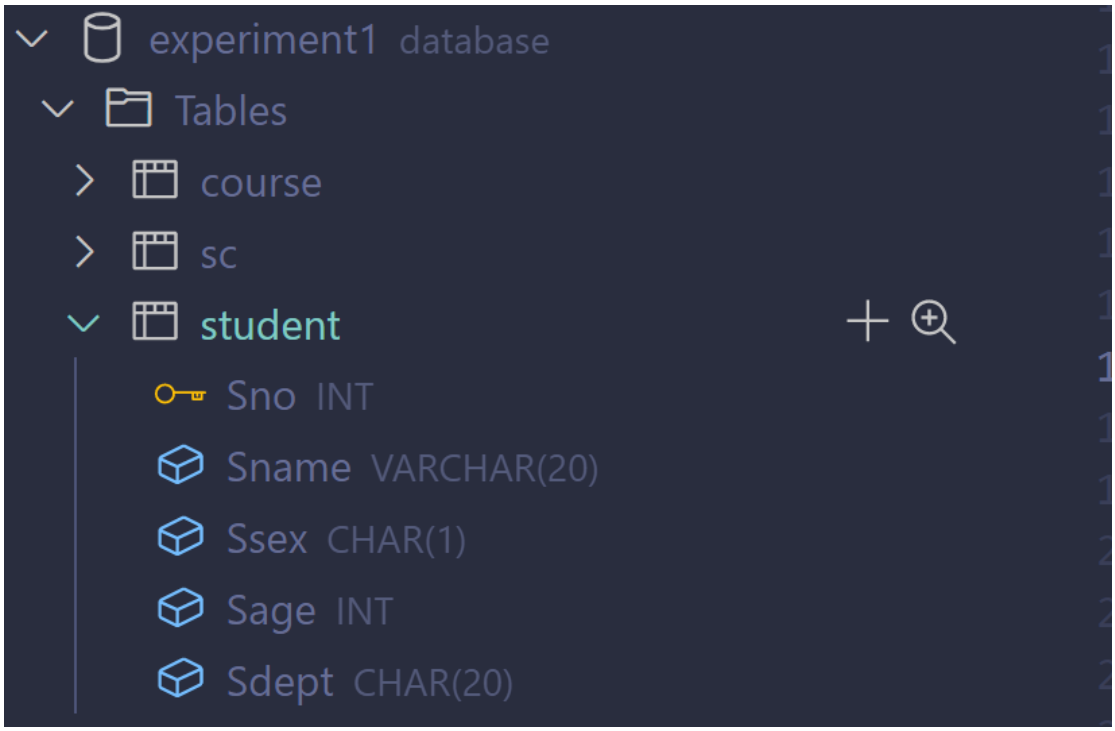
# 实验三

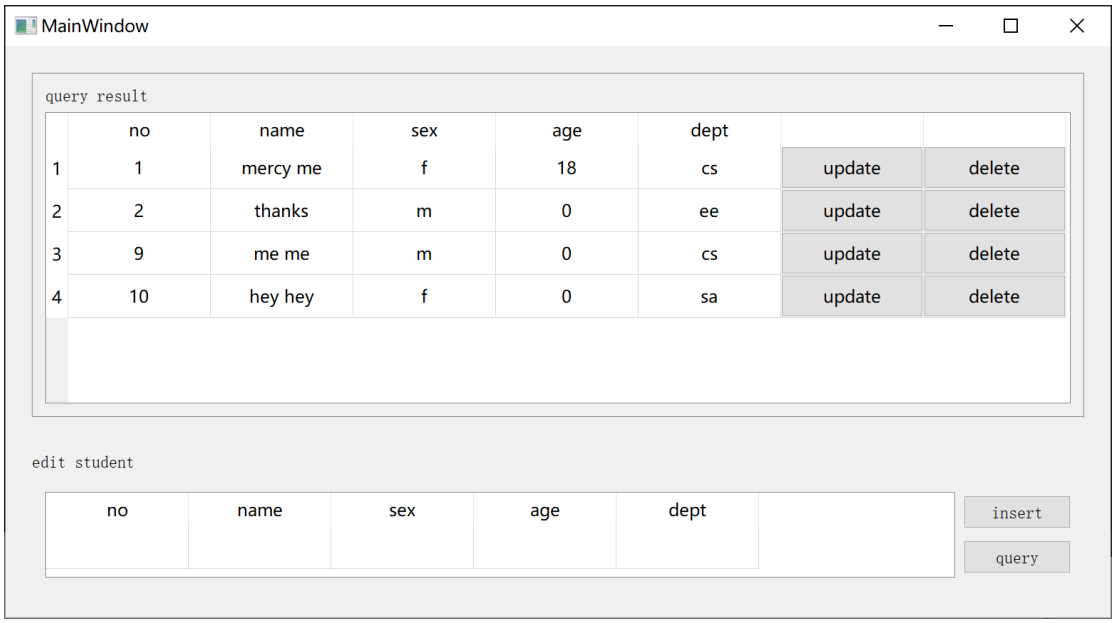
## 1. 数据库

数据库和表如下所示



# 2. 连接数据库及查询

## 2.1. 运行结果如下图所示




## 2.2. 代码如下所示

连接代码

```
StudentDAO::StudentDAO() : host("root"), password("520711YY"),
database("experiment1")
{
    mysql_init(&mysql);
    if (!mysql_real_connect(&mysql, "localhost", host, password,
database, 3306, nullptr, 0))
    {
        std::cout << "connect failed" << mysql_errno(&mysql) <<
std::endl;
    }
    else
        std::cout << "connect succeed" << std::endl;
}
StudentDAO::~StudentDAO()
{
    mysql_close(&mysql);
    printf("disconnect\n");
```

```
}
```

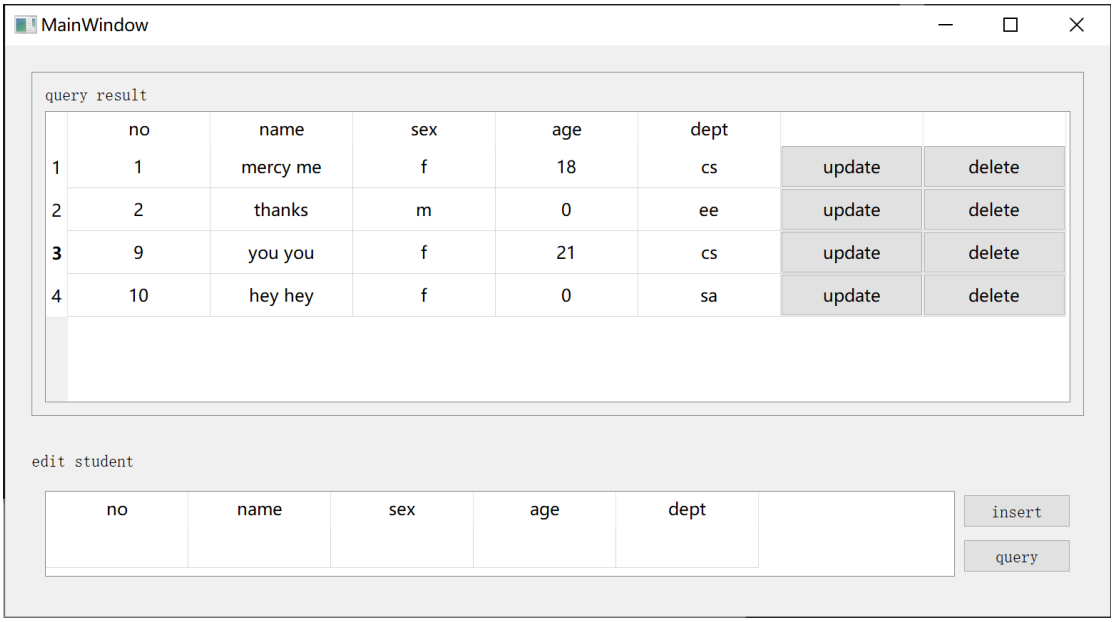
查询代码:

```
bool StudentDAO::loadStudents(std::vector<Student> &students)
{
    return queryStudents(students, "select * from student");
}

bool StudentDAO::queryStudents(std::vector<Student> &students, const
char *query)
{
    if (exec(query))
    {
        res = mysql_store_result(&mysql);
        if (res == nullptr)
        {
            std::cout << "store failed: " << mysql_errno(&mysql) <<
std::endl;
            return false;
        }
        else
            while (row = mysql_fetch_row(res))
                students.emplace_back(Student(row));
    }
    mysql_free_result(res);
    return true;
}

bool StudentDAO::queryStudents(std::vector<Student> &students, const
char *query)
{
    if (exec(query))
    {
        res = mysql_store_result(&mysql);
        if (res == nullptr)
        {
            std::cout << "store failed: " << mysql_errno(&mysql) <<
std::endl;
            return false;
        }
        else
            while (row = mysql_fetch_row(res))
                students.emplace_back(Student(row));
```

```
    }
    mysql_free_result(res);
    return true;
}
```

# 3. 更新数据

## 3.1. 运行结果如下图所示




## 3.2. 代码如下所示

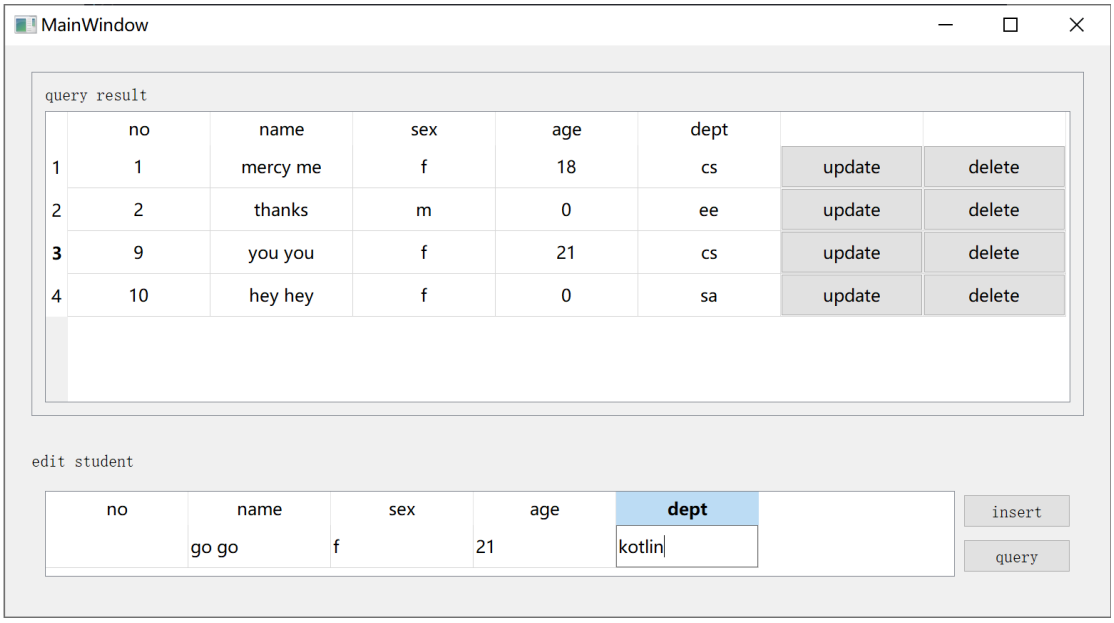
更新代码

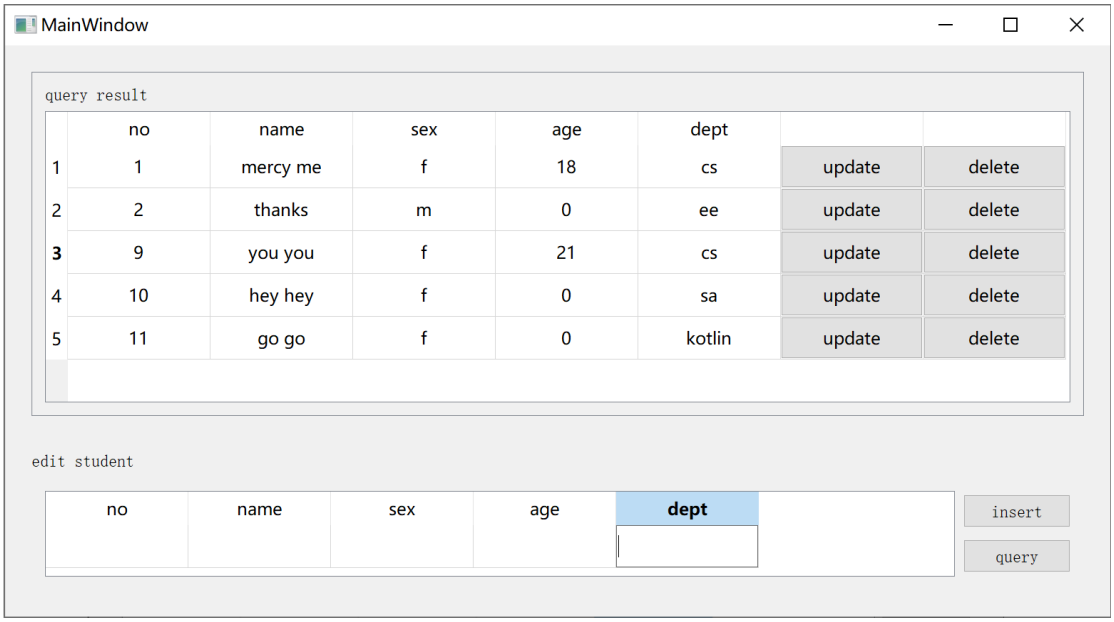
```
bool StudentDAO::updateStudent(const Student &val)
{
    std::stringstream builder;
    builder << "update student set Sname = '" << val.name << "', Ssex =
'" << val.sex << "', Sage = " << val.age
            << ", Sdept = '" << val.dept << "' where Sno = " << val.no;
    const string &str = builder.str();
    const char *query = str.c_str();
    return exec(query);
}
bool StudentDAO::queryStudents(std::vector<Student> &students, const
char *query)
```

```
{
    if (exec(query))
    {
        res = mysql_store_result(&mysql);
        if (res == nullptr)
        {
            std::cout << "store failed: " << mysql_errno(&mysql) <<
std::endl;
            return false;
        }
        else
            while (row = mysql_fetch_row(res))
                students.emplace_back(Student(row));
    }
    mysql_free_result(res);
    return true;
}
```

# 4. 插入数据

## 4.1. 运行结果如下图所示





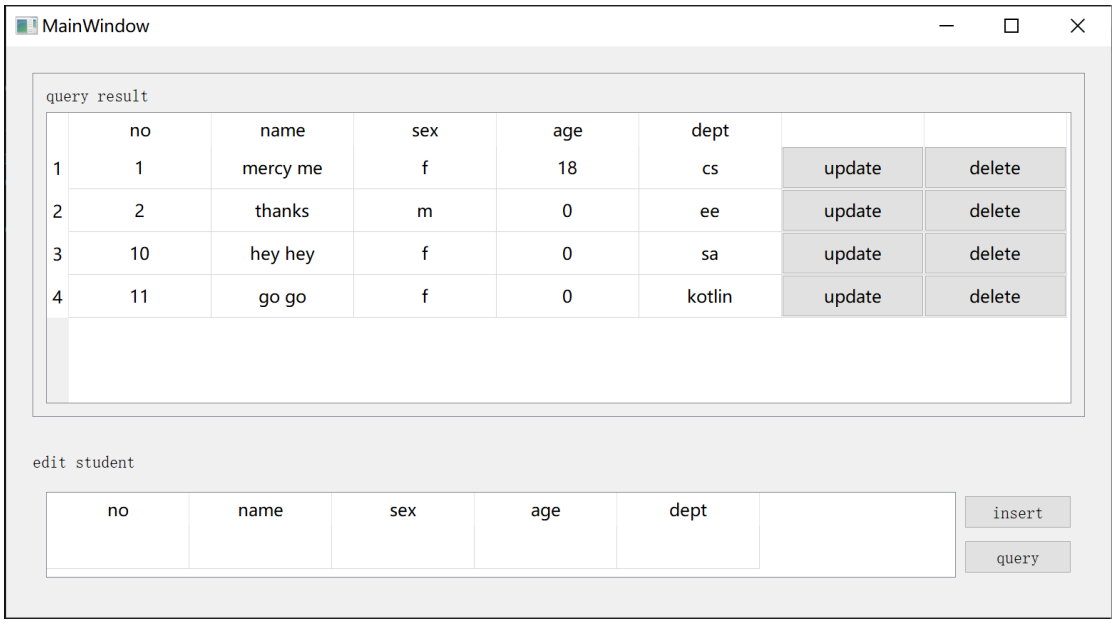

## 4.2. 代码如下所示

插入代码

```
bool StudentDAO::updateStudent(const Student &val)
{
    std::stringstream builder;
    builder << "update student set Sname = '" << val.name << "', Ssex =
'" << val.sex << "', Sage = " << val.age
            << ", Sdept = '" << val.dept << "' where Sno = " << val.no;
    const string &str = builder.str();
    const char *query = str.c_str();
    return exec(query);
}
```

# 5. 删除数据

## 5.1. 运行结果如下图所示




## 5.2. 代码如下所示

删除代码

```
bool StudentDAO::deleteStudent(const Student &student)
{
    char query[] = "delete from student where Sno = ";
    strcat(query, std::to_string(student.no).c_str());
    return exec(query);
}
```