Basi di Dati: Complementi

Docente: Prof. Pierangela Samarati

Appello di Maggio online – 07 Maggio 2010

Tempo a disposizione 2:00h

Soluzioni

Domanda 1)
Caratterizzare in modo preciso, completo ed esauriente il *modello multidimensionale* dei dati e fornirne un esempio. Descrivere quindi le operazioni di *slice-and-dice*, *roll-up* e *drill-down* fornendo per ciascuna di queste un esempio.

Domanda 2)
Rispondere in modo preciso e completo alle seguenti domande.

1. Descrivere le regole *Write Ahead Log* e *Commit-Precedenza* e dire perché servono.
2. Elencare e descrivere brevemente gli *indicatori di ordinamento* utilizzati in *XML schema*.
   Dato poi il seguente XML schema:

```
<?xml version="1.0"?>
<schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="anagrafica">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Cognome" type="xs:string" />
        <xs:element name="Nome" type="xs:string" maxOccurs="2" minOccurs="1" />
      </xs:sequence>
    </xs:complexType>
    <xs:attribute name="data" type="xs:date" use="required" />
  </xs:element>
</schema>
```

   dire se il seguente file XML è valido rispetto allo schema, giustificando la risposta.

```
<?xml version="1.0"?>
<anagrafica data="15/04/2010">
  <Nome> Andrea </Name>
  <Nome> Mario </Name>
  <Cognome> Rossi </Cognome>
</anagrafica>
```

Foglio per l'Domanda 3)

Sia data una sequenza di azioni da parte di *transazioni concorrenti*.
Si indichi per ciascuna affermazione riportata nella tabella allegata se è vera per qualsiasi istanza (*Vero*), falsa per qualsiasi istanza (*Falso*), o se *Non è possibile determinarlo* (potrebbe essere vera oppure falsa).

1. Se va a buon fine in un sistema basato su *TS multiversione*, allora va a buon fine anche in un sistema basato su *TS monoversione*.

   ☐ Vero
   ☐ Falso
   ☑ Non è possibile determinarlo

2. Se va a buon fine in un sistema basato su *2PL base*, allora va a buon fine anche in un sistema basato su *2PL stretto*.

   ☐ Vero
   ☐ Falso
   ☑ Non è possibile determinarlo

3. Se va a buon fine in un sistema basato su *2PL con lock a 2 stati*, allora va a buon fine anche in un sistema basato su *2PL con lock a 3 stati*.

   ☑ Vero
   ☐ Falso
   ☐ Non è possibile determinarlo

4. Se va a buon fine in un sistema basato su *2PL*, allora va a buon fine anche in un sistema basato su *TS*.

   ☐ Vero
   ☐ Falso
   ☑ Non è possibile determinarlo

5. Se va a buon fine in un sistema basato su *TS*, allora va a buon fine anche in un sistema basato su *2PL*.

   ☐ Vero
   ☐ Falso
   ☑ Non è possibile determinarlo

6. Se va a buon fine in un sistema basato su *CSR*, allora va a buon fine anche in un sistema basato su *2PL*.

   ☐ Vero
   ☐ Falso
   ☑ Non è possibile determinarlo

Foglio per la Domanda 4)

Si considerino i tre insiemi di oggetti X, Y e Z e le corrispondenti regole di associazione. Indicare (scegliendo la corrispondente casella ☑) quale delle affermazioni è vera, basandosi sulle relazioni fra *supporto* e *confidenza*.

Si **ricorda** di scegliere (☑) **solo una** delle risposte per ogni affermazione.

- Supponendo che: supporto($X \rightarrow Y$) > supporto($Z \rightarrow Y$)
  - ☐ X compare in più transazioni di Z
  - ☑ Non è possibile determinarlo
  - ☐ Non è possibile
- Supponendo che: supporto($\{X, Z\} \rightarrow Y$) > supporto($X \rightarrow Y$)
  - ☐ Z compare in più transazioni di X
  - ☐ Non è possibile determinarlo
  - ☑ Non è possibile
- Supponendo che: confidenza($X \rightarrow Y$) > confidenza($Z \rightarrow Y$)
  - ☐ X compare in più transazioni di Z
  - ☑ Non è possibile determinarlo
  - ☐ Non è possibile
- Supponendo che: confidenza($\{X, Y\} \rightarrow Z$) = confidenza($\{Y, Z\} \rightarrow X$)
  - ☑ X compare nello stesso numero di transazioni di Z
  - ☐ Non è possibile determinarlo
  - ☐ Non è possibile

Esercizio 1)
Avendo le seguenti informazioni riguardo una **ripresa a caldo**, si indichi per ciascuna affermazione riportata nella tabella allegata se è sicuramente vera (*Vero*), sicuramente falsa (*Falso*), o se *non è possibile determinarlo* (potrebbe essere vera oppure falsa).

- record di checkpoint: CK(T2, T3, T5);
- insieme di UNDO: {T3, T6, T7};
- insieme di REDO: {T2, T5};
- operazioni di UNDO:
  - D(T7, O7, B7) → INSERT(O7), O7 := B7
  - D(T6, O6, B6) → INSERT(O6), O6 := B6
  - U(T3, O3, B3, A3) → O3 := B3
- operazioni di REDO:
  - U(T2, O2, B2, A2) → O2 := A2
  - I(T5, O8, A8) → INSERT(O8), O8 := A8
  - I(T5, O5, A5) → INSERT(O5), O5 := A5

Foglio per l'Esercizio 1)

Si indichi per ciascuna affermazione riportata nella tabella allegata se è sicuramente vera (*Vero*), sicuramente falsa (*Falso*), o se *non è possibile determinarlo* (potrebbe essere vera oppure falsa).

1. nel log esistono esattamente due operazioni da parte della transazione T5
   - ☑ Vero
   - ☐ Falso
   - ☐ Non è possibile determinarlo
2. la transazione T2 è iniziata (*begin transaction*) prima del *checkpoint* e ha fatto *abort* dopo il *checkpoint*
   - ☐ Vero
   - ☑ Falso
   - ☐ Non è possibile determinarlo
3. la transazione T7 è iniziata (*begin transaction*) dopo che la transazione T2 è iniziata (*begin transaction*)
   - ☑ Vero
   - ☐ Falso
   - ☐ Non è possibile determinarlo
4. la transazione T3 è iniziata (*begin transaction*) dopo il *checkpoint*
   - ☐ Vero
   - ☑ Falso
   - ☐ Non è possibile determinarlo
5. la transazione T5 ha fatto *commit* dopo il *checkpoint*
   - ☑ Vero
   - ☐ Falso
   - ☐ Non è possibile determinarlo
6. la transazione T5 è iniziata (*begin transaction*) e ha fatto *commit* dopo che la transazione T3 è inziata (*begin transaction*)
   - ☐ Vero
   - ☐ Falso
   - ☑ Non è possibile determinarlo
7. il log contiene un record di *abort* per la transazione T6
   - ☐ Vero
   - ☐ Falso
   - ☑ Non è possibile determinarlo
8. l'operazione di *delete* di O7 da parte di T7 segue nel log l'operazione di *insert* di O5 da parte di T5
   - ☐ Vero
   - ☐ Falso
   - ☑ Non è possibile determinarlo
9. l'operazione di *delete* di O6 da parte di T6 segue nel log l'operazione di *update* di O3 da parte di T3
   - ☑ Vero
   - ☐ Falso
   - ☐ Non è possibile determinarlo
10. l'operazione di *insert* di O8 da parte di T5 precede nel log l'operazione di *update* di O2 da parte di T2
    - ☐ Vero
    - ☑ Falso
    - ☐ Non è possibile determinarlo

Esercizio 2)

Dati i seguenti schedule:

1. $r_1(z)\ r_3(x)\ r_2(z)\ w_1(z)\ w_2(y)\ w_4(y)\ w_1(y)\ w_2(t)\ r_4(t)\ w_3(y)$
2. $r_3(z)\ r_1(y)\ w_3(z)\ w_3(y)\ r_1(x)\ r_2(y)\ w_2(y)\ w_1(x)\ r_2(x)\ w_3(x)$

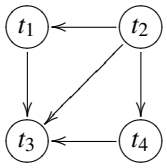
Si dica se gli schedule sono *VSR* e/o *CSR*, indicando (qualora esistano) *tutti* gli schedule seriali equivalenti. Si svolga l'esercizio illustrando dettagliatamente il processo/ragionamento seguito.

Schedule 1

1. Relazioni legge-da

LETTURA	LEGGE-DA	VINCOLI	ALTRI VINCOLI
$r_1(z)$	–		
$r_3(x)$	–		
$r_2(z)$	–		$2 \rightarrow 1$ (altrimenti introdurrebbe una leggi-da)
$r_4(t)$	$w_2(t)$	$2 \rightarrow 4$	

RISORSA	SCRITTURA FINALE	ALTRE SCRITTURE	VINCOLI
x	–		
y	$w_3(y)$	$w_1(y)$, $w_2(y)$, $w_4(y)$	$1 \rightarrow 3$ $2 \rightarrow 3$ $4 \rightarrow 3$
z	$w_1(z)$		
t	$w_2(t)$		



Non è presente alcun ciclo e quindi lo schedule potrebbe essere *VSR*.
Esistono due possibili schedule seriali view-equivalente a quello dato:

- t_2, t_1, t_4, t_3
- t_2, t_4, t_1, t_3

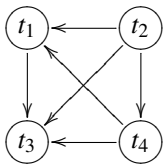
Tutti e due gli schedule equivalenti presentano le stesse relazioni *leggi-da* e le stesse *scritture finali*, dello schedule analizzato, che risulta quindi *VSR*.

Essendo *VSR* potrebbe essere anche *CSR*.

2. Valutiamo ora i conflitti presenti nello schedule:

- $r_2(z), w_1(z)$
- $w_2(y), w_4(y)$
- $w_2(y), w_1(y)$
- $w_2(y), w_3(y)$
- $w_4(y), w_1(y)$
- $w_4(y), w_3(y)$
- $w_1(y), w_3(y)$
- $w_2(t), r_4(t)$

Il grafo dei conflitti riportato di seguito è aciclico.



Si conclude che lo schedule è *CSR*. Esiste un solo schedule seriale conflict-equivalente a quello dato:

- t_2, t_4, t_1, t_3

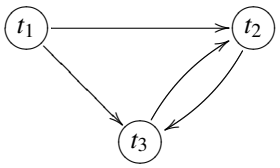
Schedule 2

1. Relazioni legge-da

LETTURA	LEGGE-DA	VINCOLI	ALTRI VINCOLI
$r_3(z)$	–		
$r_1(y)$	–		$1 \to 3$ (altrimenti introdurrebbe una leggi-da) $1 \to 2$ (altrimenti introdurrebbe una leggi-da)
$r_1(x)$	–		$1 \to 3$ (altrimenti introdurrebbe una leggi-da)
$r_2(y)$	$w_3(y)$	$3 \to 2$	
$r_2(x)$	$w_1(x)$	$1 \to 2$	$1 \to 3$ oppure $2 \to 3$ (siccome la prima non può essere per il vincolo delle scritture finali, allora deve essere la seconda)

Scritture finali

RISORSA	SCRITTURA FINALE	ALTRE SCRITTURE	VINCOLI
x	$w_3(x)$	$w_1(x)$	$1 \to 3$
y	$w_2(y)$	$w_3(y)$	$3 \to 2$
z	$w_3(z)$		

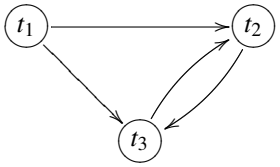


Nel grafo è presente un ciclo (tra t_2 e t_3) e quindi lo schedule non è *VSR*.
Non essendo *VSR* non è neppure *CSR*.

2. Valutiamo ora i conflitti presenti nello schedule per verificare che non sia effettivamente *CSR*:

   - $r_1(y), w_3(y)$
   - $r_1(y), w_2(y)$
   - $w_3(y), r_2(y)$
   - $w_3(y), w_2(y)$
   - $r_1(x), w_3(x)$
   - $w_1(x), r_2(x)$
   - $w_1(x), w_3(x)$
   - $r_2(x), w_3(x)$

   Il grafo dei conflitti riportato di seguito è ciclico.



   Si conclude che lo schedule non è *CSR*.

Esercizio 3)

Si consideri lo schedule:

$$r_1(y)\ r_1(z)\ r_2(x)\ r_3(y)\ w_1(x)\ w_2(t)\ r_4(t)\ w_3(x)\ w_4(y)$$

Dire se può essere stato generato da uno scheduler basato su *2PL* base, motivando opportunamente la risposta. Si consideri, per la soluzione, un lock a due stati.

- Transazioni da considerare:
  $t_1 = r_1(y)\ r_1(z)\ w_1(x)$
  $t_2 = r_2(x)\ w_2(t)$
  $t_3 = r_3(y)\ w_3(x)$
  $t_4 = r_4(t)\ w_4(y)$
- Valutiamo ora le singole operazioni:

OPERAZIONE	LOCK	x	y	z	t
init.					
$r_1(y)$	lock(t_1,y): OK		t_1		
$r_1(z)$	lock(t_1,z): OK		t_1	t_1	
$r_2(x)$	lock(t_2,x): OK	t_2	t_1	t_1	
$r_3(y)$	lock(t_2,t): OK	t_2	t_1	t_1	t_2
	unlock(t_2,x): OK		t_1	t_1	t_2
	lock(t_1,x): OK	t_1	t_1	t_1	t_2
	unlock(t_1,y): OK	t_1		t_1	t_2
	lock(t_3,y): OK	t_1	t_3	t_1	t_2
$w_1(x)$	OK	t_1	t_3	t_1	t_2
$w_2(t)$	OK	t_1	t_3	t_1	t_2
$r_4(t)$	unlock(t_2,t): OK	t_1	t_3	t_1	
	lock(t_4,t): OK	t_1	t_3	t_1	t_4
$w_3(x)$	unlock(t_1,x): OK		t_3	t_1	t_4
	lock(t_3,x): OK	t_3	t_3	t_1	t_4
$w_4(y)$	unlock(t_3,y): OK	t_3		t_1	t_4
	lock(t_4,y): OK	t_3	t_4	t_1	t_4

- Lo schedule può essere stato prodotto da uno scheduler 2PL.

Esercizio 4)

Si considerino i seguenti schemi relazionali:

AGENTE(Codice, Nome, Cognome)
PROVVIGIONE(CodiceAgente, Data, Ammontare)
STORICO(CodiceAgente, AmmontareTotale)

Scrivere un sistema di trigger che quando un AGENTE viene cancellato dal database:

- salvi nella tabella STORICO il totale di tutte le provvigioni da lui percepite;
- cancelli dalla tabella PROVVIGIONE tutti i record relativi all'agente cancellato.

```
CREATE TRIGGER AggiornaStorico
BEFORE DELETE ON Agente
FOR EACH ROW
BEGIN
     INSERT INTO Storico
     SELECT CodiceAgente, SUM(Ammontare)
     FROM   Provvigione
     WHERE  CodiceAgente = OLD.Codice
END;
```

```
CREATE TRIGGER AggiornaStorico
AFTER DELETE ON Agente
FOR EACH ROW
BEGIN
     DELETE FROM Provvigione
     WHERE  CodiceAgente = OLD.Codice
END;
```