Question 1: During checkpoint, wal-sys divides actions into three types: "PENDING", "COMMITTED" and "DONE", what is the meaning of these types?

PENDING：指在checkpoint时正在运行，还没commit的action。

COMMITTED：指在checkpoint时已经commit但还没end的action。

DONE：指在checkpoint时已经end的action。

Question 2: What is the relationship between the action categories during checkpoint ("PENDING", "COMMITTED" and "DONE") and action categories during recovery ("Winners", "Losers", and "Done")?

对于checkpoint时DONE类型的action，在recovery时仍为DONE类型，即其在checkpoint之前已经把变化写入DB，crash和recovery过程对其无影响。

对于checkpoint时COMMITTED类型的action，其在recovery时应属于Winners或DONE，取决于action在checkpoint和crash期间是否end。如果已经end，则其变化已写入DB，为recovery的DONE类型，crash和recovery过程对其无影响；否则为Winners类型，虽然在crash时数据尚未写入DB，但在recovery时会redo在checkpoint和crash之间的行为，action将以正常运行结束的状态写入DB。

对于checkpoint时的PENDING类型，在recovery时可能属于Winners或Losers，取决于action在checkpoint和crash期间是否commit。如果已经commit，应属于Winners类型，虽然在crash时数据尚未写入DB，但在recovery时会redo在checkpoint和crash之间的行为，action将以正常运行结束的状态写入DB；否则则为Losers类型，recovery时将undo这个action在checkpoint之前的行为，最后在DB中将不会体现出这个action造成的改变。

Question 3: How many lines were rolled back? What is the advantage of using checkpoints?

Note down the action_ids of "Winners", "Losers", and "Done". Use the show_state command to look at the recovered database and verify that the database recovered correctly. Crash the system, and then run wal-sys again to recover the database a second time.

第一次crash恢复后有8行被回滚，第二次crash后有4行被回滚。

使用checkpoint可以保证数据库的一致性，保证内存和硬盘上的数据是一致的；可以缩短recovery的时间，通过checkpoint可以有效减少需要通过日志进行恢复的脏块的数量，从而加快数据恢复。

Question 4: Does the second run of the recovery procedure restore "DB" to the same state as the first run? What is this property called?

是，两次恢复程序运行后的状态均为 studentB 2000 studentA 1100

这个性质叫幂等性。

Question 5: Compare the of "Winners", "Losers", and "Done" from the second recovery with those from the first. The lists are different. How does the recovery procedure guarantee the property from Question 4 even though the recovery procedure can change? (Hint: Examine the "LOG" file). `action_ids`

两次恢复时，LOG文件内容一致。但观察程序输出，两次恢复时Winners，Losers和DONE的分类略有不同。action 2在第一次恢复时为Winners类型，而在第二次恢复时为DONE类型。

第一次恢复时：

```
Winners: id: 2  Losers: id: 3  Done: id: 1
Starting forward scan ...
  REDOING: type: UPDATE action_id: 2 variable: studentB redo: "2000" undo: NULL
  REDOING: type: UPDATE action_id: 2 variable: studentA redo: "1100" undo: "1000"
  Logging END records for winners
Forward scan done
```

第二次恢复时：

```
Winners:   Losers: id: 3  Done: id: 1 id: 2
Starting forward scan ...
  Logging END records for winners
Forward scan done
```

可以看出在检索需要进行undo的log时，恢复程序只会检索并undo Winners类型的action内容，所以第二次在检索需要undo的内容时，会忽略action 2。因此第一次恢复和第二次恢复的结果一致，可以保持程序的幂等性。

Question 6 (Optional): Wal-sys has a hitherto unmentioned option: if you type wal-sys -undo it will perform undo logging and undo recovery. Try the above sequences again with undo logging to see what changes.

如果将reset变成undo，第一次执行时会将studentC写入DB

```
------------------------
On-disk DB contents:
Account: studentC Value: 2900
Account: studentB Value: 2000
Account: studentA Value: 1100
------------------------
```

第一次crash并recovery后会将studentC 相关操作undo，认为action 3为Losers

```
------------------------
On-disk DB contents:
Account: studentB Value: 2000
Account: studentA Value: 1100
------------------------
```

第二次crash并recovery后将action 3分类为Winners，DB状态与第一次recovery一致。