Basi di dati, primo modulo
Prova di autovalutazione del 26 maggio 2004

Domanda 1
Considerare i seguenti schedule:

1. $r_1(y)r_4(z)r_4(y)w_4(y)w_4(z)r_1(z)r_2(z)r_3(z)w_3(z)$
2. $w_1(z)r_2(z)r_3(z)w_3(z)w_3(y)$
3. $r_1(x)r_2(z)w_2(z)w_1(x)r_2(x)w_2(x)$
4. $r_2(x)w_2(x)r_3(x)w_3(x)r_1(y)w_2(y)$
5. $r_4(z)r_4(y)w_4(y)w_4(z)r_2(x)w_2(x)r_1(x)w_1(x)$

Specificare, con una breve giustificazione, a quali delle seguenti classi ciascuno di essi appartiene: S (seriale), VSR (view-serializzabile), CSR (conflict-serializzabile), 2PL (generabile da uno scheduler basato sul lock a due fasi) e TS (generabile da uno scheduler che utilizzi il metodo dei timestamp; si assuma che l'ordinamento degli identificatori delle transazioni corrisponda a quello dei timestamp).

Domanda 2
Il controllo della concorrenza deve tener conto anche delle pagine degli indici.

1. Spiegare perché il 2PL, certamente corretto, risulta estremamente inefficiente se applicato ai B-tree.
2. Considerare il seguente protocollo, chiamato *tree protocol* (per semplicità riferirsi solo a lock esclusivi):
   - su ciascun nodo, a parte la radice, può ottenere un lock solo una transazione che abbia il lock sul nodo genitore
   - non è possibile per una transazione ottenere due volte un lock sullo stesso nodo

   Si noti che è possibile rilasciare il lock sul genitore dopo aver ottenuto quello sul figlio e che non è imposta la condizione di 2PL.
   Spiegare, intuitivamente, perché il tree protocol garantisce la serializzabilità e perché esso risulta più efficiente di un protocollo 2PL.

Domanda 3
Il check-point, nei vari DBMS, viene realizzato in due modi diversi:

1. in alcuni sistemi si prende nota delle transazioni attive e si rifiutano (momentaneamente) nuovi commit
2. in altri si inibisce l'avvio di nuove transazioni e si attende invece la conclusione (commit o abort) delle transazioni attive

Spiegare, intuitivamente, le differenze che ne conseguono sulla gestione delle riprese a caldo.

Domanda 4 Si consideri una base di dati relativa ad un sistema di prenotazione (per viaggi aerei), con le relazioni:

- *Prenotazioni(<u>Numero</u>,Volo,Data,NumeroPosti)*
- *Disponibilità(<u>Volo</u>,<u>Data</u>,PostiTotali,PostiDisponibili)*

in cui il valore dell'attributo *PostiDisponibili* è, per ciascun volo (e giorno), pari alla differenza fra *PostiTotali* e la somma dei posti prenotati su tale volo. Sulla relazione *Prenotazioni* sono effettuati inserimenti ed eliminazioni, ma non modifiche.
Descrivere (senza preoccuparsi dei dettagli sintattici) una o più regole attive che permettano di mantenere corretto il valore di *PostiDisponibili* a fronte di inserimenti ed eliminazioni nella relazione *Prenotazioni*.