Understanding Transaction Isolation and Serializability

PostgresConf Silicon Valley
April 20, 2023

About me

- Worked with data storage, processing, retrieval technologies for 7 years.
    - Distributed data pipelines (Spark, Flink)
    - Relational databases (mostly PostgreSQL)
    - MPP data warehouses (Redshift)
- Data Engineer at Movable Ink.
- Focus on data platform and infrastructure.
- Second time speaking at a PostgreSQL conference.

Overview

- Agenda
  - Primer on transactions
  - Motivating the problem of isolation
  - Serializability
  - Isolation level: Serializable
- Disclaimers
  - I mostly work *with* databases, not *on* databases.
  - Not focusing on implementation.
- **Goal: Build intuition for the foundational concept of serializability.**

Primer on transactions

- A **transaction** is a unit of work that bundles multiple steps into a single, all-or-nothing operation.
- Transactions have properties that are guaranteed by the database:
  - Atomicity: All-or-nothing.
  - Consistency: Invariants hold.
  - Isolation: Not affected by concurrency.
  - Durability: Committed work persists.
- Allow programmers to make **simplifying assumptions** by adopting transactional model.

The **simplifying assumption** of Isolation:

Each concurrent transaction has the database **all to itself**.

Isolation is easy! 😼

- Suppose Isolation wasn’t guaranteed: who cares?
- **Concurrent transactions** have start and end times that overlap.
- Lots of concurrent transactions don’t require isolation:
    - Read-only transactions.
    - Disjoint read/write sets.
    - Single-statement transactions.

Single-statement transactions

- Simple banking application.
- Alice and Bob share an account with $1,000.
- Two concurrent transactions:
  - T1: Alice makes a deposit of $500 into account A.
  - T2: Bob checks the balance of account A.
- These are concurrent, but don’t need to be isolated.

Multi-statement transactions

- Two concurrent transactions:
  - T1: Alice reads balance B = $1,000.
  - T2: Bob reads balance B = $1,000.
  - T2: Bob withdraws $300, writes new balance of (B - $300) = $700.
  - T1: Alice deposits $500, writes new balance of (B + $500) = $1,500.
- This ordering of T1 and T2 is a **schedule**.
- Free $300, way to go Bob!
- This is the “lost update” problem.
  - Occurs when two or more transactions attempt to write to the same location.

Isolation is hard! 😾

- Forget “concurrent”, we care if transactions are **interleaved**.
- **Interleaved transactions** have statements with alternating orders.
- Lots of “textbook” examples of how things can go wrong.
- Isolation of interleaved transactions is a simplifying assumption!

Why does this matter?

- Let the database worry about it!
- Two responsibilities of applications:
    - Pick an “isolation level”.
    - Deal with the consequences.
- Basics of isolation theory are important.

Handling interleaved transactions

- Goal: Run interleaved transactions without associated problems.
- Naive solution: Don't interleave them, run them "in serial".
  - T1: Alice reads balance B = $1,000.
  - T1: Alice deposits $500, writes new balance of (B + $500) = $1,500.
  - T2: Bob reads balance B = $1,500.
  - T2: Bob withdraws $300, writes new balance of (B - $300) = $1,200.
- This is another **schedule** for the same transactions.
- Hopefully we can do better!

Handling interleaved transactions…better

- Why does executing “in serial” work?
- What if we can replicate the important parts, but still interleave transactions? That’s what **serializability** is for!
- A group of transactions is **serializable** if it “appears” that the transactions ran sequentially.
- We test for serializability by comparing the interleaved schedule to sequential schedules.
- Let’s examine some schedules to see how this works in practice.

Example: read-only transactions

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**

- *r(x = 0)* by T1

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- *r(x = 0)* by T1
- *r(y = 1)* by T2

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- *r(x = 0)* by T1
- *r(y = 1)* by T2
- *r(y = 1)* by T1

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- *$r(x = 0)$* by T1
- *$r(y = 1)$* by T2
- *$r(y = 1)$* by T1
- *$r(x = 0)$* by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- *$r(x = 0)$* by T1
- *$r(y = 1)$* by T1
- *$r(y = 1)$* by T2
- *$r(x = 0)$* by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 1)$**

Example: read-only transactions

Schedule A (interleaved):

- $(x, y) = (0, 1)$
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- $(x, y) = (0, 1)$
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Schedule C (T2 → T1):

- $(x, y) = (0, 1)$
- $r(y = 1)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 1)$**
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Example: read-only transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 1)$**
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2
- $r(x = 0)$ by T1

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Schedule C (T2 → T1):

- **(x, y) = (0, 1)**
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1

Example: read-only transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $r(y = 1)$ by T1
- $r(x = 0)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (0, 1)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2

Schedule C (T2 → T1):

- **(x, y) = (0, 1)**
- $r(y = 1)$ by T2
- $r(x = 0)$ by T2
- $r(x = 0)$ by T1
- $r(y = 1)$ by T1

Example: disjoint transactions

Example: disjoint transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**

Example: disjoint transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**

- *r(x = 0)* by T1

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **(x, y) = (1, 1)**
- *r(x = 0)* by T1
- *r(y = 1)* by T2
- *w(x = x + 1)* by T1

Example: disjoint transactions

Schedule A (interleaved):

- **(x, y) = (1, 2)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**
- $r(x = 0)$ by T1

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 1)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 1)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 1)$**

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 1)$**
- $r(y = 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **(x, y) = (1, 2)**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (1, 2)**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **(x, y) = (0, 2)**
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (0, 2)$**
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $r(x = 0)$ by T1

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (1, 2)$**
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1

Example: disjoint transactions

Schedule A (interleaved):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $r(y = 1)$ by T2
- $w(x = x + 1)$ by T1
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (1, 2)$**
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2

Schedule C (T2 → T1):

- **$(x, y) = (1, 2)$**
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $r(x = 0)$ by T1
- $w(x = x + 1)$ by T1

Example: dependent transactions

Example: dependent transactions

Schedule A (interleaved):

- **(x, y) = (0, 1)**

Example: dependent transactions

Schedule A (interleaved):

- **(x, y) = (2, 1)**

- *w(x = 2)* by T1

Example: dependent transactions

Schedule A (interleaved):

- $(x, y) = (2, 1)$
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- **(x, y) = (2, 3)**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 3)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (0, 1)$**

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 1)**
- *w(x = 2)* by T1

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (2, 3)$**
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1

Example: dependent transactions

Schedule A (interleaved):

- **(x, y) = (2, 4)**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **(x, y) = (2, 3)**
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 3)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Example: dependent transactions

Schedule A (interleaved):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $w(y = 3)$ by T1
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Schedule B (T1 → T2):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (0, 1)**

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (0, 1)**
- *r(x = 0)* by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (0, 1)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)*** **= (0, 2)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)*** **= (2, 2)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2
- *w(x = 2)* by T1

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (2, 3)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2
- *w(x = 2)* by T1
- *w(y = 3)* by T1

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (2, 3)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2
- *w(x = 2)* by T1
- *w(y = 3)* by T1

Serializability

- A concurrent schedule is **serializable** if its effects are equivalent to at least one serial schedule of the the same transactions.

- If a schedule is serializable, then it preserves the correctness of serial execution, while still allowing the efficiency of interleaved execution.

- Serializability is the “gold standard” of transaction isolation.

Isolation level: Serializable

- **Isolation levels** are configurable, and determine how the database handles concurrent transactions
- Under the **Serializable isolation level**, the database guarantees that only serializable schedules of concurrent transactions are allowed to commit.
- Different enforcement mechanisms:
  - Two-phase locking (2PL)
  - Multi-version concurrency control (MVCC) ← (what PostgreSQL uses)
- Not implemented by all databases, and rarely the default isolation level.

Serializable isolation is hard! 😾

- Database will monitor for non-serializable schedules, and abort transactions.
- For long-running transactions, this can be really expensive.
- Requires retry logic in application.
- It’s easy to construct non-serializable schedules!

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule B (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Schedule C (T2 → T1):

- ***(x, y)* = (2, 3)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2
- *w(x = 2)* by T1
- *w(y = 3)* by T1

Example: dependent transactions

Schedule A (interleaved):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (0, 1)**

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 1)**
- *w(x = 2)* by T1

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 1)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 1)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 3)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Serializable schedule:

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *w(y = 3)* by T1
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)*** **= (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- (x, y) = **(2, 2)**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- (x, y) = **(0, 1)**

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)* = (2, 1)**
- *w(x = 2)* by T1

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)* = (2, 3)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)*** **= (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)*** **= (2, 3)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)* = (2, 3)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)* = (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)* = (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- $(x, y) = (2, 2)$
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- $(x, y) = (2, 4)$
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Serial schedule (T2 → T1):

- $(x, y) = (0, 1)$

Example: non-serializable schedule

Non-serializable schedule:

- *(x, y)* = (2, 2)
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- *(x, y)* = (2, 4)
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Serial schedule (T2 → T1):

- *(x, y)* = (0, 1)
- *r(x = 0)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- (x, y) **= (2, 2)**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- (x, y) **= (2, 4)**
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Serial schedule (T2 → T1):

- (x, y) **= (0, 1)**
- $r(x = 0)$ by T2
- $r(y = 1)$ by T2

Example: non-serializable schedule

Non-serializable schedule:

- ***(x, y)*** **= (2, 2)**
- *w(x = 2)* by T1
- *r(x = 2)* by T2
- *r(y = 1)* by T2
- *w(y = 3)* by T1
- *w(y = y + 1)* by T2

Serial schedule (T1 → T2):

- ***(x, y)*** **= (2, 4)**
- *w(x = 2)* by T1
- *w(y = 3)* by T1
- *r(x = 2)* by T2
- *r(y = 3)* by T2
- *w(y = y + 1)* by T2

Serial schedule (T2 → T1):

- ***(x, y)*** **= (0, 2)**
- *r(x = 0)* by T2
- *r(y = 1)* by T2
- *w(y = y + 1)* by T2

Example: non-serializable schedule

Non-serializable schedule:

- $(x, y) = (2, 2)$
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- $(x, y) = (2, 4)$
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Serial schedule (T2 → T1):

- $(x, y) = (2, 2)$
- $r(x = 0)$ by T2
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $w(x = 2)$ by T1

Example: non-serializable schedule

Non-serializable schedule:

- **$(x, y) = (2, 2)$**
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- **$(x, y) = (2, 4)$**
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Serial schedule (T2 → T1):

- **$(x, y) = (2, 3)$**
- $r(x = 0)$ by T2
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1

Example: non-serializable schedule

Non-serializable schedule:

- $(x, y) = (2, 2)$
- $w(x = 2)$ by T1
- $r(x = 2)$ by T2
- $r(y = 1)$ by T2
- $w(y = 3)$ by T1
- $w(y = y + 1)$ by T2

Serial schedule (T1 → T2):

- $(x, y) = (2, 4)$
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1
- $r(x = 2)$ by T2
- $r(y = 3)$ by T2
- $w(y = y + 1)$ by T2

Serial schedule (T2 → T1):

- $(x, y) = (2, 3)$
- $r(x = 0)$ by T2
- $r(y = 1)$ by T2
- $w(y = y + 1)$ by T2
- $w(x = 2)$ by T1
- $w(y = 3)$ by T1

If not Serializable isolation, then what?

- ANSI SQL standard defines three other isolation levels:
  - Read Uncommitted: “Dirty reads” possible.
  - Read Committed: “Non-repeatable reads” possible. ← (Default in PostgreSQL)
  - Repeatable Read: “Phantom reads” and “write skew” possible. (Also called “Snapshot Isolation”)
- Each allows some non-serializable schedules to commit.
- Important to understand requirements of your application.
- Serializability is isolation “done right”. Everything else is **sacrificing correctness**.

Best practices for Serializable isolation

- Limit the “surface area” of transactions.
- Write to database objects in a “canonical order”.
- Use fine-grained locking tools like **SELECT FOR UPDATE**.
- Implement retry logic.

Key takeaways

1. The goal of serializable isolation is to achieve interleaved execution of transactions without sacrificing correctness.
2. You can check the serializability of a schedule by comparing it to serial schedules of the same transactions.
3. Serializability is fundamental to understanding isolation levels.
4. Other isolation levels deviate from serializability in important ways.

Questions?