Hands-on 2: Write Ahead Log System (2)

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

"PENDING" refers to those actions that have not been committed during checkpoint. If the system meets a crash and tries to recover from it, then all the pending actions will be undone.

"COMMITTED" refers to those actions that have been committed during checkpoint. If the system meets a crash and tries to recover from it, then all the committed actions will be redone.

"DONE" refers to those actions that have been done, which means whether the system meets a crash has nothing to do with those actions.

> **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")?

The three action categories actually stands for three different kinds of step: Not committed, Committed and done(ended). Their behaviors during recovery are correspondingly Undo, Redo and Do nothing. Whether a action is pending, committed or done during checkpoint depends not only on it self, but also on the timing of checkpoint. A pending action can also become committed if the checkpoint changes its occurring time.

"PENDING" is **"Loser"**; **"COMMITTED"** is **"Winner"** and **"DONE"** is **"Done"**

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

8 lines have been rolled back.

Using checkpoints can persistently save the changes of a system.

Checkpoints can:

- Avoid log size from growing too large:

  The changes before the checkpoints will have been saved persistently and they can be simply discarded. So the log file will be compacted by using checkpoints. It is also good for a programmer to analyze the log files.

- Avoid recovery from a blank state

  Since the log files can be compacted by using checkpoints, the recovery will take less time, for some of the changes in an action have been saved persistently. Therefore, using checkpoints can also reduce the time of recovering.

> **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?

The second run of the recovery procedure restores "DB" to the same state as the first run. This property is called **"All-or-nothing"**. And this kind of log is **"Redo-only"**.

> **Question 5**: Compare the `action_ids` 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:

The first recovery:

- Winners: 2
- Losers: 3
- Done: 1

The second recovery:

- Winners: none
- Losers: 3
- Done: 1 & 2

The recovery procedure has redone the winner action 2. Therefore, action 2 has been done and a log whose type is 'END' has been appended to the LOG file.

During the first recovery, the Winners have been redone to become Done.

During the second recovery, since the Winners have been redone by the first one, this recovery will do nothing new, which guarantee **"All-or-nothing"** property.

> **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.

All the changes have been flushed after appending a line of log. Different from the default logging method, this kind of log is called **"Undo-only"**. Since all the changes have been flushed, the Winners is unnecessary to be redone. The recovery will only need to focus on undoing the Losers.

After the recovery, the losers will have been aborted and a log whose type is 'Aborted' will have been appended to the LOG file.