**数据库系统project报告**

2021-2022学年第2学期（CST31106）

|  |  |
| --- | --- |
| 数据库系统project任务书 | |
| 名称 | 数据库存储管理设计与模拟实现 |
| 类型 | □验证性 □设计性 综合性 |
| 内容 | 关系型数据库采用自描述的方式进行数据库系统管理，关于关系的相关数据也存储在数据库管理系统里。  针对关系型数据库存储模块，进行需求分析，并设计一个简单的数据库系统存储模块，根据设计模拟实现数据存储管理器，主要实现：数据库创建、表格创建、数据添加、删除、更新等操作过程中，数据库系统所进行的存储方面的管理；设计索引，并比较有索引和无索引的区别。  模拟实现采用：python或者Java实现具体功能，数据存储采用excel或者文本文件，可用文本文件模拟硬盘数据块，不需要实现SQL语句的执行编译，SQL语句采用函数实现。 |
| 要求 | （1）设计方案要合理；  （2）能基于该存储管理模块实现数据的存储和读取；  （3）设计方案有一定的效率分析。 |
| 任务时间 | 2022年4月28日至2022年5月20日 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 小组成员 | | | | | | |
| 20195633 | | 20192680 | | 20195217 | |  |
| 李燕琴 | | 杨旭峰 | | 杨思怡 | |  |
| 项目评分表 | | | | | | |
| 序号 | 评分项 | | 分值 | | 得分 | |
| 1 | 需求分析 | | 3分 | |  | |
| 2 | 综合设计与实现 | | 4分 | |  | |
| 3 | 团队协作 | | 3分 | |  | |
| 项目总得分： | | | | | | |

课程项目评分标准（总分10分）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 评分项目 | 完成情况 | 得分 |
| 1 | 需求分析 | 分析合理 | 3分 |
| 分析较合理 | 2分 |
| 分析不合理 | 1分 |
| 分析完全错误 | 0分 |
| 2 | 综合设计与实现 | 设计完整，设计合理，工具使用熟练 | 4分 |
| 设计较完整，设计合理，工具使用较熟练 | 3分 |
| 设计较完整，设计较合理，工具使用较熟练 | 2分 |
| 设计较完整，设计不合理，工具使用不熟练 | 1分 |
| 抄袭、被抄袭 | 0分 |
| 3 | 团队协作 | 有团队，分工合理，密切协作 | 3分 |
| 有团队，分工合理，有一定协作 | 2分 |
| 有团队，分工不合理，无协作 | 1分 |
| 无团队，无协作 | 0分 |

# 小组分工

李燕琴：框架和思路的确定，位示图管理数据块的架构和实现

杨思怡：思路讨论，代码框架逻辑架构和实现

杨旭峰：思路讨论，文档撰写

# 正文部分

## 需求分析

### 任务描述

关系型数据库采用自描述的方式进行数据库系统管理，关于关系的相关数据也存储在数据库管理系统里。

针对关系型数据库存储模块，进行需求分析，并设计一个简单的数据库系统存储模块，根据设计模拟实现数据存储管理器，主要实现：数据库创建、表格创建、数据添加、删除、更新等操作过程中，数据库系统所进行的存储方面的管理；设计索引，并比较有索引和无索引的区别。

模拟实现采用：python或者Java实现具体功能，数据存储采用excel或者文本文件，可用文本文件模拟硬盘数据块，不需要实现SQL语句的执行编译，SQL语句采用函数实现。

### 功能需求

需求1：关于关系的相关数据需要存储在数据库管理系统中，即表的名称，以及字段名称均需单独形成文件并保存。

需求2：数据存储管理器需在数据库的相关操作中实现对数据块的管理。

1. 数据库创建：需为数据库分配空间。
2. 表格创建：需要为表格分配空间，并以一定的形式记录数据块的名称和容量信息。
3. 数据添加：需要查询当前表格的可用的数据块的名称和位置，并将数据存入数据块中。
4. 数据删除：需要首先定位到被删除数据的存储位置，在该存储块中删除数据。并同时更新数据块实际存储数据条数等信息。
5. 数据更新：需要首先定位到待更新的数据的存储位置，将新的值写入数据块中，完成数据的更新。
6. 数据查询：虽然任务中未提到数据查询的需求，但是考虑功能的完整性，我们将数据查询作为待实现需求之一。

需求3：设计索引，实现有索引数据库存储管理。另需实现无索引数据库存储管理，并比较有索引和无索引数据库的效率。

## 设计与实现

### 总体设计思路

采用python作为实现语言，借助pandas和os库实现数据管理器，对数据库创建，表格创建等操作过程中的数据存取和删除修改进行管理。

### 与关系相关的数据的存储

经分析，我们将与关系相关的数据，即表的字段名称及其类型用field.pkl记录。 将每个表的空间使用情况用block.pkl记录。

图片包含 徽标

描述已自动生成

### 有索引数据操作

（1）实现思路

有索引数据库操作实现主要包括图中的七个部分，同时每个部分需对操作的合理性进行检查，并对相关数据文件进行创建或修改操作。具体实现在后续说明。

日程表

描述已自动生成

图 1 实现思路概要图

（2）存储管理

**索引设计：**关于索引，我们设计并实现了散列索引index，在python中以dict的形式实现。以数据的主键作为索引的键，以数据在位示图中的位置作为索引的值。索引文件的存取通过\_\_load\_index()和\_\_save\_index()函数实现。即用joblib从indek.pkl的存储路径加载数据，或将修改后的dict形式的index数据存入指定存储路径。具体格式如图 1所示。

文本

中度可信度描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

图 2 index格式

**字段存储：**字段的管理中，需要用field记录字段的名称和类型。这里同样采用dict实现。以字段名称为的field的键，以字段的类型为field的值。field的存取也是通过load和save函数实现，思路与索引的存取一致。具体格式如图 2所示。

文本

中度可信度描述已自动生成

文本

中度可信度描述已自动生成

文本

描述已自动生成

图 3 field格式

**数据块管理：**我们采用位示图来进行数据块的管理，位示图中每一位的值为0说明对应的块为空可赋值，否则为不可用。位示图的实现在python中用整型列表blocks模拟，列表中的每个整数代表位示图中的一行，32位整型数字的每个二进制位对应位示图的一列，二进制位的值即为位示图单元的值。简单示意如下。

图片包含 形状

描述已自动生成

Blocks的存取实现如下。

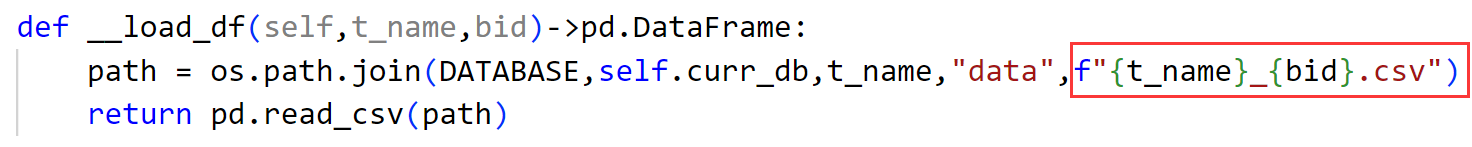
文本

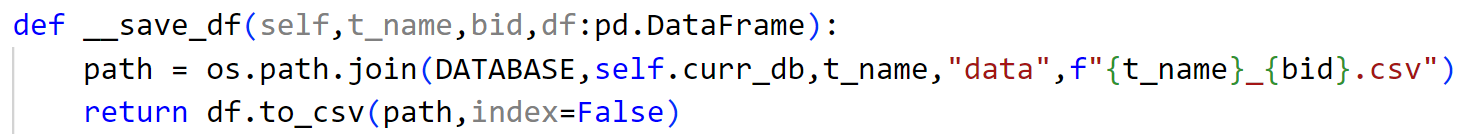
低可信度描述已自动生成

文本

中度可信度描述已自动生成

**元数据存储：**我们将数据的值存在.csv文件中，并用dataframe对csv文件进行读写。每个csv文件对应一个数据块。Csv文件的名字由表名和数据块名字决定。存取实现如下。具体格式如图 3所示。





文本

描述已自动生成

图 4 元数据存储格式

整个文件层次结构示意图如下图所示。

表格

中度可信度描述已自动生成

其中代码文件结构如下图所示。

文本

描述已自动生成

（3）数据插入操作

借鉴sql插入语句中的参数，该数据库中插入数据需要给出待插入数据的表的名字t\_name，数据元组row。列的名字cols为可选项。

执行具体操作前，先获取索引index和字段field数据。

图形用户界面, 文本

描述已自动生成

参照field中的字段生成元数据，再将待插入数据的值赋值给对应字段，同时记录主键的值pk\_value。

图形用户界面, 文本, 应用程序

描述已自动生成

此时metadata中存储的已为待插入数据，还需要找到可用的数据块，将数据写入。

文本

描述已自动生成

查找可用数据块的过程为，遍历位示图，找到第一个为空的数据单元，并返回可用数据单元所在的block的 id和列下标lid。否则返回-1，-1说明无可用块。

图形用户界面, 文本

描述已自动生成



若bid为-1，说明已经无可用数据块，则新增一个数据块，即在block中增加一个整数。

否则将找到的可用的数据单元置位，说明当前数据单元已有值。并用\_\_load\_df函数读取对应数据块。

文本

描述已自动生成

找到可用数据块后，将数据写入块中，在索引中记录该条数据的位置，并将更新后的数据块，索引和位示图保存。

文本

描述已自动生成

（4）数据删除操作

删除操作用表名t\_name和主键名pk\_value定位待删除的数据。读取表的索引和位示图信息。

文本

描述已自动生成

删除数据前需要先判断索引文件中是否有该数据的记录，否则说明表中没有该条数据，无法进行删除操作。如果有该条数据，则根据索引文件中记录的数据的位置定位到存该条数据的数据块。如果数据块为空还是说明这个数据块里实际上未存数据，操作无效。否则进行删除操作，删除时只先删除索引中数据的记录信息，让数据块中存储的数据不可获取，同时将位示图对应位置置0。

文本

描述已自动生成

（5）数据更新操作

更新数据时，同样用表名t\_name和主键值pk\_value定位数据。读取数据块对应的.csv文件，将新的值update\_value写入待更新的字段update\_col中。

文本

描述已自动生成

（6）数据查询操作

查询数据时，除表名外，还需要给出想查询的属性列表sel\_cols。主键的值为可选项。

文本

描述已自动生成

由于主键是唯一的，因此若给定主键，则只能查询到一条数据。具体实现中，用索引中主键对应的位置值定位数据。

文本

描述已自动生成

若没给定主键，则可以遍历位示图，查询到所有有值的数据块的字段的值。

文本

描述已自动生成

### 无索引数据操作

无数据库索引操作的存储管理和增删改查操作的实现基本一致。主要区别在无索引数据操作不能利用索引文件定位数据块，创建表时也不需要生成索引文件。

1. 数据插入操作

判断数据有无重复时，需遍历位示图，判断是否已存在相同主键的数据。

文本

描述已自动生成

1. 数据删除操作

数据的删除也需要遍历位示图来找到需要删除的数据的位置。

文本

描述已自动生成

1. 数据更新和查询操作

这两个操作也需要对位示图进行遍历，具体实现与数据插入和删除操作中遍历思路一致，不再赘述。

### 数据库功能测试

（1）数据准备

生成随机数据，作为待填充到数据库中数据。

文本

描述已自动生成

（2）进行测试

创建测试：记录开始时间，进行数据表创建测试。

图片包含 图表

描述已自动生成

插入测试：将随机生成数据插入表中进行测试。

文本, 信件

描述已自动生成

更新测试：对指定数据进行更新。

图片包含 文本

描述已自动生成

删除覆盖测试：删除数据后再插入相同主键的数据，测试数据能否被删除以及再插入数据后能否成功覆盖之前的数据。并进行批量测试。

图形用户界面, 文本, 应用程序

描述已自动生成

查询测试：查询单条数据，以及所有的数据输出。

图片包含 图示

描述已自动生成

记录结束时间，并计算运行时长。

图片包含 文本

描述已自动生成

（3）获得测试结果

将结果重定向到log文件中。

文本

描述已自动生成

### 测试结果

（1）插入数据

log文件中部分记录如下，可以看到，数据库根据数据量扩充了数据块数。

文本, 信件

描述已自动生成文本

描述已自动生成

（2）删除覆盖

如下图示，删除数据后再删除时，提示执行错误。并且在删除后插入与被删除的值的主键相同的数据时，可以成功执行，说明数据库支持删除覆盖，且结果正确。

文本

描述已自动生成

删除覆盖测试中，new\_num为20，因此最新的数据应为DBUSER10。查询结果中有该条数据，说明了删除覆盖的有效性。

文本, 表格

描述已自动生成

（3）更新与查询

在测试中，更新了第一条和第二条数据。日志文件的查询结果中，第一条和第二条的数据内容与更新内容一致。这说明了更新是有效的。

图片包含 图形用户界面

描述已自动生成

测试中对主键为499的数据进行了查询。查询结果与之前插入的数据一致。说明查询有效。

图片包含 文本

描述已自动生成

1. 有索引和无索引比较

运行时间比较:

无索引数据库：74.030s；有索引数据库：23.591s。

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图 5 无索引数据库运行时间

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

图 6 有索引数据库运行时间

显然，有索引数据库的用时远远短于无索引数据库。

具体耗时比较：如下为无索引数据库和有索引数据库增删改查步骤的时间分布图。

图表, 折线图, 直方图

描述已自动生成

图 7 无索引数据库运行时间趋势分析图

图表, 折线图

描述已自动生成

图 8 有索引数据库运行时间趋势分析图

从图上可以明显看到，无索引数据库最后的数据查询阶段耗时远高于有索引数据库。有索引数据库可以根据索引信息快速定位数据块，时间复杂度为O(1)，而无索引数据库每次都需要遍历位示图来查找数据块，时间复杂度为O(n)，且随着数据的增多，无索引数据库插入的时间消耗也呈线性增长，而有索引数据库表现较为稳定。因此，可以看到无索引数据库在插入、查询、删除上耗时上比有索引数据库耗时高得多，且随着数据量越大，有索引数据库的优势越明显。

# 总结

本项目基于散列索引管理所有元数据、位示图管理数据块使用情况，实现了有索引、数据块分块存放的简易数据库，完成了数据库创建、表格创建、数据增删改查等功能，并与无索引数据库在时间上进行了性能对比，并对不同数据操作的耗时趋势上进行对比分析，得出随着数据量越大，有索引数据库的优势越明显的结论。