# Appunti Basi di Dati

Luca Seggiani

24 Maggio 2024

**2PL e CSR**

Ogni schedule $2PL$ è anche conflict-serializzabile, ma non necessariamente viceversa. Ad esempio, la schedule $r_1(x)w_1(x)r_2(x)w_2(x)r_3(y)w_1(y)$ viola il $2PL$, ma è conflict-serializzabile. Questo teorema si può dimostrare. Come? cazzi tui Si ha, in definitiva, che le schedule $2PL$ formano un'ulteriore sottoinsieme, per la presciisione delle $CSR$, e contengono le schedule seriali $S'_s$. Riassumendo, si ha:

$$S'_s \subset 2PL \subset CSR \subset VSR \subset S$$

**2PL e anomalie**

La 2PL risolve le anomalie di perdita di aggiornamento, aggiornamento fantasma e letture inconsistente. Presenta pero altre anomalie:

- **Cascading rollback**
  Il fallimento di una transazione che ha scritto una risorsa deve causare il fallimento di tutte le transazioni che hanno letto il valore scritto.

- **Deadlock** (attese incrociate o stallo)
  Due transazioni possono detenere una risorsa e aspettare la risorsa detenuta dall'altra. In generale, la possibilità di deadlock è bassa, ma non nulla.

**Locking a due fasi stretto**

Una variante del lock a due fasi prevede una condizione aggiuntive, quella di fornire il rilascio dei lock solo dopo il commit. Questo elimina il rischio di letture sporche e quindi di rollback in cascata, e supera l'ipotesi di commit-proiezione.