用户手册

1. 用户数据目录结构

用户数据储存目录结构如下：

- `log/DATABASE` 储存日志文件
- `data/manager.db` 储存数据库名称
- `data/DB_NAME/` 是名为 `DB_NAME` 的数据库的文件夹
  - `data/DB_NAME/TABLE_NAME/` 是名为 `TABLE_NAME` 的表的文件夹
  - `data/DB_NAME/TABLE_NAME/_meta` 储存表结构
  - `data/DB_NAME/TABLE_NAME/TABLE_NAME.bin` 储存整棵B+树的二进制表示

2. 元数据管理模块

- `create database dbName;`

```
create database mydatabase;
create database ab;
```

- `show databases;`

```
show databases;
```

- `drop database dbName;`

```
drop database ab;
```

- `use database dbName;`

```
use mydatabase;
```

- `create table`

```
create table student(name String(256), age Int not null, id Int, PRIMARY
KEY(name));
```

- `show table tablename;`

```
show table student;
```

- `drop table tablename;`

```
drop table student;
```

3. 存储模块

- `insert`

```
insert into student values ('qwq', 47, 1),('abc', 8, 2),('lll', 19);
select * from student;
```

- `delete`

```
delete from student where age <= 18;
select * from student;
```

- `update`

```
update student set name = 'wqw' where id < 2;
select * from student;
update student set age = 20;
select * from student;
```

4. 查询模块

- 单表查询

```
select * from student;
select name, age from student where id > 1;
select age from student where name = 'lll';
```

- 多表查询（两张表的join）

```
create table choose(name String(256), course String(256), ID Int, PRIMARY
KEY(ID));
insert into choose values ('lll', 'ML', 101);
insert into choose values ('abc', 'DL', 102);
select * from student join choose on student.name = choose.name;
select choose.course from student join choose on student.name = choose.name;
```

5. 并发模块

Client A	Client B
connect 123 123	
create database mydatabase;	
use mydatabase;	
create table student(name String(256), age Int not null, id Int, PRIMARY KEY(name));	
insert into student values ('qwq', 47, 1);	
select * from student;	
// 避免脏读	
begin transaction	
update student set name = 'wqw' where id < 2;	
	connect 124 124
	select * from student; 报错
commit	
	select * from student; 正确

6. 重启恢复模块

我们的重启恢复模块可以借助`checkpoint`，该选项在`cn.edu.thssdb.utils.Global`内可以调整`public static final Boolean RECOVER_FROM_DISC`以改变. 设置为`false`时不借助`TABLE.bin`，此时可以没有B树的二进制文件目录，恢复时直接从 LOG 文件中恢复。设置为`true`时会借助`TABLE.bin`和`log/DATABASE`日志文件。默认值为`false`。

7. 进阶功能

进阶功能共7项

- 扩展SQL语句：展示所有数据库名称、展示某一表的结构
- 外键约束：在insert/update语句执行时查询父表B+树索引，检查引用完整性
- 页式存储：将B+树每个内部节点和叶子结点均存储在文件的一张页上，读写树上节点均需访问对应页的位置；首个页面存储Header和空闲页面，运行时使用空闲页面链表分配页面
- 提高I/O效率：数据库全局设立缓存池，采用写回缓存的工作方式，并采用LRU的页面置换算法，命中的页面可以直接在内存中访问，修改时也直接在内存中修改，待页面将置换下去之后再写回磁盘对应位置
- 三张表以上的join：采用递归的方式，两张表先求join生成cross_table，再继续和其他表递归求join
- 查询优化：对join查询进行索引嵌套循环连接，大幅度降低特定情况的join查询复杂度
- checkpoint功能：留有单一事务的checkpoint接口，调用时将缓存池内的数据全部写回磁盘并更改log日志，并能够成功恢复

使用方法如下：

- 外键约束

```
create table student (id Int, name String(16), PRIMARY KEY(id));
create table project (id Int, student_id Int, PRIMARY KEY(id), FOREIGN KEY
(student_id) REFERENCES student(id));

insert into student values (9973, 'alice');
insert into project values (0, 9973);
insert into project values (1, 9974); // 受到外键约束而插入失败
```

- 多表查询（三张表的join）

```
CREATE TABLE Customers (CustomerID INT, CustomerName STRING(255), PRIMARY
KEY(CustomerID));
CREATE TABLE Orders (OrderID INT, CustomerID INT, ProductID INT, PRIMARY
KEY(OrderID));
CREATE TABLE Products (ProductID INT, ProductName STRING(255), PRIMARY
KEY(ProductID));

INSERT INTO Customers VALUES (1, 'Alice');
INSERT INTO Orders (OrderID, CustomerID, ProductID) VALUES (3, 1, 3);
INSERT INTO Orders (OrderID, CustomerID, ProductID) VALUES (2, 2, 2);
INSERT INTO Products (ProductID, ProductName) VALUES (3, 'Cherries');
INSERT INTO Products (ProductID, ProductName) VALUES (4, 'Grapes');

SELECT * FROM Customers JOIN Orders JOIN Products ON Products.ProductID =
Orders.ProductID;
```

- 扩展SQL指令

```
show databases; // 展示所有数据库名称

show table Products; // 展示Products表的结构
```

- checkpoint：详见第5部分重启与恢复功能介绍，用户还可以单独输入`checkpoint`指令手动进行checkpoint
- 页式存储 & I/O效率优化 & 查询优化：效率优化的消融实验详见《系统设计文档》