L-Store: Milestone 3

ECS 165A: Database Systems

Yiling Chen, Tina Young, Olivia Tobin,
Charissa Tseng, Matthew Boentoro

2 Main Parts

1

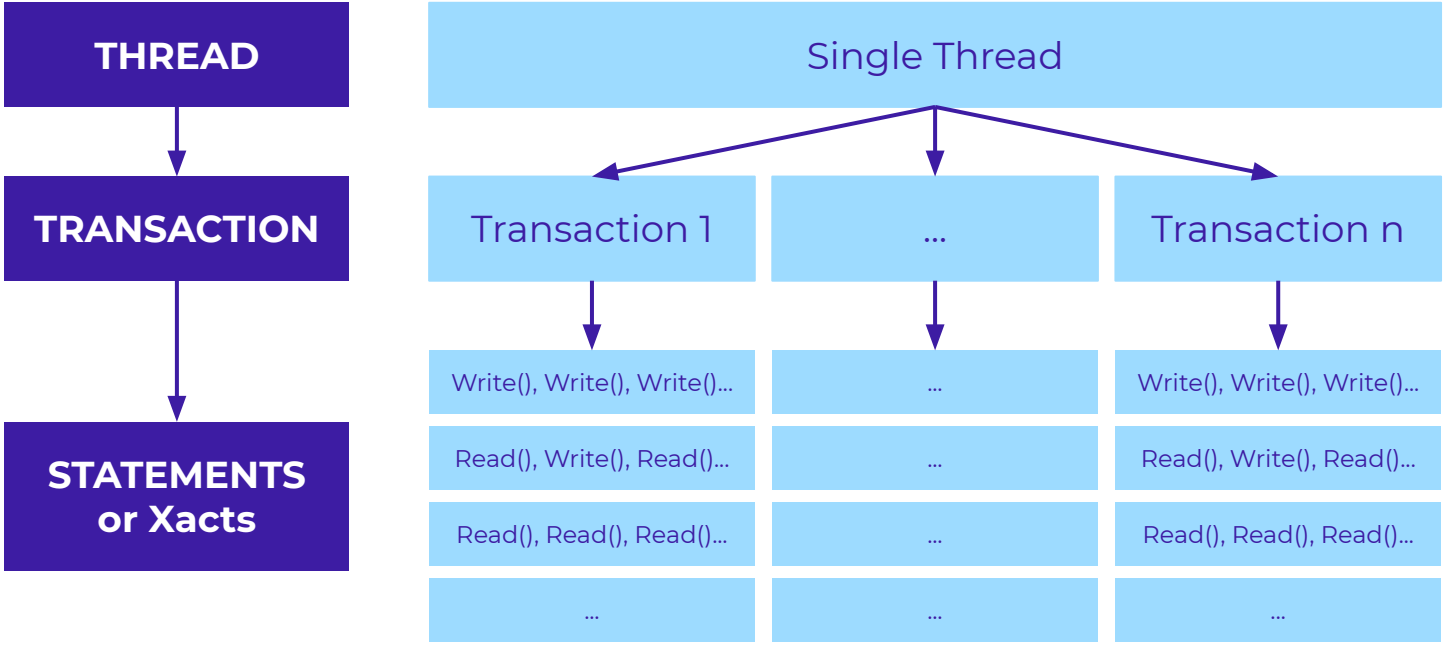
Transaction Semantics (ACID)

2

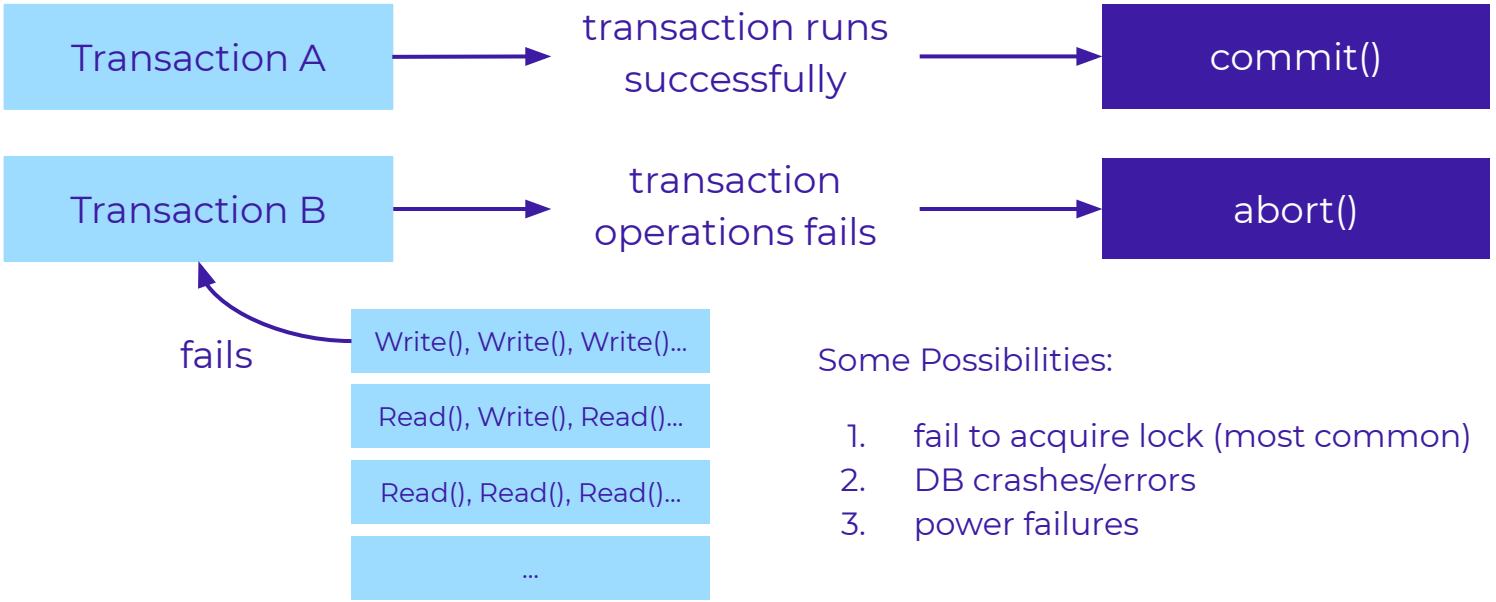
Multithreading Concurrency Control

Transaction Semantics

Threads, Transactions and Xacts



ACID: Atomicity

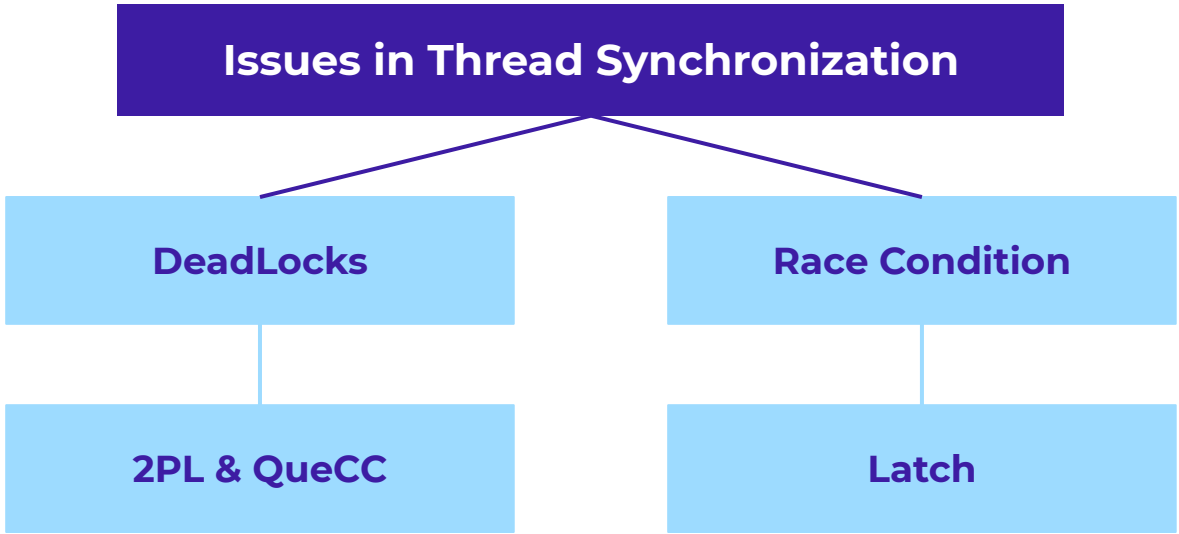


Some Possibilities:

1. fail to acquire lock (most common)
2. DB crashes/errors
3. power failures

Multithreading Concurrency Control

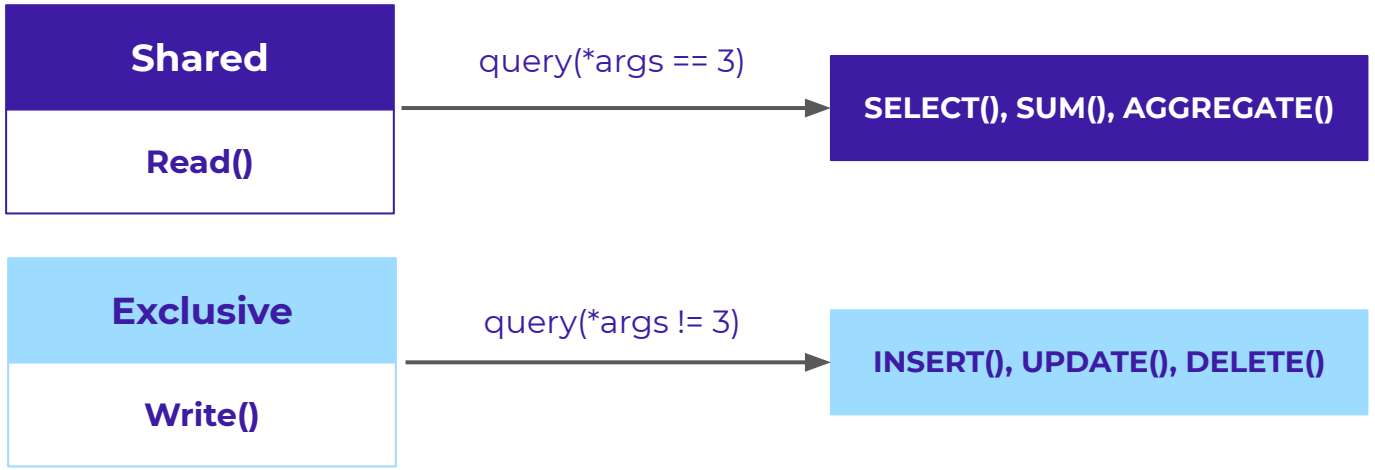
ACID: Isolation



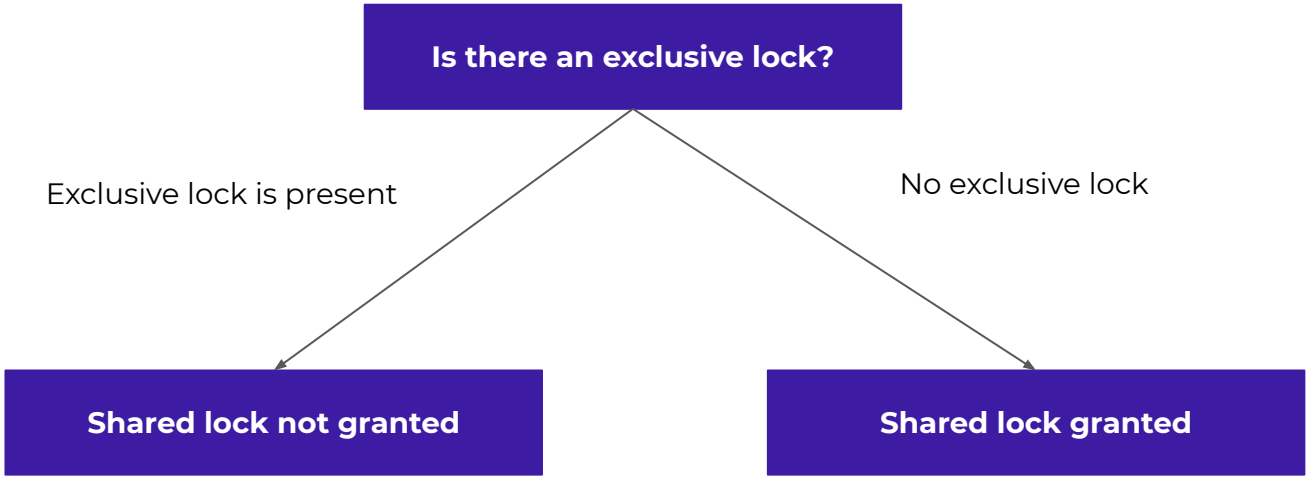
Locks vs Latches

	Locks	Latches
Separate...	User Transaction	Threads
Protect...	DB Content	In-Memory Data Structure
During	Entire Transaction	Critical Section
Kept in...	Lock Manager (Hashmap)	Protected Data Structure

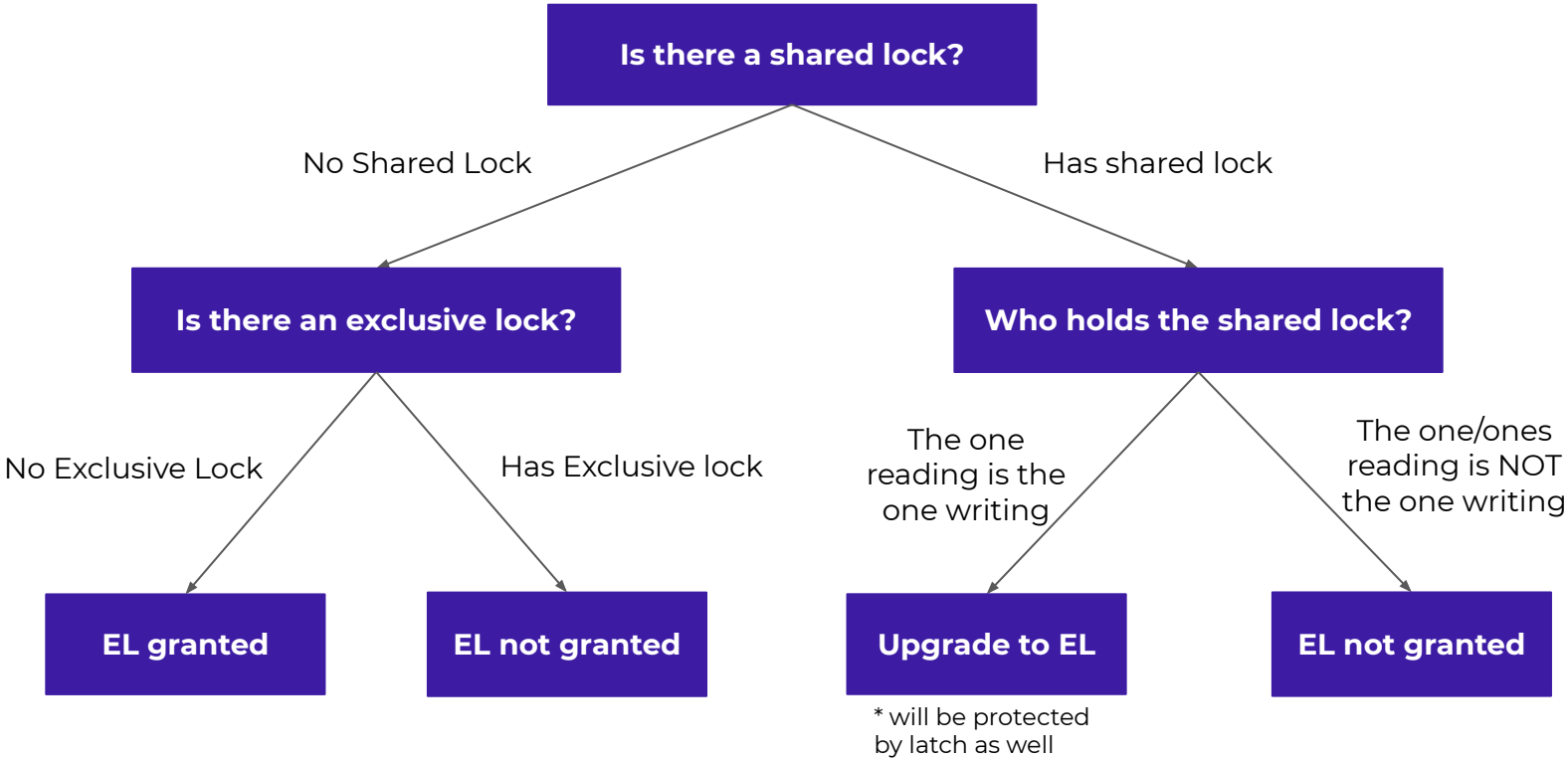
Locks: 2PL


Shared
Read()
query(*args == 3)
SELECT(), SUM(), AGGREGATE()
Exclusive
Write()
query(*args != 3)
INSERT(), UPDATE(), DELETE()

Lock: 2PL (Shared)



Lock: 2PL (Exclusive)



Locks: QueCC

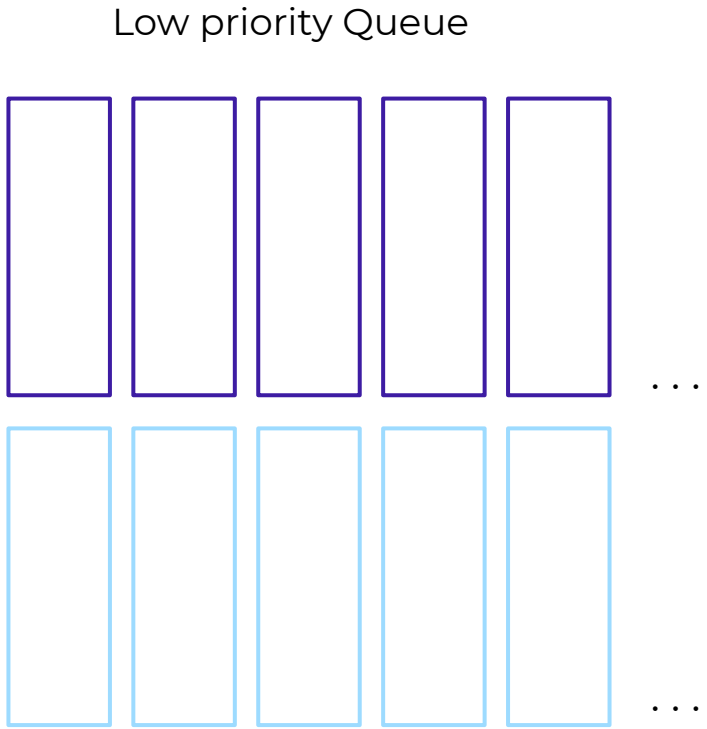
class Planner:

Transaction worker

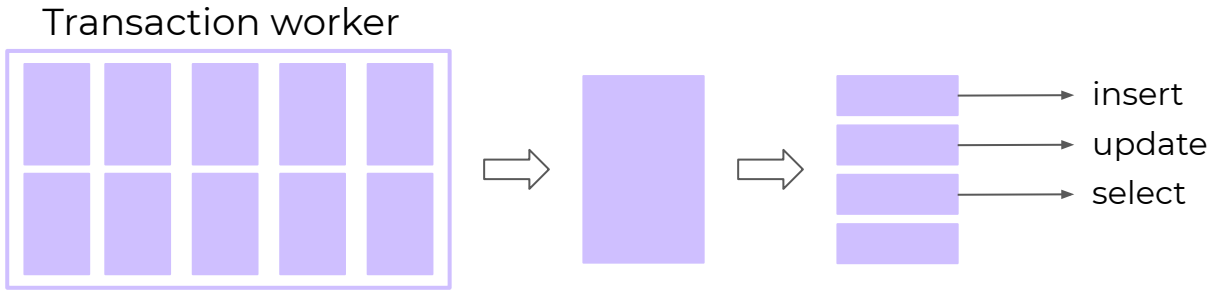
= Transactions

Planning Thread #1

Planning Thread #2



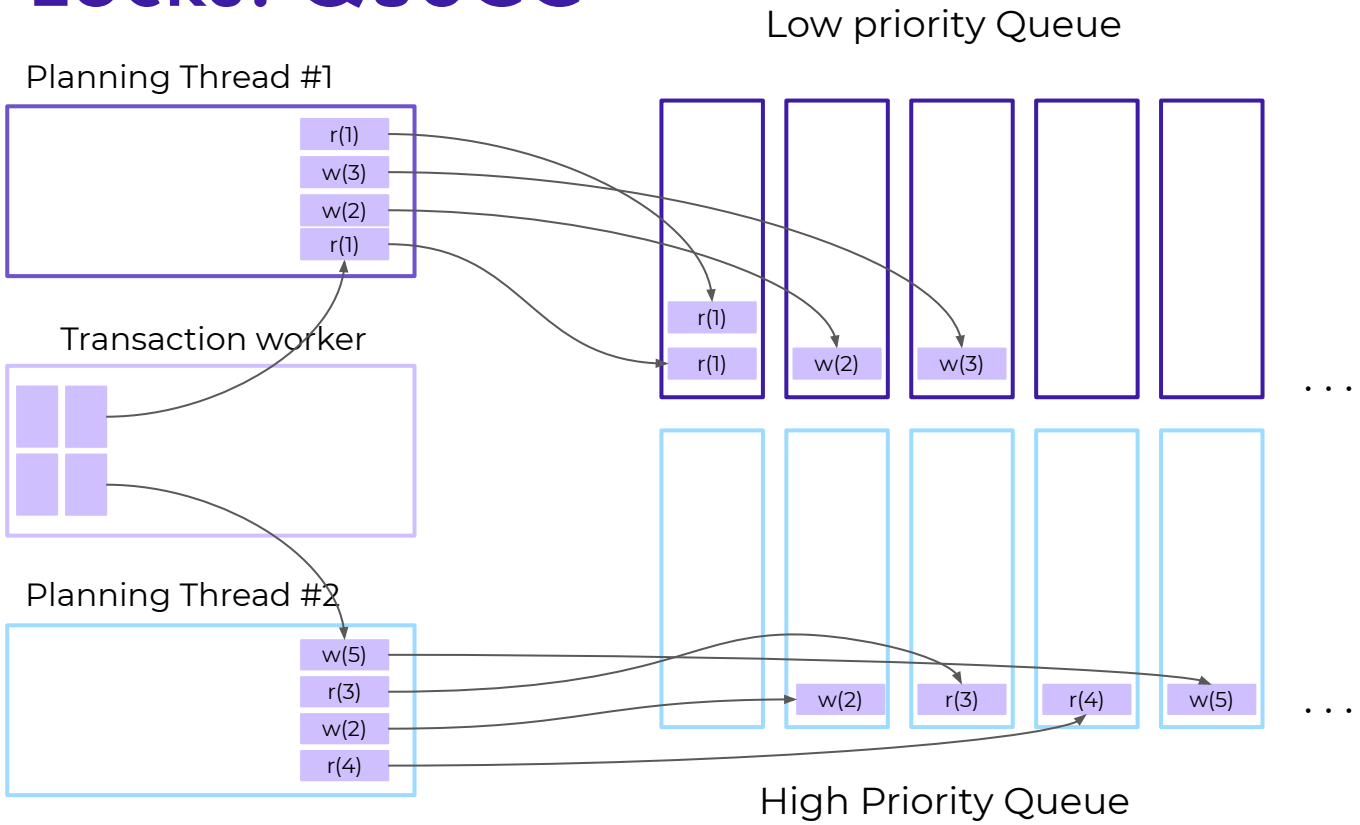
Locks: QueCC



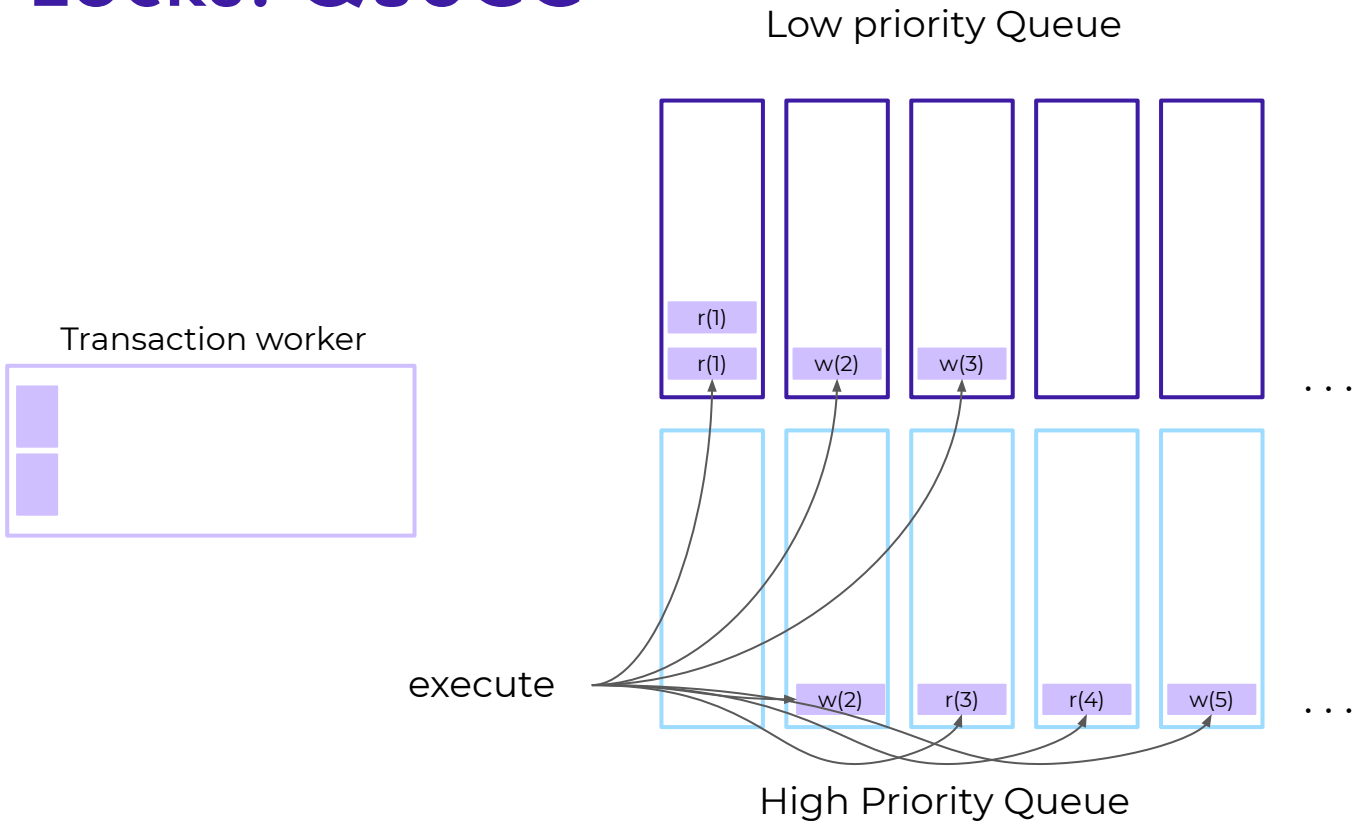
Since Ranking is Arbitrary,
Index in Priority List = RID % 10

Eg: RID = 1001, Index = 1001%10 = 1

Locks: QueCC


Low priority Queue
Planning Thread #1
r(1)
w(3)
w(2)
r(1)
Transaction worker
Planning Thread #2
w(5)
r(3)
w(2)
r(4)
r(1)
r(1)
w(2)
w(3)
w(2)
r(3)
r(4)
w(5)
High Priority Queue

Locks: QueCC


Low priority Queue
Transaction worker
r(1)
r(1)
w(2)
w(3)
execute
w(2)
r(3)
r(4)
w(5)
High Priority Queue

Additional Implementation

Additional Aggregate Functions

max()	min()	avg()	count()
Get the maximum value	Get the minimum value	Get the average value	Get the count*

*does not support multi-thread implementation

Things to Improve Upon

- Writing the program in a different language to support multithreading
  - C++ or Java
- Code writing style
  - Commenting
- Improving algorithm efficiency within functions
  - Removing nested and repeated loops

Thank You!