**Week 10: Distributed Transactions**

**Overview**

Running a database instance across multiple nodes requires coordinating data changes to maintain a consistent state of the data. In this module, we'll learn about approaches for coordinating changes across multiple machines, their failure scenarios, and recovery mechanisms.

**Reflection Questions**

* What are transaction managers and coordinators?
* What are the potential types of failures?
* Make sure you understand the two-phase commit (2PC) protocol. In what situations what a transaction be aborted? (Ignore coordinator failures for now – assume all failures / conflicts occur on the follower nodes.)
* If a follower node fails, how does the system recover? If a coordinator fails, how does the system recovery? What are some of the additional challenges if a coordinator fails? (Note the importance of logs.)
* How does distributed consensus improve upon 2PC? What are the advantages and disadvantages?
* What does it mean if reads and writes are linearizable?
* What is the majority protocol? How does quorum consensus alter this? What are the reasons for using quorum consensus?
* What is the downside of quorum consensus for reads versus the majority protocol? What are the approaches for reducing read overhead?
* What are the challenges when reconnecting failed nodes to a database cluster?