mysql_connector_c++_V1.0.1

目录

目录

目录 2
增删改查基础 3
普通查询 3
已准备的查询 4
更新记录 4
事务处理 5

增删改查基础

普通查询

```
#include <string>

#include <mysql_driver.h>
#include <cppconn/driver.h>
#include <cppconn/statement.h>
#include <cppconn/metadata.h>
#include <cppconn/exception.h>


int main(int argc, char** argv)
{
   sql::Driver *driver = nullptr;

   try
   {
         driver = sql::mysql::get_driver_instance();

      std::unique_ptr<sql::Connection> conn;
      conn.reset(driver->connect("tcp://172.17.0.2:3306", "root", "test"));
      conn->setSchema("test");

      std::unique_ptr<sql::Statement> stmt;
      stmt.reset(conn->createStatement());

      std::unique_ptr<sql::ResultSet> rs;
      rs.reset(stmt->executeQuery("select * from user"));
      while(rs->next())
         {
          int id = rs->getInt("userid");
          std::string name = rs->getString("username");
          std::cout << id << " : " << name << std::endl;
         }
   }
   catch(sql::SQLException & e)
   {
      std::cout << "exception:" << e.what() << std::endl;
   }
   catch(...)
   {
```

```
    std::cout << "exception:" << "unknown" << std::endl;
  }

  return 0;
}
```

已准备的查询

```
    sql::Driver *driver = nullptr;
    try
    {
            driver = sql::mysql::get_driver_instance();

        std::unique_ptr<sql::Connection> conn;
        conn.reset(driver->connect("tcp://172.17.0.2:3306", "root", "test"));
        conn->setSchema("test");

        std::unique_ptr<sql::PreparedStatement> stmt;
        const char* sql = "select userid, username, userpass from user"
                " where userid < ? "
                " and username = ? ";
        stmt.reset(conn->prepareStatement(sql));

        stmt->setInt(1, 10);
        stmt->setString(2, "tom");
        std::unique_ptr<sql::ResultSet> rs;
        rs.reset(stmt->executeQuery());
        while(rs->next())
            {
            int id = rs->getInt("userid");
            std::string name = rs->getString("username");
            std::cout << id << " : " << name << std::endl;
            }
    }
    catch(sql::SQLException & e)
    {
        std::cout << "exception:" << e.what() << std::endl;
    }
    catch(...)
    {
       std::cout << "exception:" << "unknown" << std::endl;
    }
```

更新记录

```
    sql::Driver *driver = nullptr;
```

```
try
{
      driver = sql::mysql::get_driver_instance();

    std::unique_ptr<sql::Connection> conn;
    conn.reset(driver->connect("tcp://172.17.0.2:3306", "root", "test"));
    conn->setSchema("test");

    std::unique_ptr<sql::PreparedStatement> stmt;
    const char* sql = "insert into user "
          " (username, userpass) values "
          " (?, md5(?))";
    stmt.reset(conn->prepareStatement(sql));

    stmt->setString(1, "zhangsan");
    stmt->setString(2, "zhangsantest");
    std::unique_ptr<sql::ResultSet> rs;
    int count = stmt->executeUpdate();
    std::cout << "execute update:" << count << std::endl;
}
catch(sql::SQLException & e)
{
    std::cout << "exception:" << e.what() << std::endl;
}
catch(...)
{
   std::cout << "exception:" << "unknown" << std::endl;
}
```

事务处理

```
sql::Driver *driver = nullptr;
 std::unique_ptr<sql::Connection> conn;

 try
 {
      driver = sql::mysql::get_driver_instance();

    conn.reset(driver->connect("tcp://172.17.0.2:3306", "root", "test"));
    conn->setSchema("test");
    conn->setTransactionIsolation(sql::TRANSACTION_REPEATABLE_READ);
    conn->setAutoCommit(false);

    std::unique_ptr<sql::PreparedStatement> stmt;
    const char* sql = "insert into user "
          " (username, userpass) values "
          " (?, md5(?))";
```

```
    stmt.reset(conn->prepareStatement(sql));

    stmt->setString(1, "zhangsan");
    stmt->setString(2, "zhangsantest");
    int count = stmt->executeUpdate();
    std::cout << "execute update:" << count << std::endl;

    sql = "insert into user "
                " (username, userpass) values "
                " (?, md5(?))";
    stmt.reset(conn->prepareStatement(sql));

    stmt->setString(1, "lisi");
    stmt->setString(2, "lisitest");
    count = stmt->executeUpdate();
    std::cout << "execute update:" << count << std::endl;

    conn->commit();
  }
  catch(sql::SQLException & e)
  {
    std::cout << "exception:" << e.what() << std::endl;
    conn->rollback();
  }
  catch(...)
  {
    std::cout << "exception:" << "unknown" << std::endl;
  }
```

备注:

(a)在设置了 autoCommit 为 false 之后，如果不显式提交事务，则在断开连接时会自动回滚。