work 8

> **事务管理**
>
> - 杨佳宇轩
> - 22375080

书本练习

10-事务管理

- **1. 试述事务的概念及事务的4个特性。恢复技术能保证事务的哪些特性**

  事务的概念：事务是用户定义的一个数据库操作序列，这些操作要么全做、要么全不做，是一个不可分割的工作单位

  保证的特性：

  - 原子性：事务中的操作要么全做，要么全不做
  - 一致性：事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态
  - 隔离性：事务执行不能被其他事务干扰
  - 持续性：一个事务一定提交，对数据库中数据的改变就是永久性的，接下来的其他操作或故障不应该对其执行结果有任何影响
- **2. 为什么事务非正常结束时会影响数据库数据的正确性？请举例说明之**

  如果事务被迫中断，这些未完成的事务对数据库所做的修改有一部分会写入数据库，这时的数据库无法保证正确状态，即可能出现不一致性

11-并发控制

- **1. 在数据库中为什么要并发控制？并发控制技术能保证事务的哪些特性**

  数据库是共享资源，通常有多个事务同时在运行，当多个事务并发存取数据库时就会产生同时读取或修改同一个数据的情况。若并发操作不加控制就可能会存取和存储不正确的数据，破坏数据库的一致性

  并发控制可以保证事务的**一致性**和**隔离性**
- **2. 并发操作可能会产生哪几类数据不一致？用什么方法能避免各种不一致的情况**

  数据不一致类型：

  - 修饰修改
  - 不可重复读

- 读“脏”数据

如何避免：使用并发控制。常用的并发控制技术包括封锁技术、时间戳方法、乐观控制方法和多版本并发控制方法等

- **3. 什么是封锁？基本的封锁类型有几种？试述它们的含义**

封锁的定义：封锁就是事务 T 在对某个数据对象列入操作之前，先向系统发出请求，对其加锁。加锁后事务 T 就对该数据对象有了一定的控制，在事务 T 释放它的锁之前，其他的事务不能更新或读取此数据

类型：

- 排它锁（X 锁）

  又称为写锁。若事务 T 对数据对象 A 加上了 X 锁，则只允许 T 读取和修改 A，其他的任何事务都不能再对 A 加任何类型的锁，直到 T 释放 A 上的锁。

- 共享锁（S 锁）

  又称为读锁。若事务 T 对数据对象 A 加上了 S 锁，则事务 T 可以读 A 但不能修改 A，其他事务只能再对 A 加 S 锁，而不能加 X 锁，直到 T 释放 A 上的 S 锁。

- **4. 如何用封锁机制保证数据的一致性**

在对数据进行读写操作之前首先对数据执行封锁操作，DBMS按照一定的封锁协议对并发操作进行控制，使得多个并发操作有序地执行，就可以避免丢失修改、不可重复读和读“脏”数据等数据不一致性

- **6. 什么是死锁？请给出预防死锁的若干方法**

一组事务对于锁发生了循环等待，且永远不能结束，形成死锁

预防方法：

- 一次封锁法：要求每个事务必须一次将所有要使用的数据全部加锁，否则就不能继续执行
- 顺序封锁法：预先对数据对象规定一个封锁顺序，所有事务都按照这个顺序实行封锁

- **7. 请给出检测死锁发生的一种方法，当发生死锁后如何解除死锁?**

数据库系统一般采用的方法时允许死锁的放生，DBMS检测到死锁之后加以解除

解除方法：一般使用超时法或事务等待图法，之后采用选择一个处理死锁代价最小的事务，将其撤销，释放此事务持有的所有锁，使其他事物得以继续运行下去

- **8. 什么样的并发调度是正确的调度**

可串行化的调度是正确的调度（多个事务的并发执行是正确的，当且仅当其结果与按某一次序串行执行它们时结果相同）

- **9.**

- (1) 可能的结果是：2、4、8、16

顺序	结果
123	16
132	8
213	4
231	2
312	4
321	2

- (2)

```
Slock A
Y = A = 0
Unlock A
Xlock A
A = Y + 2       Slock A
写回 A          等待
Unlock A        等待
                Y = A = 2
                Unlock A
                Xlock A         Slock A
                A = Y * 2       等待
                写回 A          等待
                Unlock A        等待
                                Y = A = 4
                                Unlock A
                                Xlock A
                                A = Y * Y
                                写回 A
                                Unlock A
```

- (3)

```
Slock A
Y = A = 0
Unlock A        Slock A
Xlock A         Y = A = 0
A = Y + 2       Unlock A
写回 A
Unlock A
```

```
                Xlock A         Slock A
                A = Y * 2       等待
                写回 A          等待
                Unlock A        等z待
                                Y = A = 4
                                Unlock A
                                Xlock A
                                A = Y * Y
                                写回 A
                                Unlock A
```

- (4)

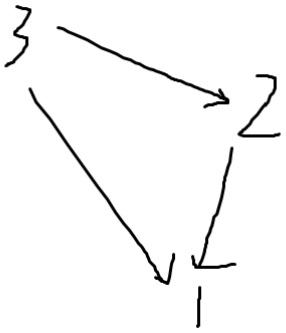
```
Slock A
Y = A = 0
Xlock A
A = Y + 2       Slock A
写回 A          等待
Unlock A        等待
                Y = A = 2
                Xlock A
Unlock A        等待            Slock A
                A = Y * 2       等待
                写回 A          等待
                Unlock A        等待
                                Y = A = 4
                Unlock A
                                Xlock A
                                A = Y * Y
                                写回 A
                                Unlock A
                                Unlock A
```

- (5)

```
Slock A
Y = A = 0
            Slock A
            Y = A = 0
Xlock A
等待
            Xlock A
            等待
                        Slock A
                        Y = A = 0
                        Xlock A
                        等待
```

- **10.**

  由前驱图可以得到该调度是冲突可串行化的



- **12. 举例说明对并发事务的一个调度是可串行化的，而这些并发事务不一定遵守两段锁协议**

```
Slock A
Y = A = 0
Unlock A
Xlock A
A = Y + 2     Slock A
写回 A         等待
Unlock A      等待
              Y = A = 2
              Unlock A
              Xlock A      Slock A
              A = Y * 2    等待
              写回 A        等待
              Unlock A     等待
                           Y = A = 4
                           Unlock A
                           Xlock A
```

```
A = Y * Y
写回 A
Unlock A
```

课堂练习

有如下调度：

r1(X) w1(X) r2(X) r1(Y) w1(Y) r2(Y) w2(Y)

请问是正确的调度吗？若是，等价于什么样的串行调度？

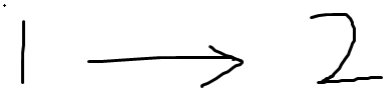
是否是一个2PL的调度？

这是个2PL的调度，加上锁操作后如下 (L指lock，U指unlock) ：

L1(X) L1(Y) r1(X) w1(X) U1(X) L2(X) r2(X) r1(Y) w1(Y) U1(Y) L2(Y) r2(Y) w2(Y)U2(Y)

请问是严格两阶段协议的调度吗？

- 对于第一个问题，冲突图如下



所以是一个可串行化的调度

等价的串行调度就是

```
r1(X) w1(X) r1(Y) w1(Y) r2(x) r2(Y) w2(Y)
```

- 1号事务在对 X 写之后释放了 X 的锁，而之后还进行了 Y 的读写，故不是一个严格两阶段协议的调度