《数据库系统原理》实验报告（）					
题目：实验五					
学号	2351753	姓名	黄保翔	日期	5/7

实验环境：docker miniob

实验步骤及结果截图：

1.在 Docker 中建立 miniob 环境（参考第 0 章的教程 PPT 和第三节的内容）。

```
# git clone https://github.com/oceanbase/miniob.git
fatal: destination path 'miniob' already exists and is not an empty directory.
# bash build.sh --make -j4
bash: build.sh: No such file or directory
# cd miniob
/bin/sh: 3: cd: not found
# cd miniob
# bash build.sh --make -j4

# cd build
# ./bin/observer -s miniob.sock -f ../etc/observer.ini &
#
Welcome to the OceanBase database implementation course.

Copyright (c) 2021 OceanBase and/or its affiliates.

Learn more about OceanBase at https://github.com/oceanbase/oceanbase
Learn more about MiniOB at https://github.com/oceanbase/miniob

Successfully load ../etc/observer.ini

# ./bin/obclient -s miniob.sock

Welcome to the OceanBase database implementation course.

Copyright (c) 2021 OceanBase and/or its affiliates.

Learn more about OceanBase at https://github.com/oceanbase/oceanbase
Learn more about MiniOB at https://github.com/oceanbase/miniob

miniob >
```

2.创建一张表，包括学号，姓名，绩点，学分。

```
miniob > create table Student(
    No int ,
    Name char(10),
    Grade float,
    Credit float
);
SUCCESS
```

3. 向该表插入几行数据，其中需要包含一条包含个人学号、姓名的数据

```
miniob > insert into Student
VALUES (2350001, '张三', 4.8, 120.0);
SUCCESS
```

```
miniob > insert into Student
VALUES (2350002, '李四', 4.7, 124.5);
SUCCESS

miniob > insert INTO Student
values (2350003, '王五', 4.2, 99.5);
SUCCESS

miniob > insert INTO Student
values (2351753,'黄保翔',4.9,100.5);
SUCCESS
```

4.使用 select 语句展示学号，绩点，学分。

```
miniob > SELECT No, Grade, Credit
FROM Student;
No | Grade | Credit
2350002 | 4.7 | 124.5
2350003 | 4.2 | 99.5
2350001 | 4.8 | 120
2351753 | 4.9 | 100.5
```

5.尝试修改指定行的绩点或学分如下表所示，能否成功？为什么？

不能成功

```
miniob > UPDATE Student
SET Grade = 4.82
WHERE No = 2350001;
FAILURE
```

```
miniob > UPDATE Student
SET Grade = 4.65
WHERE No = 2350003;
FAILURE
miniob > UPDATE Student
SET Credit = 103.5
WHERE No = 2350003;
FAILURE
```

文件地址：https://github.com/oceanbase/miniob/blob/main/src/observer/sql/stmt/update_stmt.cpp

Code Blame 26 lines (22 loc) · 882 Bytes Raw

```
/* Copyright (c) 2021 OceanBase and/or its affiliates. All rights reserved.
miniob is licensed under Mulan PSL v2.
You can use this software according to the terms and conditions of the Mulan PSL v2.
You may obtain a copy of Mulan PSL v2 at:
         http://license.coscl.org.cn/MulanPSL2
THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OF ANY KIND,
EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO NON-INFRINGEMENT,
MERCHANTABILITY OR FIT FOR A PARTICULAR PURPOSE.
See the Mulan PSL v2 for more details. */

//
// Created by Wangyunlai on 2022/5/22.
//

#include "sql/stmt/update_stmt.h"

UpdateStmt::UpdateStmt(Table *table, Value *values, int value_amount)
    : table_(table), values_(values), value_amount_(value_amount)
{}

RC UpdateStmt::create(Db *db, const UpdateSqlNode &update, Stmt *&stmt)
{
  // TODO
  stmt = nullptr;
  return RC::INTERNAL;
}
```

Update 函数为空

6.删除张三的记录

```
miniob > DELETE FROM Student
WHERE Name = '张三';
SUCCESS
```

7.使用 select 语句展示你的学号。

```
miniob > SELECT No
FROM Student
WHERE Name = '黄保翔'
No
2351753
```

8. 对 miniob 源码进行阅读，主要选取一个功能（如 create table、insert、 delete 等）进行分析理解，做简要报告（不超过两页）

选取 create 功能进行分析

源码地址：https://github.com/oceanbase/miniob/blob/main/src/observer/sql/stmt/create_table_stmt.cpp

类定义地址：

https://github.com/oceanbase/miniob/blob/main/src/observer/sql/stmt/create_table_stmt.h#L28

源码分析：

```
  RC CreateTableStmt::create(Db *db, const CreateTableSqlNode
&create_table, Stmt *&stmt)
{
  StorageFormat storage_format =
get_storage_format(create_table.storage_format.c_str());
  if (storage_format == StorageFormat::UNKNOWN_FORMAT) {
    return RC::INVALID_ARGUMENT;
  }
  stmt = new CreateTableStmt(create_table.relation_name,
create_table.attr_infos, create_table.primary_keys,
storage_format);
  sql_debug("create table statement: table name %s",
create_table.relation_name.c_str());
  return RC::SUCCESS;
}
```

该部分定义 create 函数，属于 CreateTableStmt 类，用于创建表示 CREATE TABLE 语句的语句对象。

```
StorageFormat storage_format =
get_storage_format(create_table.storage_format.c_str());
```

调用 get_storage_format 函数，将 create_table.storage_format（字符串形式的存储格式）转换为 StorageFormat 枚举值，并存储在 storage_format 变量中。

```
if (storage_format == StorageFormat::UNKNOWN_FORMAT) {
    return RC::INVALID_ARGUMENT;
  }
```

如果用户提供的存储格式字符串不被支持，返回表示存储格式无效

```
stmt = new CreateTableStmt(create_table.relation_name,
create_table.attr_infos, create_table.primary_keys,
storage_format)
```

动态分配一个 CreateTableStmt 对象，并将其地址赋值给输出参数 stmt

sql_debug 函数负责进行日志打印

最后返回枚举类型表示创建对象成功

```
StorageFormat CreateTableStmt::get_storage_format(const char
*format_str) {
    StorageFormat format = StorageFormat::UNKNOWN_FORMAT;
    if (strlen(format_str) == 0) {
      format = StorageFormat::ROW_FORMAT;
    } else if (0 == strcasecmp(format_str, "ROW")) {
      format = StorageFormat::ROW_FORMAT;
    } else if (0 == strcasecmp(format_str, "PAX")) {
      format = StorageFormat::PAX_FORMAT;
    } else {
      format = StorageFormat::UNKNOWN_FORMAT;
    }
    return format;
  }
```

该函数定义了 上面用到的 get_storage_format 函数 ，将输入的存储格式字符串转换为 StorageFormat 枚举值。

类定义部分：

CreateTableSqlNode 提供了解析后的结构化数据，CreateTableStmt::create 函数利用这些数据构造语句对象。构造函数处初始化对象，接收表名、列定义、主键列表和存储格式初始化私有成员变量。

const string &table_name() const { return table_name_; }

const vector<AttrInfoSqlNode> &attr_infos() const { return attr_infos_; }

const vector<string> &primary_keys() const { return primary_keys_; }

const StorageFormat storage_format() const { return storage_format_; }

此处定义外部接口允许外部查询对象属性

综上，创建表的过程如下

解析 SQL 语句：

- 解析器生成 CreateTableSqlNode，包含表名、列定义、主键和存储格式。

验证存储格式：

- get_storage_format 将存储格式字符串转换为枚举值，确保有效性（ROW_FORMAT 或 PAX_FORMAT）。

构造语句对象：

- create 函数调用构造函数，创建 CreateTableStmt 对象，封装表定义信息。

存储信息：

- 构造函数初始化私有成员变量（table_name_、attr_infos_、primary_keys_、storage_format_）。

提供访问接口：

- 访问接口（如 table_name()）允许后续模块获取表定义。

出现的问题：

1.一直插入错误

```
miniob > insert into Student
values (2350001, '张三', 4.8, 120);
FAILURE
```

```
miniob > insert INTO Student
VALUES (2350001, '张三', 4.8, 120),
       (2350002, '李四', 4.7, 124.5),
       (2350003, '王五', 4.2, 99.5),
       (2351753,'黄保翔',4.9,100);
FAILURE
```

解决方案：

1.不能在一个语句中插入多个元组，并且由于设置的是 float 类型所以必须是小数类型

```
miniob > insert into Student
VALUES (2350001, '张三', 4.8, 120.0);
SUCCESS
```