Auora

1. Insights

AWS provides a service called Auora for OLTP databases. The database engine is forked from InnoDB. Due to the boom in cloud architecture as it offers flexible on demand capacity and compute based on the IT workload. AWS decouples compute and storage to handle issues such as scaling, unreachable host and writer's failure, operational issue of managing the storage fleet. It also gives a brief comparison with salesforce.

Because of the distributed database nature, the typical I/O bottlenecks are moved from compute and storage to the network. There are other synchronous issues to handle such as miss in database buffer cache. Transactions are also expensive because having a chatty algorithm like two phase commit is not feasible in an environment with hard and soft failure.

Auora tries to solve this issues by decoupling redo logging, durable storage, crash recovery, and backup/restore to the storage service as shown in figure 1. By doing so, the network IOPS is reduced significantly. The backup and redo recovery which are synchronous operation in typical architecture are now converted into an asynchronous operation distributed over the nodes. This removes the need for checkpoint, is cheaper since it is asynchronous and continues in the background. Section 2 explores the quorum model. Aurora believes in the policy that data once written, can be read. Since a significant component of the database is offloaded to the storage tier, it is necessary to make them resilient to failure as well. This is typically achieved by redundancy and using quorum based voting(such as paxos?) in distributed environment. Usually for single node, 2/3 are enough. But as explained in the paper, to cater the needs of the availability zones(AZ), they use quorum with 6 votes. It also focuses on reducing MTTR instead of MTTF and has found this parameter successful during the 18 month operational period. Since the model is designed for long term failure, it can easily handle short time segment failure and makes sure only one PG is patched at a time. Section 3 explores the benefits of moving log to the storage tier and the also behind it.It works on minimizing the background write request based on the contention in the network.

2. Comments:

I really like how they provided the bird eye view of Aurora with other AWS services in the section 5.

I also appreciate them having benchmarked on industry standard benchmarks and didn't just give a black box to the readers saying we did this on AWS dataset that cannot be public.

Also really liked the overall paper as it was from a general point of view from a cloud vendor unlike the rocksDB paper which ideally just focused on Facebook(or now meta).