

《Rust程序设计语言》课程项目

基于Rust的数据库实现

授课教师: 冯洋

项目助教:潘中颢

1071992145@qq.com

南京大学计算机学院





- 本项目要求利用Rust实现一个简单的数据库。要求同学们实现的基础 功能包括:
 - ▶仅需实现基本的数据类型,包括int、varchar;
 - ➤至少支持单行与多行注释、select、insert、update、delete增删改查和create、drop数据表等基本操作,所有表操作在仅一个默认Database中执行;
 - ▶持久化存储引擎,能够将数据存储在磁盘上;
 - ▶执行引擎,能够读入SQL语句并执行,返回表结果或报错信息。
 - ▶其他拓展功能(至少一项)

项目目标



• 项目背景:

- ▶数据库(DataBase,简称DB)是长期储存在计算机内、有组织的、可共享的大量数据的集合。数据库的基本特点包括数据按一定的数据模型组织、描述和存储,具有较高的数据独立性,冗余度较小,且易于扩展。数据库的主要功能包括数据定义、数据组织、存储和管理,数据操纵(如插入、删除、修改和查询),事务管理和运行管理,以及数据库的建立和维护等。
- ▶数据库可分为基于关系模型的关系数据库(MySQL、SQLServer等)和不依赖 关系模型的非关系数据库(MongoDB、Redis等)。
- ▶本选题要求基于Rust实现一个简单的关系数据库操作平台。



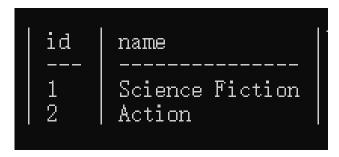
•基础需求细节:输入输出

▶接口

请在lib.rs中书写函数 "pub fn execute_sql(sql_statement: &str) -> bool", pub use它, 该函数将作为执行sql语句的统一接口,测试时将从这个方法传sql语句;

▶输出

 为了避免格式的不统一,请在输出结果到终端时,都用以下的统一样式。需要注意的是, 所有字段在表单元格中,列中最长字段距离左右边界各1个空格,其他字段与最长字段向 左对齐。每个单元格左右边界相距至少五个空格,若最长字段小于3,则也向左对齐,空 余用空格补足:



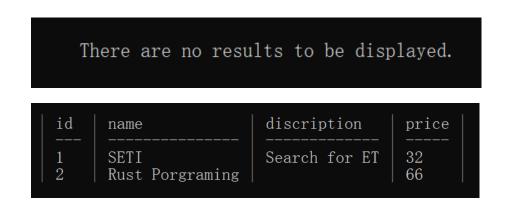
• 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行sql语句,点击copy Markdown查看数据表的正确格式。

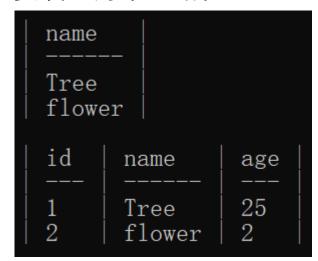


•基础需求细节:输入输出

▶输出

- 如果执行输入语句时有SELECT语句输出为空,则不输出任何消息;但如果函数 execute_sql执行所有sql语句后,发现结果仍为空,则打印 "There are no results to be displayed." 到终端。
- 如果字段为空(null), 全部以空格补齐。
- 连续输出两次表格时,表格与表格之间要间隔一行。
- 持久化数据请以相对路径设置在项目目录下,不要设置为本地路径。







- •基础需求细节:输入输出
 - ▶输出
 - 出现主键PRIMARY KEY重复INSERT的错误:输出 "Error: Duplicate entry '\$value' for key 'PRIMARY'"。

Error: Duplicate entry '1' for key 'PRIMARY'

• 出现主键PRIMARY KEY或不允许空值NOT NULL列存放NULL的错误:输出 "Field '\$col_name' doesn't have a default value"。

Error: Field 'price' doesn't have a default value

• 出现SQL语句的语法错误: 直接报错 "Error: Syntax error"

Error: Syntax error

- 若在一段SQL语句出现多个错误,自行安排输出顺序即可。
- 错误处理建议使用Result返回来恢复现场。另外感兴趣的同学可尝试使用RUST_BACKTRACE 功能锁定源码错误位置,但请记得提交代码运行用例时不要开启该功能。



- 基础需求细节:测试用例包含的sql语法
 - **≻**CREATE
 - 数据类型为INT和VARCHAR, 需要都能标注位数;
 - 主键PRIMARY KEY
 - 非空NOT NULL
 - **≻DROP**
 - 删除一或多个数据表;

• 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行sql语句,点击copy Markdown 查看数据表的正确格式。若与PPT规则不一致,以PPT为准。



- 基础需求细节:测试用例包含的sql语法
 - **≻**DELETE
 - 删除满足某些条件的单个表的数据;
 - >UPDATE
 - 更新满足某些条件的单个表的数据;
 - >INSERT
 - 向单个表按列字段或完整插入一或多条数据;
 - >SELECT
 - 按条件查询单表全部或部分列数据,查询列数据后直接计算(如 "SELECT col*2 FROM table_name"),能按正序或倒序排序;
 - 计算表达式(即简单算式,如SELECT 1 + 1,注意结果数据表的列的写法);
- 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行sql语句,点击copy Markdown 查看数据表的正确格式。若与PPT规则不一致,以PPT为准。



- 基础需求细节:测试用例包含的sql语法
 - ▶约束
 - where指定一个或多个简单布尔条件(>, <, =, 为空, 不为空);
 - ▶其他
 - 支持单行和多行注释;
 - 支持sql语句拆分多行输入(以 "\n"分隔)
 - 支持输出主键列数据相同值重复插入错误、按列插入数据不含NOT NULL字段数据的错误、 语法错误;支持没有输出结果时输出要求的提示信息
 - 实现至少一种前述要求以外的特色功能
- 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行sql语句,点击 copy Markdown查看数据表的正确格式。若与PPT规则不一致,以PPT为准





- •测试用例示例
- 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行sql语句,点 击copy Markdown查看数据表的 正确格式。若与PPT规则不一致, 以PPT为准。

```
测试结果:
  name
   Action |
```

```
▶测试用例2号
   CREATE TABLE plants_test2 (
     id INT(32) PRIMARY KEY,
     name VARCHAR(100) NOT NULL
   -- 插入数据
   INSERT INTO plants_test2 VALUES (1, "Science
   Fiction");
   DROP TABLE plants_test2;
   CREATE TABLE plants_test2 (
     id INT(32) PRIMARY KEY,
     name VARCHAR(100) NOT NULL
   INSERT INTO plants_test2 VALUES (1, "Action");
   -- 查询表中的所有数据
   SELECT * FROM plants_test2;
```





- •测试用例示例
- 可在 https://www.db-fiddle.com/ 在 DB FIDDEL运行sql语句,点击copy Markdown查看数据表的正确格式。若与PPT规则不一致,以PPT为准。

```
测试结果:
|age|
|---|
|25|
|1|
```

```
▶测试用例3号
   CREATE TABLE plants (
     id INT(32) PRIMARY KEY,
     name VARCHAR(100) NOT NULL,
     age INTEGER
   );
   -- 插入数据
   INSERT INTO plants VALUES (1, "Tree", 25);
   INSERT INTO plants VALUES (2, "flower", 1);
   查询表中年龄
   SELECT age FROM plants;
```





- •测试用例示例
- 可在 https://www.db-fiddle.com/ 在DB FIDDEL运行 sql语句,点击copy Markdown 查看数据表的正确格式。若与 PPT规则不一致,以PPT为准。

测试结果:

Error: Duplicate entry '1' for key 'PRIMARY'

```
▶测试用例17号
    CREATE TABLE books_test17 (
      id INT(32) PRIMARY KEY,
      name VARCHAR(100),
      left_num INT(32),
      discription VARCHAR(150),
      price INT NOT NULL
    INSERT INTO books_test17 (id, name,
    discription, price) VALUES (1, "SETI", "Search for
    ET", 32);
    INSERT INTO books_test17 (left_num, id, name,
    price) VALUES (23, 1, "Rust Programing", 66);
    SELECT * FROM books test17;
```

评分标准



- 提交形式:项目源代码 + 项目文档 + 汇报PPT (以后续通知为准)
- 评分标准(项目分数满分100分,详细赋分标准请看说明文档)
 - ▶测试用例(50分)
 - 获得分数 = $\frac{通过测试用例数目}{总测试用例数目}*50$
 - ▶现场报告(30分)
 - 系统基础讲解(10分):展示四个基础需求的核心代码与运行示例截图,说明系统架构及其实现方式,包括sql解析器parser、持久化存储模块等,体现系统使用了哪些rust特性(所有权、生命周期、零成本抽象、智能指针、动态派发等),满足了哪些质量属性(易理解性、鲁棒性、可扩展性、可重用性、性能等),是否能通过cargo test测试系统;
 - 特色功能讲解(15分):基础需求之外的功能实现,包括但不限于实现了新数据类型、约束,或者我们更鼓励的基于B+树的存储引擎和查询优化、事务管理等。特色功能需要现场演示;
 - 现场问答(5分)
 - ▶代码质量与文档(20分)





- 评分标准(项目分数满分100分,详细赋分标准请看说明文档):
 - ▶其他标准
 - 若系统的基础功能没有完成,特色功能的分数会扣 5-15分;
 - 系统可以借鉴已有的Rust开源项目,但必须确保与原项目有极大的区分,相应的要求会在现有评分标准的特色功能上提高,即基础功能的分数将全部划归到特色功能中,基础功能不再需要汇报,改为仅特色功能占25分,赋分梯度改为18分-25分,9分-17分,0分-8分三档。不可借助闭源项目。借鉴的项目必须给出对应开源项目网址。项目源码会进行抄袭检测,若存在抄袭或严重欺瞒,按项目总分0分处理。

参考资料



- CMU 15445 https://15445.courses.cs.cmu.edu/fall2023/
- ➤ Building a Simple DB in Rust https://johns.codes/blog/build-a-db/part01
- ➤B+ Tree Visualization
 https://www.cs.usfca.edu/~galles/visualization/BPlusTree.html



《Rust程序设计语言》课程项目

基于Rust的数据库实现

授课教师: 冯洋

项目助教:潘中颢

1071992145@qq.com

南京大学计算机学院