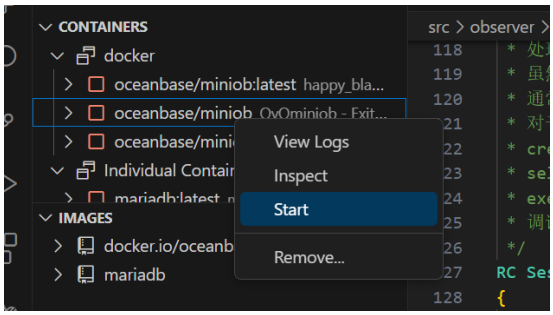
《数据库系统原理》实验报告 5					
题目：MINIOB 实验二					
学号	2152809	姓名	曾崇然	日期	2023-11-25

实验环境：在 docker 中布置 miniob 环境，使用 git 克隆 miniob 的代码到本地，将获取到的文件与 docker 容器映射连接，在安装了 docker 插件的 VSCODE 中进行代码的修改，编译和调试

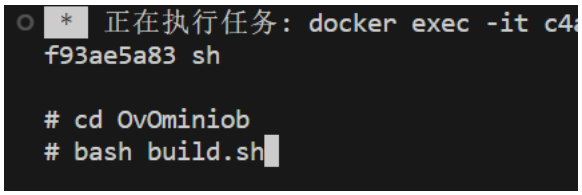
实验步骤及结果截图：

1. 开发调试环境准备

（1）启动镜像环境



（2）进行编译检查是否环境搭建成功



2. 修改代码

（1）选择题目:LIKE

（2）过程：

①在词法规则文件中添加能识别 LIKE 和 NOT LIKE 的词法规则

```
LIKE                                    RETURN_TOKEN(LIKE_OvO);
NOT[ \t]+LIKE                           RETURN_TOKEN(NOTLIKE_OvO);
```

②在语法规则文件中添加定义和对应的规则

a. TOKEN 定义

```
LIKE_OvO
NOTLIKE_OvO
```

b. 添加识别规则

```
comp_op:
      EQ { $$ = EQUAL_TO; }
    | LT { $$ = LESS_THAN; }
    | GT { $$ = GREAT_THAN; }
    | LE { $$ = LESS_EQUAL; }
    | GE { $$ = GREAT_EQUAL; }
    | NE { $$ = NOT_EQUAL; }
    | LIKE_OvO { $$ = LIKE; }
    | NOTLIKE_OvO { $$ = NOTLIKE; } /*jing_like代码,第4处修改，将like标识符转化为类中的对应属性*/
    ;
```

c. 在定义文件中添加对应的运算符

```
enum CompOp
{
  EQUAL_TO,     ///< "="
  LESS_EQUAL,   ///< "<="
  NOT_EQUAL,    ///< "<>"
  LESS_THAN,    ///< "<"
  GREAT_EQUAL,  ///< ">="
  GREAT_THAN,   ///< ">"
  LIKE,
  NOTLIKE,          ///< "LIKE" jing_like代码，
  NO_OP
};
```

d. 在具体执行比较的代码中添加 LIKE 和 NOTLIKE 的具体实现

```
case LIKE: {
  std::string input = left.get_string();
  std::string pattern = right.get_string();
  input = std::regex_replace(input, std::regex("([.*])"), "\\$1");
  pattern = std::regex_replace(pattern, std::regex("([.*])"), "\\$1");// 转义 . 和 *
  pattern = std::regex_replace(pattern, std::regex("%"), ".*");
  pattern = std::regex_replace(pattern, std::regex("_"), ".");
  pattern = std::regex_replace(pattern, std::regex("['']"), ""); // 排除单引号
  std::regex regex(pattern);
  result = std::regex_match(input, regex);
} break;
case NOTLIKE: {
  std::string input = left.get_string();
  std::string pattern = right.get_string();
  input = std::regex_replace(input, std::regex("([.*])"), "\\$1");
  pattern = std::regex_replace(pattern, std::regex("([.*])"), "\\$1");// 转义 . 和 *
  pattern = std::regex_replace(pattern, std::regex("%"), ".*");
  pattern = std::regex_replace(pattern, std::regex("_"), ".");
  pattern = std::regex_replace(pattern, std::regex("['']"), ""); // 排除单引号
  std::regex regex(pattern);
  result = std::regex_match(input, regex);
  result = !result;
} break;
```

3. 编译、运行和验证

（1）编译运行

```
[100%] Built target client_performance_test
[100%] Built target record_manager_test
# ./bin/observer -f ../etc/observer.ini -P cli
sh: 3: ./bin/observer: not found
# cd build_debug
# ./bin/observer -f ../etc/observer.ini -P cli
Successfully load ../etc/observer.ini
miniob > 
```

（2）验证

```
miniob > select * from OvOteam;
name
lei
jing
long

miniob > select * from OvOteam where name like '%n%';
name
jing
long
```

4. 提交测试

Branch	Commit id	任务状态	成绩	结果
jing_Like	c223895620e966...	● 执行成功	50.000	成功 5 失败 20

出现的问题：

第一次提交测试没有通过测试，经过查看测试结果发现 LIKE 还同时要求了 NOT LIKE 的实现，需要将 NOT LIK 同样识别为一个关键字

解决方案：

不能直接在 NOT 和 LIKE 之间添加空格使其识别，应当在其中间添加[\t]来表明有一个空格，之后在对应的语法，和具体实现中添加上对应的代码，即完成了 NOT LIKE 的实现