Data-Intensive Applications

THE BIG IDEAS BEHIND RELIABLE, SCALABLE, AND MAINTAINABLE SYSTEMS

Chapter 7 Part-3: Transactions, Drawbacks of Read Committed and Snapshot Isolation

Preventing Lost Updates

- What a read only transaction can see in the presence of concurrent writes. We have ignored the issue of two transactions writing concurrently.
- Lost update – If application reads some value from the database, modifies it and writes back the modified value(a read-modify-write cycle). If two transactions do this concurrently, one of the modifications can be lost.
- What are different scenarios?
  - Incrementing a counter or updating an account balance.
  - Making a local change to a complex value.
  - Two uses editing the wiki page at the same time.

Ways to prevent lost updates

- Atomic Operations
- Implemented by taking an exclusive lock on the object when it is read so that no other transaction can read it until the update has been applied. This technique is known as cursor stability.
- Or simply force all atomic operations to be executed on a single thread.
- Application to explicitly lock objects

Automatically detecting lost updates

- Allow the transactions to execute in parallel and if the transaction manager detects a lost update, abort the transaction and force it to retry its read-modify-write cycle.
- Databases can perform this check efficiently in conjunction with snapshot isolation.
- Compare and Set – Allow an update to happen only if the value has not changed since you last read it. If the current value does not match what you previously read, the update has no effect and read-modify-write cycle must be retried.

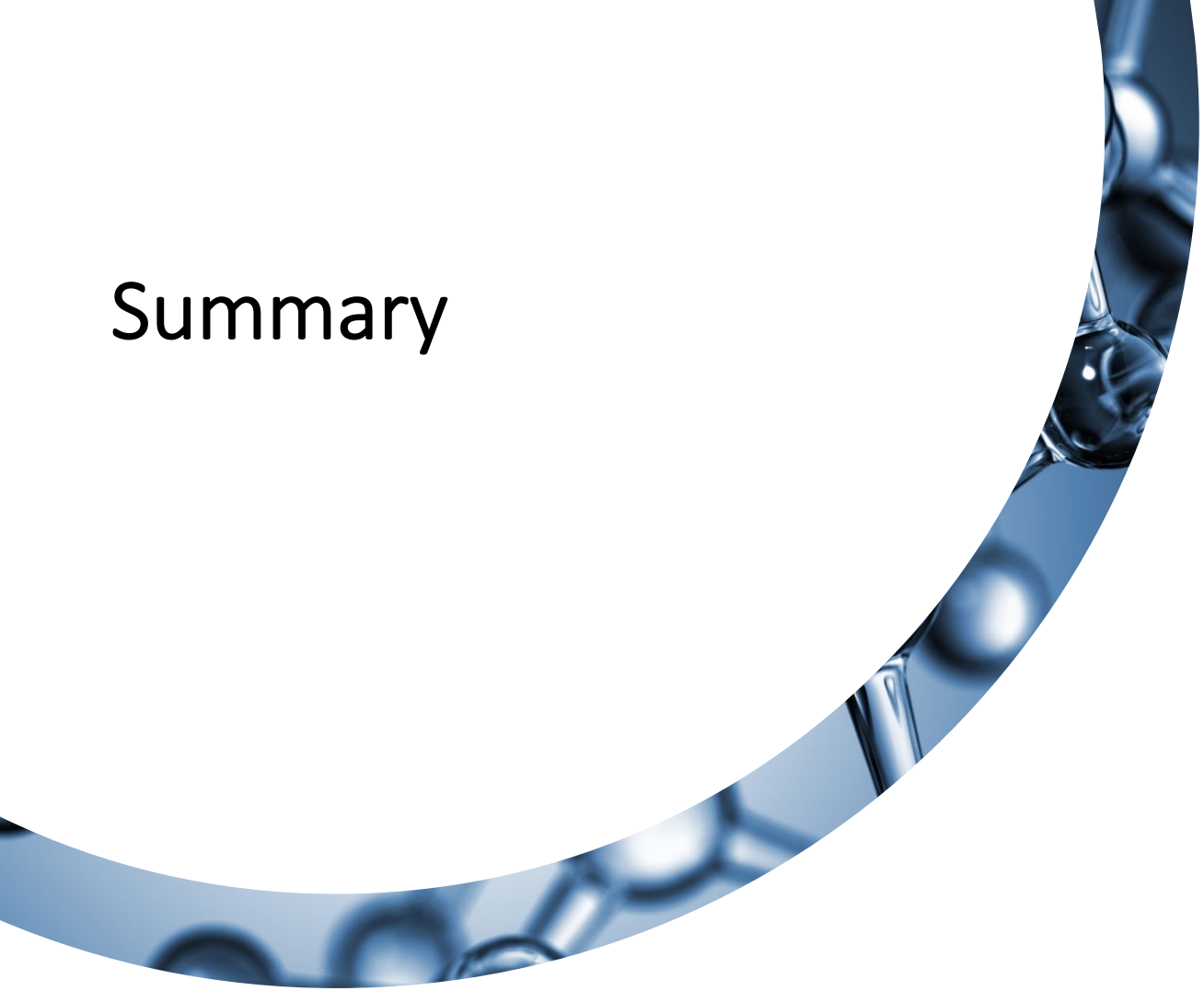
Write Skew and Phantoms

- Dirty writes
- Lost updates
- Write Skew – Two transactions read the same objects and then update some of those objects .
- Atomic single-object operations don’t work, as multiple objects are involved.
- Automatic detection of write skew is not detectable, it requires true serializable isolation.
- Effect where a write in one transaction changes the result set of a search query in another transaction is called phantom.

What next?

- Serializability
  - Isolation levels are hard to understand and inconsistently implemented in different databases
  - Its hard to tell from application, which particular isolation level is safe to run
  - No good tools to detect race conditions
- Three ways:
  - Actual Serial Execution
  - Two Phase Locking
  - Serializable Snapshot Isolation

Summary



Preventing Lost Updates

Ways to prevent lost updates

Automatically detecting lost updates

Write Skews and Phantoms

Serializability

Thank You!