

SQL Backup to FTP server

Generated by Doxygen 1.9.3

1 SQL Backup to FTP server	1
1.1 Introduction	1
1.2 How to use	1
1.2.1 Step 1: Set up an Filezilla FTP server and create a user in it.	1
1.2.2 Step 2: Run the program by setting options via the console.	1
2 Namespace Index	3
2.1 Namespace List	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Namespace Documentation	9
5.1 ftp Namespace Reference	9
5.1.1 Function Documentation	9
5.1.1.1 getDateTIme()	9
5.2 ftp::dbmgr Namespace Reference	9
5.2.1 Function Documentation	10
5.2.1.1 createTable()	10
5.2.1.2 init()	10
5.2.1.3 insertData()	11
5.2.1.4 removeDb()	11
5.2.1.5 selectData()	11
5.3 ftp::query Namespace Reference	12
5.3.1 Variable Documentation	12
5.3.1.1 createTable	12
5.3.1.2 insertData	12
5.3.1.3 selectData	12
6 Class Documentation	13
6.1 ftp::CmdLineOptionParser Class Reference	13
6.1.1 Detailed Description	13
6.1.2 Member Function Documentation	13
6.1.2.1 addOption()	13
6.1.2.2 checkOption()	14
6.1.2.3 getIntOptionValue()	14
6.1.2.4 getStringOptionValue()	14
6.1.2.5 parseArgs()	14
6.1.2.6 printHelp()	15
6.2 ftp::FTPManager Class Reference	15
6.2.1 Detailed Description	15

6.2.2 Constructor & Destructor Documentation	15
6.2.2.1 FTPManager()	15
6.2.2.2 ~FTPManager()	16
6.2.3 Member Function Documentation	16
6.2.3.1 endSession()	16
6.2.3.2 getErrorMsg()	16
6.2.3.3 startSession()	16
6.2.3.4 transfer()	16
6.3 ftp::Logger Class Reference	17
6.3.1 Detailed Description	17
6.3.2 Member Function Documentation	17
6.3.2.1 error()	17
6.3.2.2 info()	17
6.4 ftp::SQLBackup Class Reference	17
6.4.1 Detailed Description	18
6.4.2 Constructor & Destructor Documentation	18
6.4.2.1 SQLBackup()	18
6.4.3 Member Function Documentation	18
6.4.3.1 start()	18
6.5 ftp::dbmgr::TableData Class Reference	18
6.5.1 Detailed Description	19
6.5.2 Member Function Documentation	19
6.5.2.1 dumpToFile()	19
6.5.2.2 isEmpty()	19
6.5.2.3 print()	19
6.5.3 Member Data Documentation	19
6.5.3.1 data	19
7 File Documentation	21
7.1 CmdLineOptionParser.cpp File Reference	21
7.2 CmdLineOptionParser.h File Reference	21
7.3 CmdLineOptionParser.h	22
7.4 DbManager.cpp File Reference	22
7.5 DbManager.h File Reference	22
7.6 DbManager.h	23
7.7 FTPManager.cpp File Reference	23
7.8 FTPManager.h File Reference	24
7.9 FTPManager.h	24
7.10 Logger.h File Reference	25
7.11 Logger.h	25
7.12 main.cpp File Reference	25
7.12.1 Function Documentation	26

7.12.1.1 main()	26
7.13 QueryCollection.h File Reference	26
7.14 QueryCollection.h	26
7.15 SQLBackup.cpp File Reference	27
7.16 SQLBackup.h File Reference	27
7.17 SQLBackup.h	27
Index	29

Chapter 1

SQL Backup to FTP server

1.1 Introduction

The app was written by using **C++17** standart.

Tests are written by using **Google Test** framework.

The app uses **sqlite3** database and **Filezilla** FTP server.

App command line options:

-ftp_host - Flag to specify the backup FTP server's host.

-ftp_port - Flag to specify the backup FTP server's port.

-ftp_uname - Flag to specify the backup FTP server's User name.

-ftp_pass - Flag to specify the backup FTP server's User password.

-ftp_dir - Flag to specify the backup FTP server's copy directory.

sqlite3 doesn't require a **connection properties** so I didn't add an option to add a db property.

1.2 How to use

1.2.1 Step 1: Set up an Filezilla FTP server and create a user in it.

1.2.2 Step 2: Run the program by setting options via the console.

example: `./app -ftp_host 127.0.0.1 -ftp_port 21 -ftp_uname username -ftp_passwd password -ftp_dir ftpDir`

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

ftp	9
ftp::dbmgr	9
ftp::query	12

Chapter 3

Class Index

3.1 Class List

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

ftp::CmdLineOptionParser	13
ftp::FTPManager	15
ftp::Logger	17
ftp::SQLBackup	17
ftp::dbmgr::TableData	18

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

CmdLineOptionParser.cpp	21
CmdLineOptionParser.h	21
DbManager.cpp	22
DbManager.h	22
FTPManager.cpp	23
FTPManager.h	24
Logger.h	25
main.cpp	25
QueryCollection.h	26
SQLBackup.cpp	27
SQLBackup.h	27

Chapter 5

Namespace Documentation

5.1 ftp Namespace Reference

Namespaces

- namespace **dbmgr**
- namespace **query**

Classes

- class **CmdLineOptionParser**
- class **FTPManager**
- class **Logger**
- class **SQLBackup**

Functions

- std::string **getDateTime** ()

5.1.1 Function Documentation

5.1.1.1 getDateTime()

```
std::string ftp::getDateTime ( )
```

5.2 ftp::dbmgr Namespace Reference

Classes

- class **TableData**

Functions

- bool **init** (const std::string &dbPath, std::string &errorMsg)
- **TableData selectData** (const std::string &query, std::string &errorMsg)
- bool **removeDb** (const std::string &dbPath, std::string &errorMsg)
- bool **insertData** (const std::string &query, std::string &errorMsg)
- bool **createTable** (const std::string &query, std::string &errorMsg)

5.2.1 Function Documentation

5.2.1.1 createTable()

```
bool ftp::dbmgr::createTable (
    const std::string & query,
    std::string & errorMsg )
```

Creates a table in the database according to the given query.

Parameters

<i>errorMsg</i>	if an error occurs then the error text is assigned to this variable
-----------------	---

Returns

true if there is no errors

5.2.1.2 init()

```
bool ftp::dbmgr::init (
    const std::string & dbPath,
    std::string & errorMsg )
```

Creates a database at the given path.

Parameters

<i>dbPath</i>	path to database
<i>errorMsg</i>	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.3 insertData()

```
bool ftp::dbmgr::insertData (
    const std::string & query,
    std::string & errorMsg )
```

Adds data to the database according to the given query.

Parameters

<i>errorMsg</i>	if an error occurs then the error text is assigned to this variable
-----------------	---

Returns

true if there is no errors

5.2.1.4 removeDb()

```
bool ftp::dbmgr::removeDb (
    const std::string & dbPath,
    std::string & errorMsg )
```

Removes a database at the given path.

Parameters

<i>dbPath</i>	path to database
<i>errorMsg</i>	if an error occurs then the error text is assigned to this variable

Returns

true if there is no errors

5.2.1.5 selectData()

```
TableData ftp::dbmgr::selectData (
    const std::string & query,
    std::string & errorMsg )
```

Fetches data from the database

Parameters

<i>errorMsg</i>	if an error occurs then the error text is assigned to this variable
-----------------	---

5.3 ftp::query Namespace Reference

Variables

- const std::string **createTable**
- const std::string **insertData**
- const std::string **selectData** = "SELECT * FROM Person;"

5.3.1 Variable Documentation

5.3.1.1 createTable

```
const std::string ftp::query::createTable
```

Initial value:

```
=  
"CREATE TABLE IF NOT EXISTS Person(  
"ID INT PRIMARY KEY      NOT NULL, "  
"NAME                    VARCHAR NOT NULL, "  
"AGE                    INT      NOT NULL, "  
"ADDRESS                VARCHAR, "  
"SALARY                 REAL );"
```

5.3.1.2 insertData

```
const std::string ftp::query::insertData
```

Initial value:

```
=  
"INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1, 'AAAA', 11, 'Armenia', 10000 ); "  
"INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'BBBB', 22, 'Germany', 20000 ); "  
"INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'CCCC', 33, 'France', 30000 ); "  
"INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'DDDD', 44, 'USA', 40000 ); "  
"INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'EEEE', 55, 'UK', 500000 ); "
```

5.3.1.3 selectData

```
const std::string ftp::query::selectData = "SELECT * FROM Person;"
```

Chapter 6

Class Documentation

6.1 ftp::CmdLineOptionParser Class Reference

```
#include <CmdLineOptionParser.h>
```

Public Member Functions

- void **parseArgs** (int argc, char *argv[])
- bool **checkOption** (const std::string &option) const
- void **addOption** (const std::string &option, const std::string &description)
- int **getIntOptionValue** (const std::string &option) const
- std::string **getStringOptionValue** (const std::string &option) const
- void **printHelp** () const

6.1.1 Detailed Description

Class to parse the application's command line options.

6.1.2 Member Function Documentation

6.1.2.1 addOption()

```
void ftp::CmdLineOptionParser::addOption (
    const std::string & option,
    const std::string & description )
```

Adds an option and its description to the map.

6.1.2.2 checkOption()

```
bool ftp::CmdLineOptionParser::checkOption (
    const std::string & option ) const
```

Checks if the given option is in the map.

Returns

True if option found

6.1.2.3 getIntOptionValue()

```
int ftp::CmdLineOptionParser::getIntOptionValue (
    const std::string & option ) const
```

Returns the value for the given key as an integr.

Exceptions

<i>runtime_exception</i>	when option not found
--------------------------	-----------------------

6.1.2.4 getStringOptionValue()

```
std::string ftp::CmdLineOptionParser::getStringOptionValue (
    const std::string & option ) const
```

Returns the value for the given key as an string.

Exceptions

<i>runtime_exception</i>	when option not found
--------------------------	-----------------------

6.1.2.5 parseArgs()

```
void ftp::CmdLineOptionParser::parseArgs (
    int argc,
    char * argv[] )
```

Parses the arguments and their values and stores them in the map.

Parameters

<i>argc</i>	argument count
<i>argv</i>	argument list

6.1.2.6 printHelp()

```
void ftp::CmdLineOptionParser::printHelp ( ) const
```

Prints help to the console.

6.2 ftp::FTPManager Class Reference

```
#include <FTPManager.h>
```

Public Member Functions

- **FTPManager** (int port, const std::string &host, const std::string &uname, const std::string &passwd, const std::string &dirPath)
- **~FTPManager** ()
- void **endSession** ()
- bool **startSession** ()
- std::string **getErrMsg** () const
- bool **transfer** (const std::string &localfilePath, const std::string &newRemotefile)

6.2.1 Detailed Description

Class to manage file transfer between Application and FTP Server.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 FTPManager()

```
ftp::FTPManager::FTPManager (
    int port,
    const std::string & host,
    const std::string & uname,
    const std::string & passwd,
    const std::string & dirPath )
```

6.2.2.2 ~FTPManager()

```
ftp::FTPManager::~~FTPManager ( )
```

6.2.3 Member Function Documentation

6.2.3.1 endSession()

```
void ftp::FTPManager::endSession ( )
```

Disconnects from the FTP server and from the session.

6.2.3.2 getErrorMsg()

```
std::string ftp::FTPManager::getErrorMsg ( ) const
```

If one of the functions does not work correctly and there will be an error, then each function returns false and assigns the current error to the errorMsg_ variable.

Returns

the current error message

6.2.3.3 startSession()

```
bool ftp::FTPManager::startSession ( )
```

Establishes a connection with the FTP server and creates a directory dirPath _ if not already created and starts a new session.

Returns

true if everything is ok

6.2.3.4 transfer()

```
bool ftp::FTPManager::transfer (
    const std::string & localfilePath,
    const std::string & newRemoteFile )
```

Sends a file to specified FTP Server directory.

Parameters

<i>localfilePath</i>	local absolute path of a file
<i>newremotefile</i>	new name of a file

Returns

true if transfer was successful

6.3 ftp::Logger Class Reference

```
#include <Logger.h>
```

Static Public Member Functions

- static void **error** (std::string_view fileName, std::string_view msg)
- static void **info** (std::string_view fileName, std::string_view msg)

6.3.1 Detailed Description

Siple logger.

6.3.2 Member Function Documentation

6.3.2.1 error()

```
static void ftp::Logger::error (  
    std::string_view fileName,  
    std::string_view msg ) [inline], [static]
```

6.3.2.2 info()

```
static void ftp::Logger::info (  
    std::string_view fileName,  
    std::string_view msg ) [inline], [static]
```

6.4 ftp::SQLBackup Class Reference

```
#include <SQLBackup.h>
```

Public Member Functions

- **SQLBackup** ()
- void **start** (int argc, char *argv[])

6.4.1 Detailed Description

Class to manage the entire workflow in a project.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 SQLBackup()

```
ftp::SQLBackup::SQLBackup ( )
```

6.4.3 Member Function Documentation

6.4.3.1 start()

```
void ftp::SQLBackup::start (
    int argc,
    char * argv[] )
```

Creates data for a SQLite database, and then exports the database data to a file and sends the file to an FTP server.

Parameters

<i>argc</i>	argument count
<i>argv</i>	argument list

6.5 ftp::dbmgr::TableData Class Reference

```
#include <DbManager.h>
```

Public Member Functions

- void **print** ()
- void **dumpToFile** (const std::string &filePath)
- bool **isEmpty** () const

Public Attributes

- `std::vector< std::vector< std::string > >` **data**

6.5.1 Detailed Description

Class container which holds the data that is returned from db.

6.5.2 Member Function Documentation

6.5.2.1 dumpToFile()

```
void ftp::dbmgr::TableData::dumpToFile (
    const std::string & filePath )
```

6.5.2.2 isEmpty()

```
bool ftp::dbmgr::TableData::isEmpty ( ) const
```

6.5.2.3 print()

```
void ftp::dbmgr::TableData::print ( )
```

6.5.3 Member Data Documentation

6.5.3.1 data

```
std::vector<std::vector<std::string> > ftp::dbmgr::TableData::data
```


Chapter 7

File Documentation

7.1 CmdLineOptionParser.cpp File Reference

```
#include "CmdLineOptionParser.h"  
#include "Logger.h"  
#include <iostream>  
#include <string.h>  
#include <filesystem>
```

Namespaces

- namespace **ftp**

7.2 CmdLineOptionParser.h File Reference

```
#include <unordered_map>
```

Classes

- class **ftp::CmdLineOptionParser**

Namespaces

- namespace **ftp**

7.3 CmdLineOptionParser.h

Go to the documentation of this file.

```

1 #ifndef FTP_CMDLINEOPTIONPARSER_H_
2 #define FTP_CMDLINEOPTIONPARSER_H_
3
4 #include <unordered_map>
5
6 namespace ftp {
7
11     class CmdLineOptionParser
12     {
13     public:
14         using HashContainerType = std::unordered_map<std::string, std::string>;
15
22         void parseArgs(int argc, char* argv[]);
23
29         bool checkOption(const std::string& option) const;
30
34         void addOption(const std::string& option, const std::string& description);
35
41         int getIntOptionValue(const std::string& option) const;
42
49         std::string getStringOptionValue(const std::string& option) const;
50
54         void printHelp() const;
55
56     private:
57         HashContainerType optionToValueMap_;
58         HashContainerType optionToDescriptionMap_;
59     };
60
61 } // end namespace ftp
62 #endif // FTP_CMDLINEOPTIONPARSER_H_

```

7.4 DbManager.cpp File Reference

```

#include "DbManager.h"
#include "Logger.h"
#include <fstream>
#include <filesystem>
#include <winsqlite/winsqlite3.h>

```

Namespaces

- namespace **ftp**
- namespace **ftp::dbmgr**

Functions

- bool **ftp::dbmgr::init** (const std::string &dbPath, std::string &errorMsg)
- TableData **ftp::dbmgr::selectData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::removeDb** (const std::string &dbPath, std::string &errorMsg)
- bool **ftp::dbmgr::insertData** (const std::string &query, std::string &errorMsg)
- bool **ftp::dbmgr::createTable** (const std::string &query, std::string &errorMsg)

7.5 DbManager.h File Reference

```

#include <string>
#include <vector>
#include <sqlite3.h>
#include <filesystem>

```

Classes

- class `ftp::dbmgr::TableData`

Namespaces

- namespace `ftp`
- namespace `ftp::dbmgr`

Functions

- bool `ftp::dbmgr::init` (const std::string &dbPath, std::string &errorMsg)
- bool `ftp::dbmgr::insertData` (const std::string &query, std::string &errorMsg)
- bool `ftp::dbmgr::createTable` (const std::string &query, std::string &errorMsg)
- TableData `ftp::dbmgr::selectData` (const std::string &query, std::string &errorMsg)
- bool `ftp::dbmgr::removeDb` (const std::string &dbPath, std::string &errorMsg)

7.6 DbManager.h

Go to the documentation of this file.

```

1 #ifndef FTP_DBMANAGER_H_
2 #define FTP_DBMANAGER_H_
3
4 #include <string>
5 #include <vector>
6 #include <sqlite3.h>
7 #include <filesystem>
8
9 namespace ftp::dbmgr {
10
11     class TableData
12     {
13     public:
14         void print();
15         void dumpToFile(const std::string& filePath);
16         bool isEmpty() const;
17
18         std::vector<std::vector<std::string>> data;
19     };
20
21     bool init(const std::string& dbPath, std::string& errorMsg);
22
23     bool insertData(const std::string& query, std::string& errorMsg);
24
25     bool createTable(const std::string& query, std::string& errorMsg);
26
27     TableData selectData(const std::string& query, std::string& errorMsg);
28
29     bool removeDb(const std::string& dbPath, std::string& errorMsg);
30
31 } // end namespace dbmgr
32 #endif // FTP_LOGGER_H_

```

7.7 FTPManager.cpp File Reference

```

#include "FTPManager.h"
#include "Logger.h"

```

Namespaces

- namespace `ftp`

7.8 FTPManager.h File Reference

```
#include <string>
#include <Windows.h>
#include <WinInet.h>
```

Classes

- class `ftp::FTPManager`

Namespaces

- namespace `ftp`

7.9 FTPManager.h

Go to the documentation of this file.

```
1 #ifndef FTP_FTPMANAGER_H_
2 #define FTP_FTPMANAGER_H_
3
4 #include <string>
5 #include <Windows.h>
6 #include <WinInet.h>
7
8 namespace ftp {
9
10     class FTPManager
11     {
12     public:
13         FTPManager(int port, const std::string& host, const std::string& uname,
14             const std::string& passwd, const std::string& dirPath);
15
16         ~FTPManager();
17
18         void endSession();
19
20         bool startSession();
21
22         std::string getErrorMsg() const;
23
24         bool transfer(const std::string& localfilePath, const std::string& newRemotefile);
25
26     private:
27         bool setCurrentDir(const std::string& path);
28         bool createDirIfNotExists(const std::string& path);
29
30         int port_;
31         std::string host_;
32         std::string uname_;
33         std::string passwd_;
34         std::string dirPath_;
35         std::string errorMsg_;
36
37         HINTERNET session_ = nullptr;
38         HINTERNET internet_ = nullptr;
39     };
40 } // end namespace ftp
41 #endif // FTP_FTPMANAGER_H_
```

7.10 Logger.h File Reference

```
#include <iostream>
#include <string_view>
```

Classes

- class `ftp::Logger`

Namespaces

- namespace `ftp`

7.11 Logger.h

Go to the documentation of this file.

```
1 #ifndef FTP_LOGGER_H_
2 #define FTP_LOGGER_H_
3
4 #include <iostream>
5 #include <string_view>
6
7 namespace ftp {
8
9     class Logger
10     {
11     public:
12         static void error(std::string_view fileName, std::string_view msg)
13         {
14             std::cerr << "ERROR: in " << fileName << ": " << msg << std::endl;
15         }
16
17         static void info(std::string_view fileName, std::string_view msg)
18         {
19             std::cout << "INFO: " << fileName << ": " << msg << std::endl;
20         }
21
22     private:
23         Logger() = delete;
24         ~Logger() = delete;
25         Logger(const Logger&) = delete;
26         void operator=(const Logger&) = delete;
27         Logger(Logger&&) noexcept = delete;
28         Logger& operator=(Logger&&) noexcept = delete;
29     };
30
31 } // end namespace ftp
32 #endif // FTP_LOGGER_H_
```

7.12 main.cpp File Reference

```
#include "SQLBackup.h"
```

Functions

- int `main` (int argc, char *argv[])

7.12.1 Function Documentation

7.12.1.1 main()

```
int main (
    int argc,
    char * argv[] )
```

7.13 QueryCollection.h File Reference

```
#include <string>
```

Namespaces

- namespace **ftp**
- namespace **ftp::query**

Variables

- const std::string **ftp::query::createTable**
- const std::string **ftp::query::insertData**
- const std::string **ftp::query::selectData** = "SELECT * FROM Person;"

7.14 QueryCollection.h

Go to the documentation of this file.

```
1 #ifndef FTP_QUERYCOLLECTION_H_
2 #define FTP_QUERYCOLLECTION_H_
3
4 #include <string>
5
6 namespace ftp::query {
7
8     const std::string createTable =
9         "CREATE TABLE IF NOT EXISTS Person("
10         "ID INT PRIMARY KEY NOT NULL, "
11         "NAME VARCHAR NOT NULL, "
12         "AGE INT NOT NULL, "
13         "ADDRESS VARCHAR, "
14         "SALARY REAL );";
15
16     const std::string insertData =
17         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (1, 'AAAA', 11, 'Armenia', 10000 ); "
18         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (2, 'BBBB', 22, 'Germany', 20000 ); "
19         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (3, 'CCCC', 33, 'France', 30000 ); "
20         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (4, 'DDDD', 44, 'USA', 40000 ); "
21         "INSERT INTO Person (ID,NAME,AGE,ADDRESS,SALARY) VALUES (5, 'EEEE', 55, 'UK', 500000 ); ";
22
23     const std::string selectData = "SELECT * FROM Person;";
24
25 } // end namespace ftp::query
26 #endif // FTP_QUERYCOLLECTION_H_
```


7.15 SQLBackup.cpp File Reference

```
#include "SQLBackup.h"
#include "Logger.h"
#include "DbManager.h"
#include "FTPManager.h"
#include "QueryCollection.h"
#include <ctime>
#include <fstream>
#include <stdexcept>
#include <filesystem>
```

Namespaces

- namespace `ftp`

Functions

- `std::string ftp::getDateTime ()`

7.16 SQLBackup.h File Reference

```
#include "CmdLineOptionParser.h"
```

Classes

- class `ftp::SQLBackup`

Namespaces

- namespace `ftp`

7.17 SQLBackup.h

Go to the documentation of this file.

```
1 #ifndef SQLBACKUP_H_
2 #define SQLBACKUP_H_
3
4 #include "CmdLineOptionParser.h"
5
6 namespace ftp {
7
8     class SQLBackup
9     {
10     public:
11         SQLBackup();
12
13         void start(int argc, char* argv[]);
14
15     private:
16         void initArgParser();
17
18         CmdLineOptionParser cmdParser_;
19     };
20
21 } // end namespace ftp
22 #endif // SQLBACKUP_H_
```


Index

- ~FTPManager
 - ftp::FTPManager, 15
- addOption
 - ftp::CmdLineOptionParser, 13
- checkOption
 - ftp::CmdLineOptionParser, 13
- CmdLineOptionParser.cpp, 21
- CmdLineOptionParser.h, 21
- createTable
 - ftp::dbmgr, 10
 - ftp::query, 12
- data
 - ftp::dbmgr::TableData, 19
- DbManager.cpp, 22
- DbManager.h, 22
- dumpToFile
 - ftp::dbmgr::TableData, 19
- endSession
 - ftp::FTPManager, 16
- error
 - ftp::Logger, 17
- ftp, 9
 - getDateTime, 9
- ftp::CmdLineOptionParser, 13
 - addOption, 13
 - checkOption, 13
 - getIntOptionValue, 14
 - getStringOptionValue, 14
 - parseArgs, 14
 - printHelp, 15
- ftp::dbmgr, 9
 - createTable, 10
 - init, 10
 - insertData, 10
 - removeDb, 11
 - selectData, 11
- ftp::dbmgr::TableData, 18
 - data, 19
 - dumpToFile, 19
 - isEmpty, 19
 - print, 19
- ftp::FTPManager, 15
 - ~FTPManager, 15
 - endSession, 16
 - FTPManager, 15
 - getErrMsg, 16
 - startSession, 16
 - transfer, 16
- ftp::Logger, 17
 - error, 17
 - info, 17
- ftp::query, 12
 - createTable, 12
 - insertData, 12
 - selectData, 12
- ftp::SQLBackup, 17
 - SQLBackup, 18
 - start, 18
- FTPManager
 - ftp::FTPManager, 15
- FTPManager.cpp, 23
- FTPManager.h, 24
- getDateTime
 - ftp, 9
- getErrMsg
 - ftp::FTPManager, 16
- getIntOptionValue
 - ftp::CmdLineOptionParser, 14
- getStringOptionValue
 - ftp::CmdLineOptionParser, 14
- info
 - ftp::Logger, 17
- init
 - ftp::dbmgr, 10
- insertData
 - ftp::dbmgr, 10
 - ftp::query, 12
- isEmpty
 - ftp::dbmgr::TableData, 19
- Logger.h, 25
- main
 - main.cpp, 26
- main.cpp, 25
 - main, 26
- parseArgs
 - ftp::CmdLineOptionParser, 14
- print
 - ftp::dbmgr::TableData, 19
- printHelp
 - ftp::CmdLineOptionParser, 15
- QueryCollection.h, 26

- removeDb
 - ftp::dbmgr, 11
- selectData
 - ftp::dbmgr, 11
 - ftp::query, 12
- SQLBackup
 - ftp::SQLBackup, 18
- SQLBackup.cpp, 27
- SQLBackup.h, 27
- start
 - ftp::SQLBackup, 18
- startSession
 - ftp::FTPManager, 16
- transfer
 - ftp::FTPManager, 16