, 61
get_user_by_username
genleg::GLDatabase, 61
gldatabase.cpp
boolstring_to_bool, 131
gldb::DBConn, 41
DBConn, 42
m_imp, 43
operator=, 42
query, 42
select, 43
gldb::DBConnCouldNotConnect, 43
DBConnCouldNotConnect, 44
gldb::DBConnCouldNotQuery, 44
DBConnCouldNotQuery, 45
gldb::DBConnDummy, 46
∼DBConnDummy, 47
DBConnDummy, 47
operator=, 47
query, 47
select, 47
gldb::DBConnException, 48
DBConnException, 48
gldb::DBConnImp, 49
∼DBConnImp, 49
DBConnImp, 49
query, 50
select, 50
gldb::DBConnMySQL, 50
∼DBConnMySQL, 52
DBConnMySQL, 51, 52
m_conn, 53
operator=, 52
query, 52
select, 52
gldb::MySQLResult, 69
∼MySQLResult, 69
m_num_fields, 70
m_result, 70
MySQLResult, 69
num_fields, 70
operator=, 70
result, 70
gldb::Table, 70
∼Table, 72
append_record, 73
create_from_file, 73
get_field, 73
get_headers, 74
insert_query, 74
m_headers, 75
m_quoted, 75
m_records, 76
num_fields, 74
num_records, 74
operator=, 74, 75
set_quoted, 75
Table, 72
gldb::TableBadInputFile, 76
TableBadInputFile, 77
gldb::TableCouldNotOpenInputFile, 77
TableCouldNotOpenInputFile, 78
gldb::TableException, 78
TableException, 79
gldb::TableField, 79
∼TableField, 82
length, 82
m_data, 84
operator std::string, 82
operator<<, 84
operator+=, 82
operator=, 82, 83
TableField, 81
gldb::TableMismatchedRecordLength, 85
TableMismatchedRecordLength, 85
gldb::TableNoSuchField, 86
TableNoSuchField, 87
gldb::TableNoSuchRecord, 87
TableNoSuchRecord, 88
gldb::TableRow, 88
∼TableRow, 90
append_field, 90, 91
begin, 91
end, 91, 92
m_fields, 93
operator=, 92
print, 93
record_string, 93
size, 93
TableRow, 89, 90
gluser_pass.cpp
_XOPEN_SOURCE, 138
generate_salt, 139
grant
genleg::DBSQLStatements, 57
genleg::GLDatabase, 62

id
genleg::GLUser, 66

insert_query
gldb::Table, 74
is_set
genleg::Config, 34

join
General purpose utilities., 21

lastname
genleg::GLUser, 66
length
gldb::TableField, 82
lib/config/config.cpp, 95
lib/config/config.h, 96
lib/config/config_getopt.cpp, 97
lib/database/data_structures.h, 98
lib/database/database.h, 99
lib/database/dbconn.cpp, 101
lib/database/dbconn.h, 102
lib/database/dbconnimp.h, 103
lib/database/table.cpp, 105
lib/database/table.h, 106
lib/database/tablefield.cpp, 107
lib/database/tablefield.h, 108
lib/database/tablerow.cpp, 109
lib/database/tablerow.h, 110
lib/database_imp/database_imp.h, 112
lib/database_imp/dummy/dbconn_dummy_imp.cpp, 114
lib/database_imp/dummy/dbconn_dummy_imp.h, 115
lib/database_imp/mysql/dbconn_mysql_functions.cpp, 117
lib/database_imp/mysql/dbconn_mysql_imp.cpp, 118
lib/database_imp/mysql/dbconn_mysql_imp.h, 119
lib/database_imp/mysql/dbconn_mysql_result.cpp, 121
lib/database_imp/mysql/dbconn_mysql_result.h, 122
lib/dbsql/dbsql.h, 123
lib/dbsql/dbsql_dummy.h, 124
lib/dbsql/dbsql_implementations.h, 126
lib/dbsql/dbsql_mysql.h, 127
lib/dbsql/dbsqlstatements.cpp, 128
lib/dbsql/dbsqlstatements.h, 129
lib/gldb/gldatabase.cpp, 130
lib/gldb/gldatabase.h, 131
lib/gldb/gldb.h, 133
lib/gldb/glexception.h, 134
lib/gldb/gluser.cpp, 136
lib/gldb/gluser.h, 136
lib/gldb/gluser_pass.cpp, 137
lib/pgutils/pgutils.h, 139
lib/pgutils/stringhelp.cpp, 140
lib/pgutils/stringhelp.h, 140
load_sample_data
genleg::GLDatabase, 62
login
Database program., 26
Reporting program., 27
User administration program., 30

m_conn
gldb::DBConnMySQL, 53
m_data
gldb::TableField, 84
m_dbc
genleg::GLDatabase, 62
m_enabled
genleg::GLUser, 68
m_fields
gldb::TableRow, 93
m_firstname
genleg::GLUser, 68
m_headers
gldb::Table, 75
m_id
genleg::GLUser, 68
m_imp
gldb::DBConn, 43
m_lastname
genleg::GLUser, 68
m_num_fields
gldb::MySQLResult, 70
m_opts_set
genleg::Config, 35
m_opts_supp
genleg::Config, 35
m_pass_hash
genleg::GLUser, 68
m_pass_salt
genleg::GLUser, 68
m_perms
genleg::GLUser, 68
m_quoted
gldb::Table, 75
m_records
gldb::Table, 76
m_result
gldb::MySQLResult, 70
m_sql
genleg::GLDatabase, 63
m_tables
genleg::GLDatabase, 63
m_username
genleg::GLUser, 68
m_views
genleg::GLDatabase, 63
main
Database program., 26
Reporting program., 27
User administration program., 31
MySQLResult
gldb::MySQLResult, 69

next_content_line
General purpose utilities., 22
num_fields
gldb::MySQLResult, 70
gldb::Table, 74
num_records
gldb::Table, 74

operator std::string
gldb::TableField, 82
operator<<
gldb::TableField, 84
operator+=
gldb::TableField, 82
operator=
gldb::DBConn, 42
gldb::DBConnDummy, 47
gldb::DBConnMySQL, 52
gldb::MySQLResult, 70
gldb::Table, 74, 75
gldb::TableField, 82, 83
gldb::TableRow, 92

pass_hash
genleg::GLUser, 66
pass_salt
genleg::GLUser, 66
permissions
genleg::GLUser, 67
populate_from_cmdline
genleg::Config, 34
populate_from_file
genleg::Config, 35
print
gldb::TableRow, 93
Program configuration module, 15
progs/gl_db/gl_db_main.cpp, 142
progs/gl_report/gl_report_main.cpp, 143
progs/gl_user/gl_user_main.cpp, 145

query
gldb::DBConn, 42
gldb::DBConnDummy, 47
gldb::DBConnImp, 50
gldb::DBConnMySQL, 52

record_string
gldb::TableRow, 93
replace
General purpose utilities., 22
Reporting program., 27
login, 27
main, 27
set_configuration, 28
result
gldb::MySQLResult, 70
revoke
genleg::DBSQLStatements, 57
genleg::GLDatabase, 62

SQL statements module, 19
select
gldb::DBConn, 43
gldb::DBConnDummy, 47
gldb::DBConnImp, 50
gldb::DBConnMySQL, 52
set_configuration
Database program., 26
Reporting program., 28
User administration program., 31
set_enabled
genleg::GLUser, 67
set_firstname
genleg::GLUser, 67
set_lastname
genleg::GLUser, 67
set_password
genleg::GLUser, 67
set_quoted
gldb::Table, 75
set_user_password
User administration program., 31
set_username
genleg::GLUser, 67
show_user_details
User administration program., 31
size
gldb::TableRow, 93
split
General purpose utilities., 22
split_lines
General purpose utilities., 23

Table
gldb::Table, 72
TableBadInputFile
gldb::TableBadInputFile, 77
TableCouldNotOpenInputFile
gldb::TableCouldNotOpenInputFile, 78
TableException
gldb::TableException, 79
TableField
gldb::TableField, 81
TableMismatchedRecordLength
gldb::TableMismatchedRecordLength, 85
TableNoSuchField
gldb::TableNoSuchField, 87
TableNoSuchRecord
gldb::TableNoSuchRecord, 88
TableRow
gldb::TableRow, 89, 90
trim
General purpose utilities., 23
trim_back
General purpose utilities., 23
trim_front
General purpose utilities., 23

update_user
genleg::DBSQLStatements, 58
genleg::GLDatabase, 62
User administration program., 29
check_db_parameters, 29
check_help_and_version, 30
check_user_password, 30
enable_user, 30

get_user, 30
login, 30
main, 31
set_configuration, 31
set_user_password, 31
show_user_details, 31
user_by_id
genleg::DBSQLStatements, 58
user_by_username
genleg::DBSQLStatements, 58
username
genleg::GLUser, 68