Checkpoints + Recovery

Final Project Presentation

Kaige Liu, Tianlei Pan, Xuanxuan Ge

Overviews

Implement full recovery from WAL + Checkpoints.

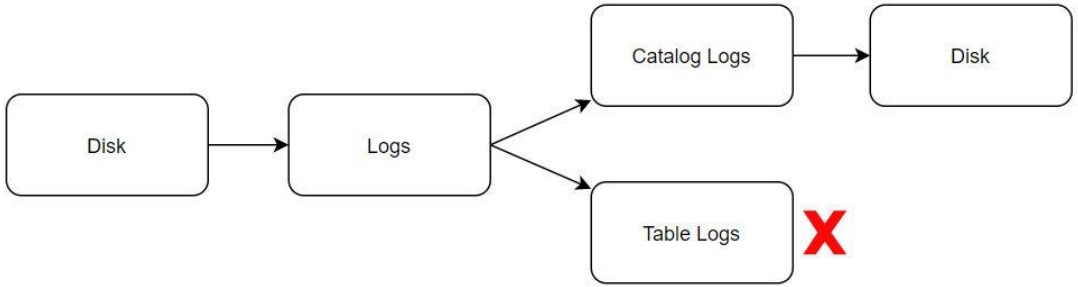
→ 1. Separate Catalog Logs

→ 2. Store DataTables

→ 3. Recover using WAL + Checkpoint.

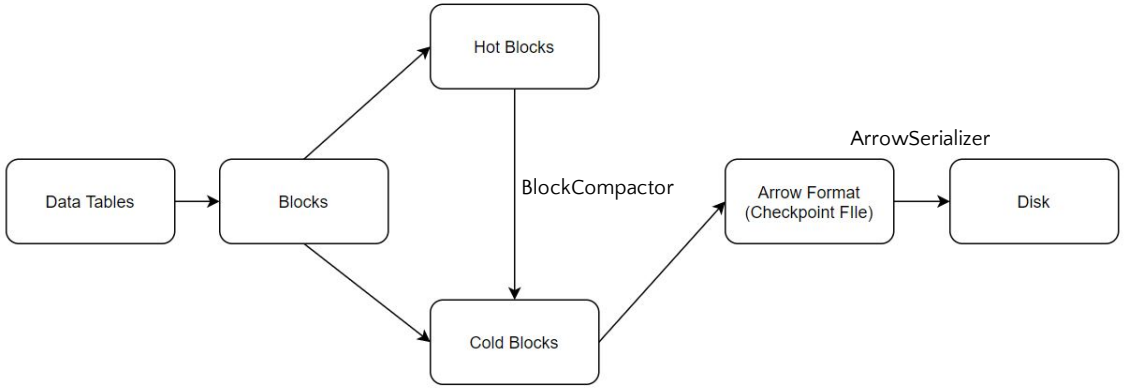
Pipeline

1. Catalog Log Filtering (Per Checkpoint)



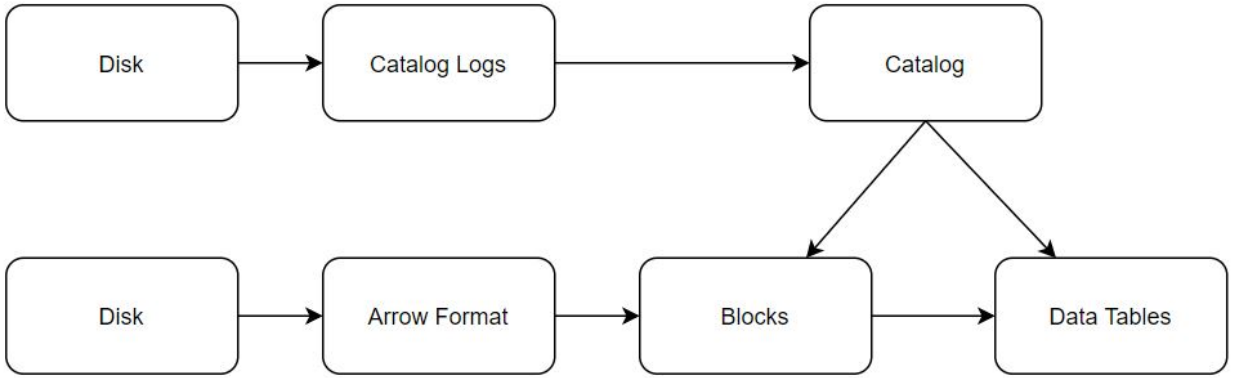
Pipeline

2. Saving Data Tables (Checkpoint)



Pipeline

3. Recover Checkpoint -> Continuous Logging



Goals & Progress

→ **75%: implement user table recovery from the most recent checkpoint.**

→ **100%: enable catalog and user table recovery from checkpoint + continuous logs recovery.**

→ **125%: periodic checkpoints** + performance improvement (e.g. **parallel checkpoint taking**).

Modification

Catalog:

terrier/src/catalog/*

Logging:

terrier/src/storage/write_ahead_log/*

Recovery:

terrier/src/storage/recovery/*

Checkpoint (new):

terrier/src/storage/checkpoint/*

Validation of the Work

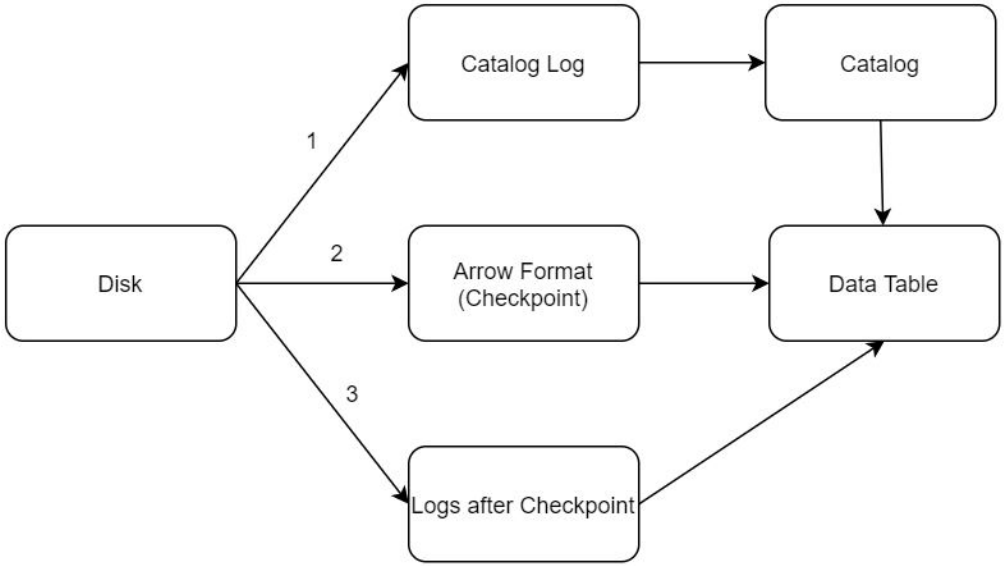
Tests:

`test/storage/recovery_test.cpp`

Performance Measurement:

`benchmark/storage/checkpoint_recovery_benchmark.cpp`

Testing Procedure



Benchmark Results

Recovery with checkpoint:

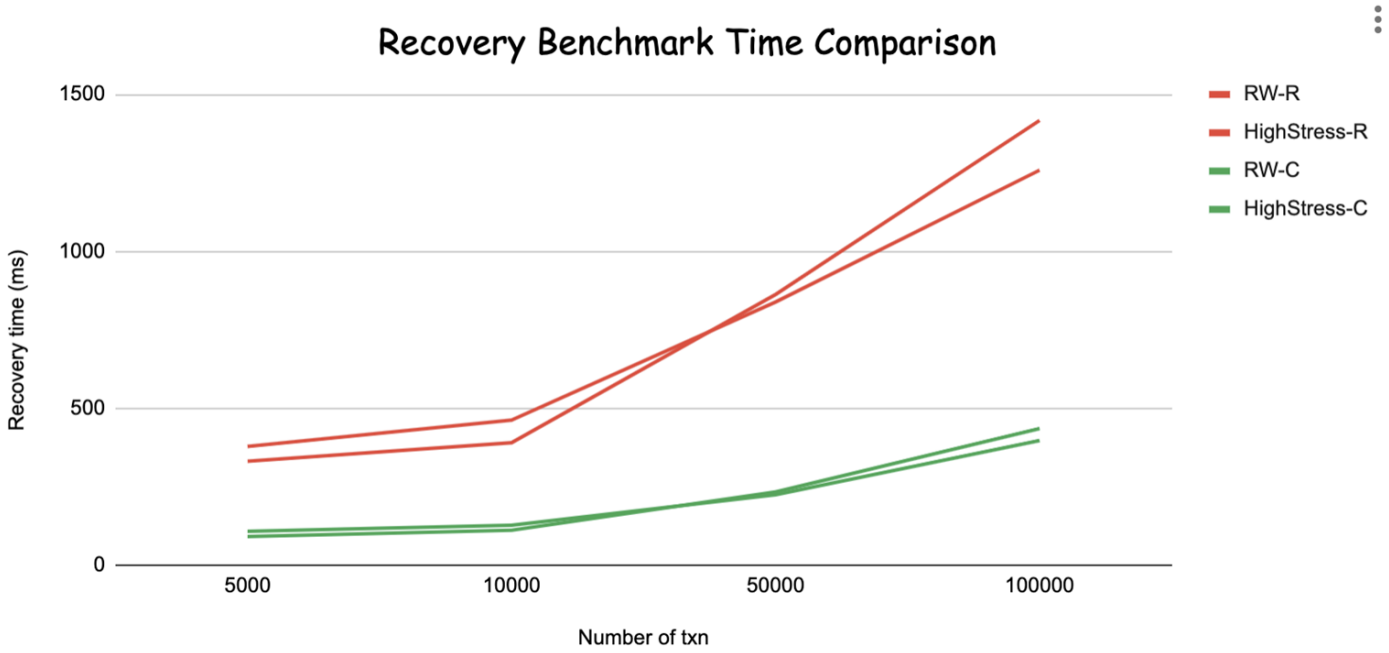
Benchmark	Time	CPU	Iterations	
CheckpointRecoveryBenchmark/ReadWriteWorkload/min_time:10.000/manual_time	804 ms	2144 ms	16	121.425k items/s
CheckpointRecoveryBenchmark/HighStress/min_time:10.000/manual_time	720 ms	1328 ms	19	135.604k items/s

Checkpoint time: 6.9 s

Recovery with log:

Benchmark	Time	CPU	Iterations	
RecoveryBenchmark/ReadWriteWorkload/min_time:10.000/manual_time	3164 ms	1441 ms	6	30.8664k items/s
RecoveryBenchmark/HighStress/min_time:10.000/manual_time	2671 ms	396 ms	5	36.5672k items/s

Benchmark Result



Future work

→ Further collaborate w/ data compression group to ensure the block compaction correctness.

→ Better parallelization schema when taking checkpoint.

Thanks