70 测试案例

在这个案例中，一方面实践了QSqlDatabase类和QSqlQuery类的基本用法，另一方面也验证了在Qt程序中访问MySQL数据库的可行性。

70.1 构建MySQL驱动

Qt和MySQL是来自不技术社区和企业的软件产品。二者在版本上很难保持协调一致。因此从Qt5以后，Qt官方不再提供预构建的MySQL驱动。需要MySQL驱动的用户可以利用Qt官方提供的驱动源码，结合其系统上实际安装的MySQL，自行构建MySQL驱动，以实现驱动与数据库的完美匹配。

70.1.1 修改MySQL驱动项目文件

编辑C:\Qt\Qt5.12.8\5.12.8\Src\qtbase\src\plugins\sqldrivers目录下的qsqldriverbase.pri文件，注释第4行，添加第5行：

```
1  QT  = core core-private sql-private
2
3  # For QMAKE_USE in the parent projects.
4  # include($$shadowed($$PWD)/qtsqldrivers-config.pri)
5  include(./configure.pri)
6
7  PLUGIN_TYPE = sqldrivers
8  load(qt_plugin)
9
10 DEFINES += QT_NO_CAST_TO_ASCII QT_NO_CAST_FROM_ASCII
```

编辑C:\Qt\Qt5.12.8\5.12.8\Src\qtbase\src\plugins\sqldrivers\mysql目录下的mysql.pro文件，注释第6行，添加第13、14、15行：

```
1  TARGET = qsqlmysql
2
3  HEADERS += $$PWD/qsql_mysql_p.h
4  SOURCES += $$PWD/qsql_mysql.cpp $$PWD/main.cpp
5
6  # QMAKE_USE += mysql
7
8  OTHER_FILES += mysql.json
9
10 PLUGIN_CLASS_NAME = QMYSQLDriverPlugin
11 include(../qsqldriverbase.pri)
12
13 INCLUDEPATH += "C:\Program Files\MySQL\MySQL Server 8.0\include"
14 LIBS += "C:\Program Files\MySQL\MySQL Server 8.0\lib\libmysql.lib"
15 DESTDIR = "C:\Qt\Qt5.12.8\5.12.8\mingw73_64\plugins\sqldrivers"
```

70.1.2 构建MySQL驱动动态链接库文件

用QtCreator打开C:\Qt\Qt5.12.8\5.12.8\Src\qtbase\src\plugins\sqldrivers\mysql目录下的mysql.pro文件，构建Release和Debug版本。在C:\Qt\Qt5.12.8\5.12.8\mingw73_64\plugins\sqldrivers目录下可以看到下面这两个动态链接库文件：

- qsqlmysql.dll：Release版本的MySQL驱动

- qsqlmysqld.dll：Debug版本的MySQL驱动

70.1.3 复制MySQL连接器动态链接库文件

将C:\Program Files\MySQL\MySQL Server 8.0\lib目录下的libmysql.dll文件，拷贝到C:\Qt\Qt5.12.8\5.12.8\mingw73_64\bin目录下。

70.2 案例

70.2.1 创建项目

通过QtCreator，在C:\Users\Minwei\Projects\Qt路径下，创建名为DBTest的控制台（Console）项目，并在项目文件中添加第2行：

```
QT -= gui
QT += sql

CONFIG += c++11 console
CONFIG -= app_bundle

# The following define makes your compiler emit warnings if you use
# any Qt feature that has been marked deprecated (the exact warnings
# depend on your compiler). Please consult the documentation of the
# deprecated API in order to know how to port your code away from it.
DEFINES += QT_DEPRECATED_WARNINGS

# You can also make your code fail to compile if it uses deprecated APIs.
# In order to do so, uncomment the following line.
# You can also select to disable deprecated APIs only up to a certain version of Qt.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs deprecated before Qt 6.0.0

SOURCES += \
        main.cpp

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target
```

70.2.2 建库建表

C:\Users\Minwei\Projects\Qt\DBTest\qt_testdb.sql：

```
DROP DATABASE IF EXISTS qt_testdb;
CREATE DATABASE qt_testdb;
USE qt_testdb;

SET NAMES utf8mb4;

CREATE TABLE `t_department`  (
  `id` int UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` varchar(64) DEFAULT NULL,
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

INSERT INTO `t_department` VALUES (1, '研发部');
INSERT INTO `t_department` VALUES (2, '市场部');
INSERT INTO `t_department` VALUES (3, '人力资源部');

CREATE TABLE `t_employee`  (
  `id` int UNSIGNED NOT NULL AUTO_INCREMENT,
  `name` varchar(32) DEFAULT NULL,
  `gender` varchar(16) DEFAULT NULL,
  `department_id` int UNSIGNED DEFAULT NULL,
  `salary` decimal(8,2) DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `employee_of_department` FOREIGN KEY (`department_id`) REFERENCES `t_department` (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

INSERT INTO `t_employee` VALUES (1, '张飞', '男', 1, 20000);
INSERT INTO `t_employee` VALUES (2, '赵云', '男', 1, 30000);
INSERT INTO `t_employee` VALUES (3, '关羽', '男', 2, 15000);
INSERT INTO `t_employee` VALUES (4, '黄忠', '男', 2, 25000);
INSERT INTO `t_employee` VALUES (5, '马超', '男', 3, 20000);
```

在C:\Users\Minwei\Projects\Qt\DBTest目录下启动控制台，并执行如下命令：

```
C:\Users\Minwei\Projects\Qt\DBTest> mysql -uroot -p123456
mysql> source qt_testdb.sql
...
mysql> exit
```

70.2.3 实现功能

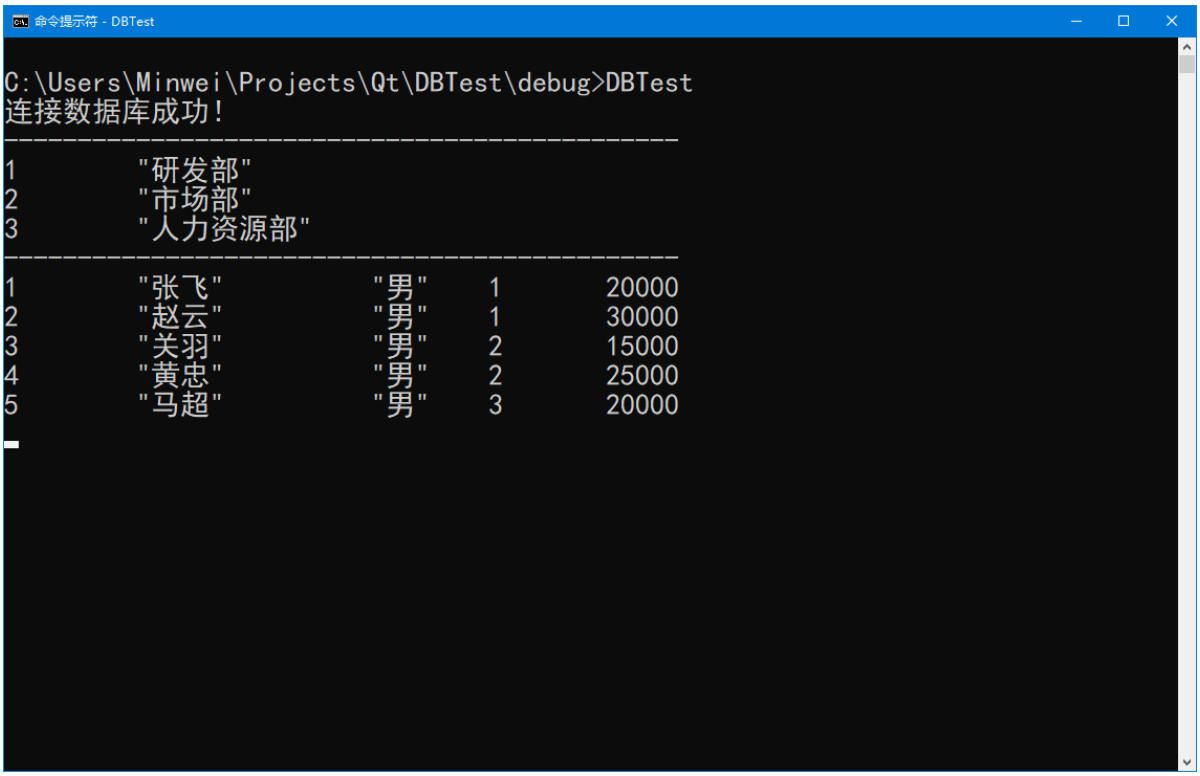
C:\Users\Minwei\Projects\Qt\DBTest\main.cpp：

```
#include <QCoreApplication>
#include <QSqlDatabase>
#include <QSqlQuery>
#include <QSqlError>
#include <QDebug>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    QSqlDatabase db = QSqlDatabase::addDatabase("QMYSQL");
    db.setHostName("localhost");
    db.setPort(3306);
    db.setUserName("root");
    db.setPassword("123456");
    db.setDatabaseName("qt_testdb");
    if (!db.open())
    {
        qDebug() << "连接数据库失败:" << db.lastError().text();
        return -1;
```

```
    }

    qDebug() << "连接数据库成功! ";

    qDebug() << "-----------------------------------------------";

    QSqlQuery query;

    query.exec("select * from t_department");
    while (query.next())
    {
        qDebug()
            << query.value(0).toInt() << '\t'
            << query.value(1).toString();
    }

    qDebug() << "-----------------------------------------------";

    query.exec("select * from t_employee");
    while (query.next())
    {
        qDebug()
            << query.value(0).toInt() << '\t'
            << query.value(1).toString() << '\t'
            << query.value(2).toString() << '\t'
            << query.value(3).toInt() << '\t'
            << query.value(4).toDouble();
    }

    return a.exec();
}
```

70.2.4 测试验证

运行效果如图所示：


命令提示符 - DBTest
C:\Users\Minwei\Projects\Qt\DBTest\debug>DBTest
连接数据库成功!
1 "研发部"
2 "市场部"
3 "人力资源部"
1 "张飞" "男" 1 20000
2 "赵云" "男" 1 30000
3 "关羽" "男" 2 15000
4 "黄忠" "男" 2 25000
5 "马超" "男" 3 20000