Before-or-after & Serializability &Transaction

> **Consistency can break under fault, even for the strongest consistency model**
>
> **Consistency can break under concurrency, even for the strongest consistency model +all-or-nothing atomicity**

- **The race condition problem（P18）**
- **Race condition is hard to control（P19）**

Before-or-After Atomicity

- **Goal: before-or-after atomicity（P24、25）**
- **Use locking to achieve before-or-after: global lock（P27、40）**
- **Use locking to achieve before-or-after: fine-grained locking（P43-44）**
  - 可能会出现的问题：**accessing multiple records**（P45）
  - 解决方案：**acquire all locks first, and release them at last（P62）**
- **Two-phase locking (2PL)（P63-64）**
  - benefit（P74-75）

Serializability

- 定义（P77）
- **Serializability types（P81）**

Conflict Serializability

- **conflict 和Conflict Serializability的定义（P83)**
- **Conflict Graph(P85)**

View Serializability(P91-92)

- 各类serializability的关系（P93）
- **Conflict Serializability VS. View Serializability（P94）**
- **Use locking to achieve before-or-after: 2PL（P95）**

Serializability simplifies enforcing concurrent correctness / data consistency

- **Correctness (app-semantic consistency)（P98）**
- **Correctness and serializability（P99）**
- **Why serializability is ideal（lec next P11-12)**

drawbacks of 2PL

- **Deadlock（P101）**

    - example（P102-P106）
- **Resolving deadlock（P107）**
- data management in a distributed setting（P108）

Transactions

- 概述（P110）
- **TX: an abstraction to ease data management（P111）**
- **What can a TX provide? ACID（P112-116）**
- **Enforcing ACID properties（P117）**
- Why we need to use a TX（next lec P14）

Is 2PL sufficient to guarantee serializability?
ans: Depends on whether the lock is properly held（正确拿锁放锁）