UC DAVIS
UNIVERSITY OF CALIFORNIA

MileStone #3

ECS 165 - Sadoghi

By: Kiwi Calvin Bear Squid

Members: Teeranade Cheng, Nikko Sanchez, and Diego Rafael

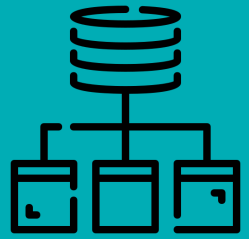
Bug Fixes For Previous Milestone

- Clean up code (lots of technical debt)
- Bufferpool
- Debug index create and drop
- Full scan when index is not available

Our Structure

To introduce multi-statement transactions where either all operations succeed and commit, or none do and the transactions abort

Transaction Semantics



Enable concurrent transactions with serializable isolation using 2PL, avoiding lock waits.

Concurrency Control

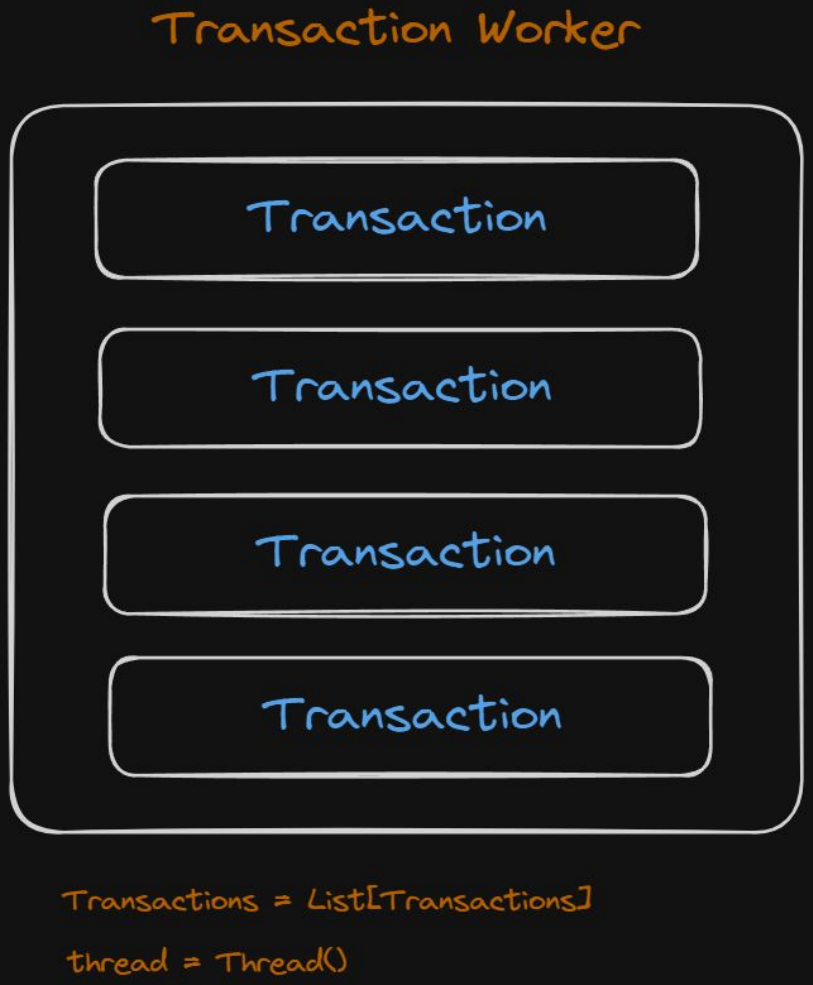
Transaction Semantics Slide(s):

Transaction

- Consists of a list of queries
- If a query fails, the transaction is aborted and rolled back.
- Commit (Write to Disk) if no aborts happen during the transaction.
- W() - Insert, Delete, Update Queries
- R() - Select Query

Transaction Worker

- Initialized with a list of transactions
- Keeps track of the status of transactions
- Single thread for executing all transactions
- Will be running concurrently with other Transaction workers



ACID - for each Transaction

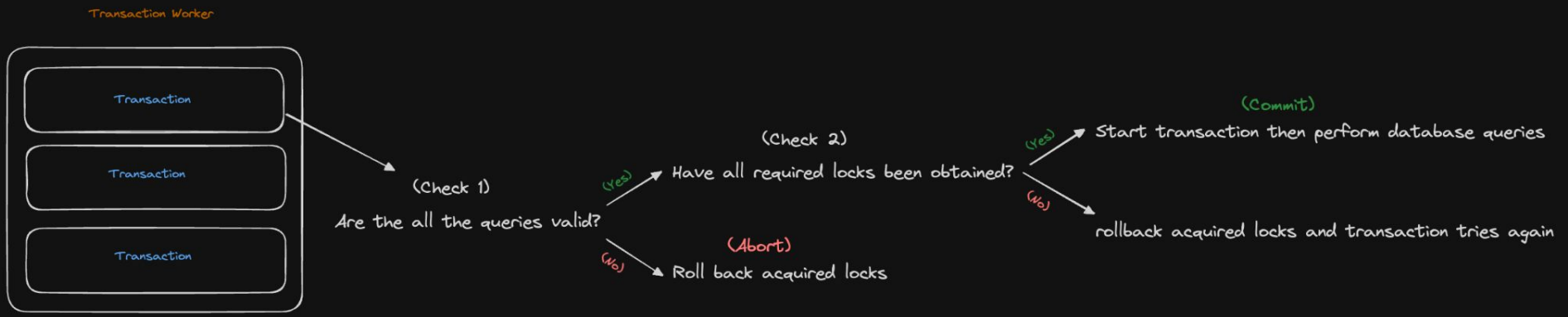
Atomicity: All or nothing

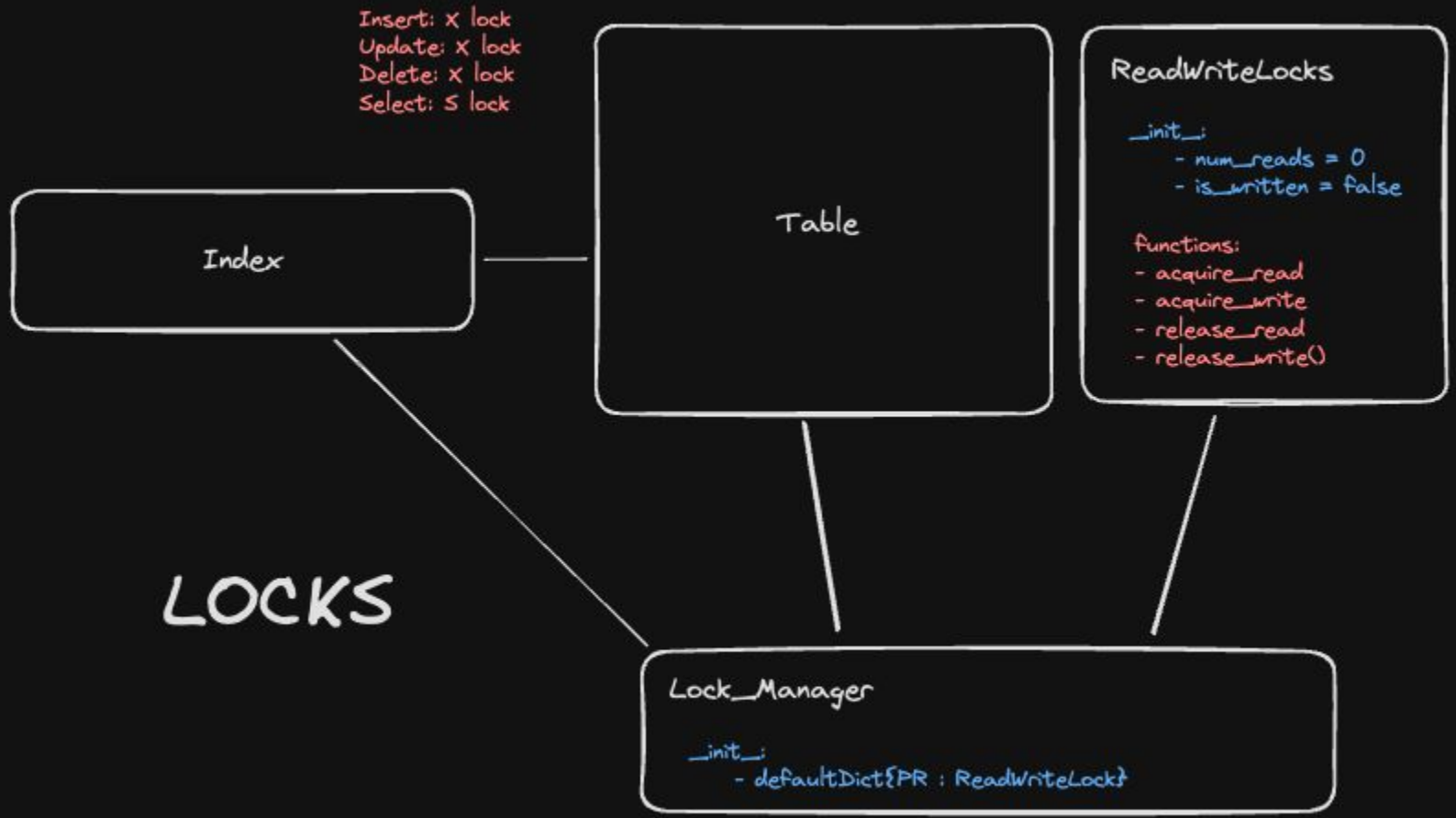
Consistency: ensures that each transaction transforms the database from one valid state to another, maintaining data integrity and adherence to defined constraints and rules.

Isolation: ensures that the concurrent execution of transactions leaves the database in the same state that would have been obtained if the transactions were executed sequentially.

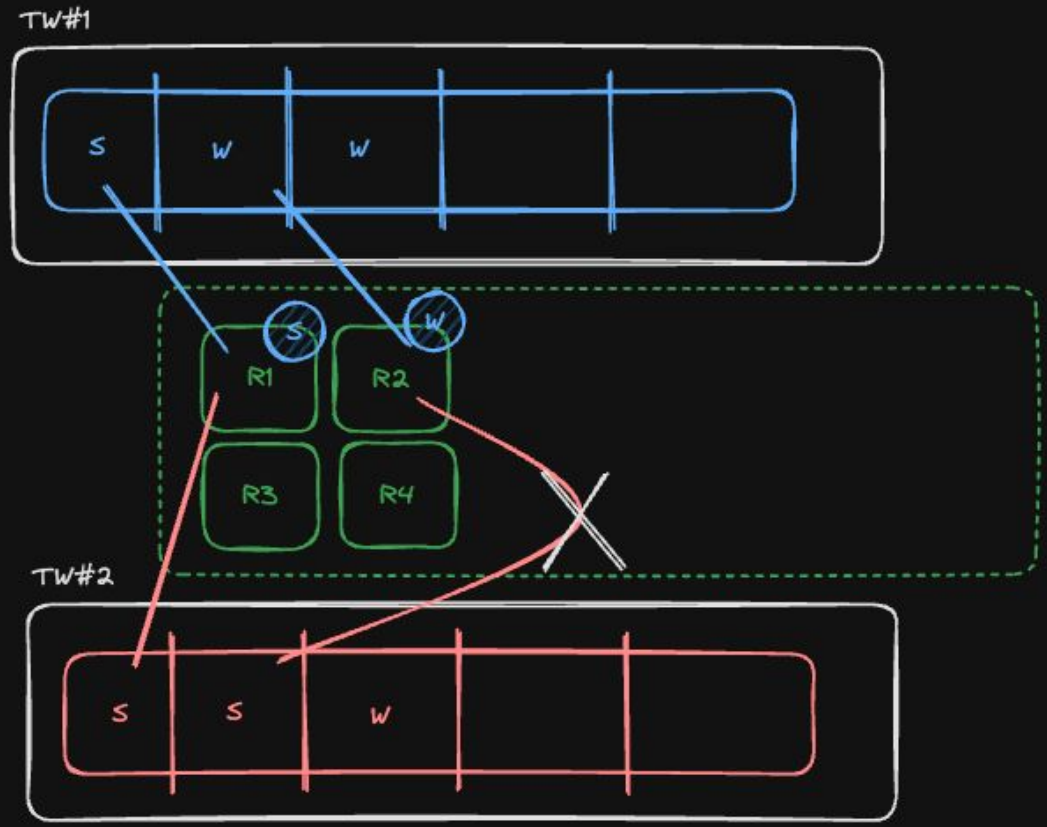
Durability: ensures that, in the event of a system failure, a transaction will stay committed once it has been committed.

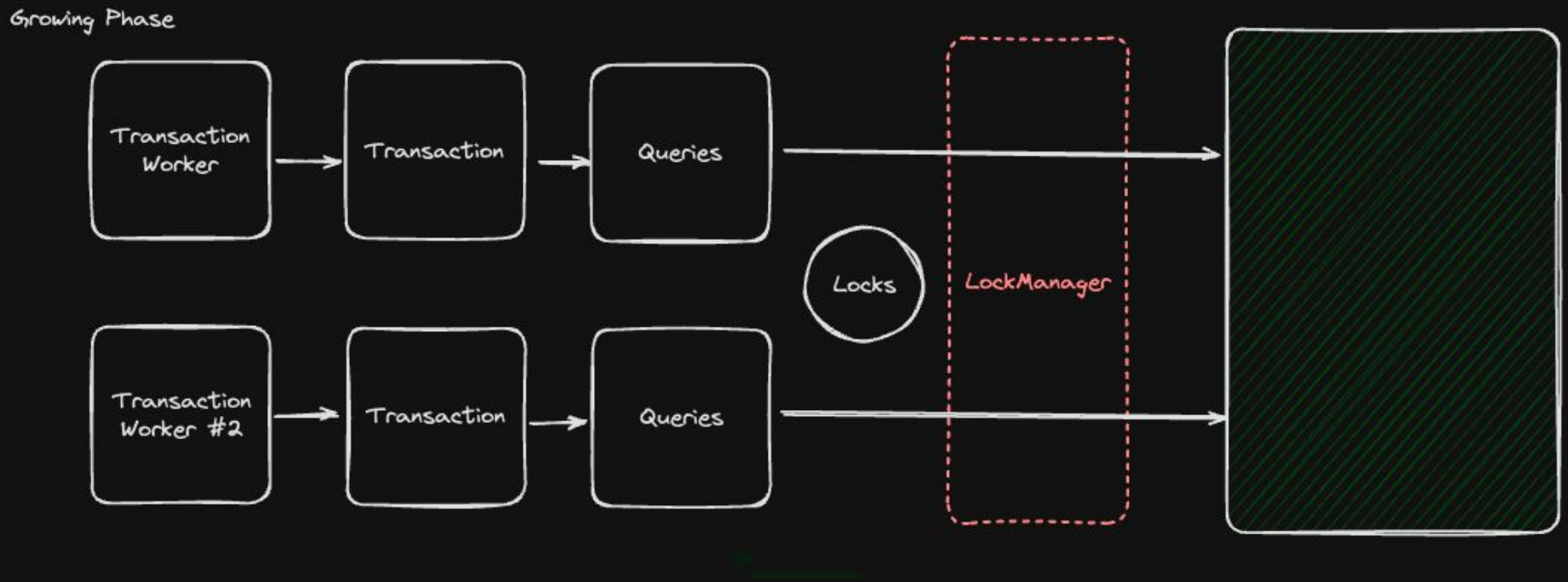
Transaction Worker Pathway



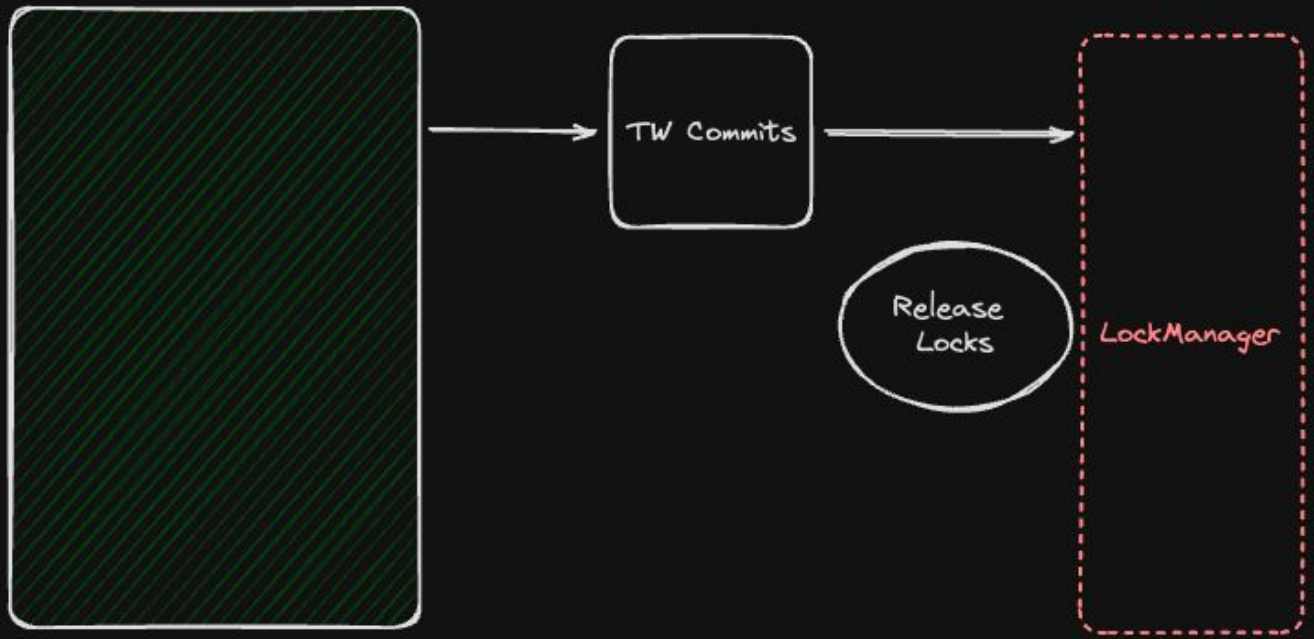

Insert: X lock
Update: X lock
Delete: X lock
Select: S lock
Index
Table
ReadWriteLocks
__init__:
- num_reads = 0
- is_written = false
functions:
- acquire_read
- acquire_write
- release_read
- release_write()
LOCKS
Lock_Manager
__init__:
- defaultDict{PR : ReadWriteLock}

Transaction_Worker threading happens concurrently
Transaction happens sequentially




Growing Phase
Transaction Worker
Transaction
Queries
Transaction Worker #2
Transaction
Queries
Locks
LockManager

Shrinking Phase



Questions (Q/A)!