

1° F.N → ATTRIBUTE SINGOLI  
INDIRIZZO { VIA  
CIVICO

2° F.N

DIP. FUNZIONALIS

A → B      B → C  
C → D

STUDENTE

↓ E ANCHE A

PIÙ CORSI

→ [ STUDENTE / CORSO / PROF. ] 1° TABELLA A RANCO  
PER STUDENTI  
RIBONDEGGIO?

→ [ PROF. / CORSO / TABELLA A ]

PROF / CORSO

STUDENTE / CORSO / ID  $\leftarrow$  D.F.

CORSO MANOVRA

3<sup>o</sup> F.N / BCNF



SUPERCHIAVE

Una relazione  $R$  con chiavi  $K_1, \dots, K_n$  è in Terza Forma Normale se:

Per ogni dipendenza funzionale non banale  $X \rightarrow Y$ , almeno una delle seguenti condizioni sono valide:

- $X$  è superchiave (BCNF)
- ogni attributo in  $Y$  è contenuto in almeno una tra le chiavi  $K_1, \dots, K_n$ .

TABELLA 1 ~~X~~ TABELLA 2

Sia data la relazione  $R(A, B, C, D, E, F)$  con copertura ridotta  $G = \{C \rightarrow B, C \rightarrow D, C \rightarrow F, BE \rightarrow C, B \rightarrow A, AD \rightarrow E\}$

- Trovare la/e chiave/i di  $R$ , motivando la risposta.
- Quali dipendenze violano la 3NF? Motivare la risposta
- Effettuare una decomposizione in 3NF.
- La decomposizione è anche in BCNF? Motivare la risposta

$C^+ = \{C, A, B, D, E, F\}$  /

$AD^+ = \{A, D, E\}$

$B^+ = \{B, A\}$

$BE^+ = \{B, E, C, D, A, F\}$



SUBSET PIÙ GRANDI  $\Rightarrow$  CHIARI

C / BG

2°  $\rightarrow$  3 F. N

Una relazione R con chiavi  $K_1, \dots, K_n$  è in Terza Forma Normale se:

Per ogni dipendenza funzionale non banale  $X \rightarrow Y$ , almeno una delle seguenti condizioni sono valide:

- X è superchiave (BCNF)
- ogni attributo in Y è contenuto in almeno una tra le chiavi  $K_1, \dots, K_n$ .

SK  
|  
 $\emptyset \rightarrow \bar{A}$      $\underline{B} \rightarrow A$   
|  
NO SK  
 $AD \rightarrow C$

$\emptyset \rightarrow B$      $\emptyset \rightarrow D$      $\emptyset \rightarrow C$   
|            |            |  
SK        SK        SK

① Decomposiz.  $\rightarrow$  ?

G  $\begin{cases} \text{SUBSET 1} \\ \text{SUBSET 2} \end{cases}$  s.t.  $X \rightarrow A$   
 $Y \rightarrow B$   
 $X^+ = Y^+$

$\{ C \rightarrow B, C \rightarrow D, C \rightarrow F, BD \rightarrow C \}$

$\rightarrow \{ B \rightarrow A \}$   $S_1, S_2, S_3$

$\rightarrow \{ AD \rightarrow C \}$

② sottoinsiemi

$$R_1 \subseteq \{ \underline{B}, \underline{D}, \underline{F}, \underline{G} \}$$

$$R_2 \subseteq \{ \underline{B}, \underline{A} \}$$

$$R_3 \subseteq \{ \underline{A}, \underline{D}, \underline{G} \}$$

③  $A \subseteq B \subseteq C$

$$x \subseteq y$$

$$(x \subseteq y)$$

non A C C A D S

④  $\exists K$  (chiusa)

per cui no relazione

aggiungila  $\rightarrow \exists^o \text{ F.N}$

B CNF

$\rightarrow \forall D, F, \text{ STA IN DSD}$

$\rightarrow \forall D, P. X \rightarrow A,$   
 $x \in \text{superchiusa}$

- $R_1(C,B,D,F,E)$ :  $C \rightarrow B$ ,  $C \rightarrow D$ ,  $C \rightarrow F$  soddisfano BCNF poiché  $C$  è chiave;  $BE \rightarrow C$  soddisfa BCNF poiché  $BE$  è chiave
- $R_2(B,A)$ :  $B \rightarrow A$  soddisfa BCNF poiché  $B$  è chiave
- $R_3(A,D,E)$ :  $AD \rightarrow E$  soddisfa BCNF poiché  $AD$  è chiave

Tutte le relazioni rispettano la BCNF, quindi la decomposizione è anche in BCNF.

Data la relazione  $R(A,B,C,D)$  con dipendenze funzionali  
 $\{ C \rightarrow D, C \rightarrow A, B \rightarrow C \}$ .

$B$  è chiave.  $\leftarrow \textcircled{B} = \{B, C, A, D\}, C^+ = \{C, A, D\}$

2. Dire quale dipendenze violano la forma normale di Boyce Codd (BCNF), spiegandone la ragione.

3. Decomporre in BCNF

$\textcircled{C} \Rightarrow \{C \rightarrow D, C \rightarrow A\}$   
 NO SUPERFLUOUS

BCNF

$R_1 = \{B, C, A, D\}$        $R_2 \subseteq R_1$   
 $R_2 = \{C, A, D\}$

$\rightarrow R_1 = \{B, C, A, D\}$

$$C \rightarrow A, \quad \cancel{C \rightarrow D}$$

$$R_1 = \{ \cancel{A}, B, \underline{C} \}$$

(V)

$$R_2 = \{ \underline{C}, D \}$$

~~C → A~~

$$R_1 = \{ \underline{B}, \underline{C} \}$$

$$R_2 = \{ \underline{C}, A \}$$

$$R_3 = \{ \underline{C}, D \}$$

(V)

> DEC. SONDA  
POSITA

$$X \rightarrow A, \quad T \rightarrow B, \quad X^+ = Y^+$$

≡

$$R_1 = \{ \underline{B}, \underline{C} \} \text{ e}$$

$$R_2 = \{ \underline{C}, A, D \}$$

Considerare uno schema di relazione R (E, N, L, C, S, D, M, P, A) con le seguenti dipendenze funzionali:

$E \rightarrow NS$ ,  
 $NL \rightarrow EMD$ ,  
 $EN \rightarrow LCD$ ,  
 $C \rightarrow S$ ,  
 $D \rightarrow M$ ,  
 $M \rightarrow D$ ,  
 $EPD \rightarrow A$ ,  
 $NLCP \rightarrow A$ .

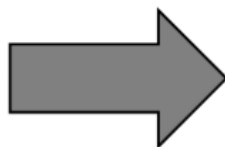
Calcolare una **copertura ridotta** per tale insieme e decomporre la relazione in **terza forma normale**.

I passi per calcolare la copertura ridotta di una relazione sono i seguenti:

1. Sostituzione dell'insieme dato con quello equivalente che ha tutti i secondi membri costituiti da singoli attributi;
2. Per ogni dipendenza verifica dell'esistenza di attributi eliminabili dal primo membro;
3. Eliminazione delle dipendenze ridondanti.

1. Sostituzione dell'insieme dato con quello equivalente che ha tutti i secondi membri costituiti da singoli attributi

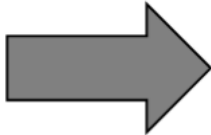
$E \rightarrow NS$   
 $NL \rightarrow EMD$   
 $EN \rightarrow LCD$   
 $C \rightarrow S$   
 $D \rightarrow M$   
 $M \rightarrow D$   
 $EPD \rightarrow A$   
 $NLCP \rightarrow A$



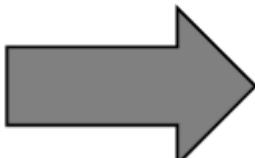
$E \rightarrow S$   
 $E \rightarrow N$   
 $NL \rightarrow E$   
 $NL \rightarrow M$   
 $NL \rightarrow D$   
 $EN \rightarrow L$   
 $EN \rightarrow C$   
 $EN \rightarrow D$   
 $C \rightarrow S$   
 $D \rightarrow M$   
 $M \rightarrow D$   
 $EPD \rightarrow A$   
 $NLCP \rightarrow A$

$AB \rightarrow C$   
 $A \rightarrow B$   
 2° n. singolo

2. Per ogni dipendenza verifica dell'esistenza di attributi eliminabili dal primo membro

$E \rightarrow S$		$E \rightarrow S$	
$E \rightarrow N$		$E \rightarrow N$	
$NL \rightarrow E$		$NL \rightarrow E$	
$NL \rightarrow M$		$NL \rightarrow M$	
$NL \rightarrow D$		$NL \rightarrow D$	
$EN \rightarrow L$		$E \rightarrow L$	$(EN \rightarrow L, E \rightarrow N)$
$EN \rightarrow C$		$E \rightarrow C$	$(EN \rightarrow C, E \rightarrow N)$
$EN \rightarrow D$		$E \rightarrow D$	$(EN \rightarrow D, E \rightarrow N)$
$C \rightarrow S$		$C \rightarrow S$	
$D \rightarrow M$		$D \rightarrow M$	
$M \rightarrow D$		$M \rightarrow D$	
$EPD \rightarrow A$		$EP \rightarrow A$	$(EPD \rightarrow A, E \rightarrow D)$
$NLCP \rightarrow A$		$NLP \rightarrow A$	$(NLCP \rightarrow A, NL \rightarrow E, E \rightarrow C)$

3. Eliminazione delle dipendenze ridondanti

<del><math>E \rightarrow S</math></del>		
$E \rightarrow N$		$E \rightarrow N$
$NL \rightarrow E$		$NL \rightarrow E$
<del><math>NL \rightarrow M</math></del>		
<del><math>NL \rightarrow D</math></del>		
$E \rightarrow L$		$E \rightarrow L$
$E \rightarrow C$		$E \rightarrow C$
$E \rightarrow D$		$E \rightarrow D$
$C \rightarrow S$		$C \rightarrow S$
$D \rightarrow M$		$D \rightarrow M$
$M \rightarrow D$		$M \rightarrow D$
<del><math>EP \rightarrow A</math></del>		
$NLP \rightarrow A$		$NLP \rightarrow A$



# TRANSAZIO N1

CK = T5, T6  
↓

Sia data la seguente porzione di log fino al guasto: CK(T5, T6), B(T7), U(T7, O6, B6, A6), U(T6, O3, B7, A7), B(T8), I(T8, O5, A5), C(T8), A(T5). Sapendo che occorre effettuare l'UNDO di T5, T6 e T7 e il REDO di T8, quale è la prima operazione da effettuare per la ripresa a caldo?

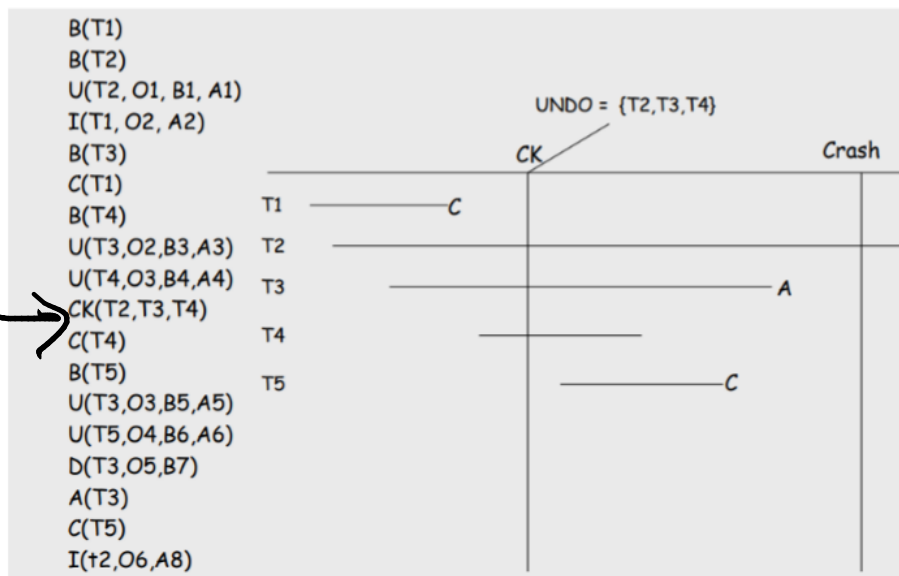
1. D(O5)
2. O3=B7
3. O6=B6
4. O5=A5

REDO = d T8 y, UNDO = d T5, T6 y

La ripresa a caldo quindi:

- trova l'ultimo checkpoint ripercorrendo il log a ritroso
- costruisce gli insiemi UNDO/REDO
- ripercorre il log all'indietro fino alla più vecchia fra le transazioni UNDO/REDO, disfacend le azioni delle transazioni in UNDO
- ripercorre il log in avanti, rifacendo tutte le azioni delle transazioni in REDO

Un esempio pratico:



Glossario  
iniziali:

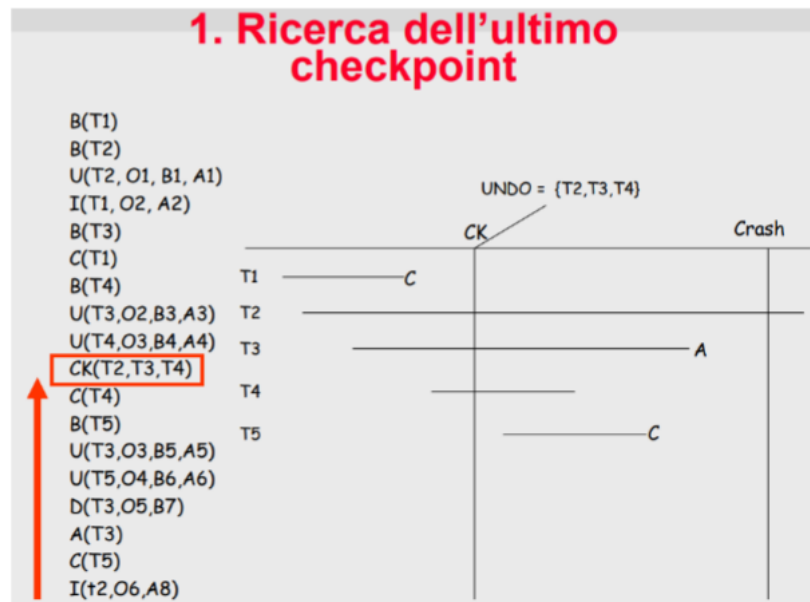
I = Insert  
D = Delete  
B = Begin  
C = Commit  
U = Update  
A = Abort  
CK = Check

HOT RELOAD → RECOVERY  
LAST

CRASH CHECKPOINT

REDO / UNDO

1) Si cerca l'ultimo checkpoint, partendo da sotto e arrivando a:



2) Si costruiscono gli insiemi UNDO e REDO (nota: questi due insiemi si costruiscono partendo dal basso e arrivando al primo CK/CHECK. Le operazioni vanno comunque rifatte per UNDO e REDO su tutto il file, ma si considerano solo le operazioni che coinvolgono questi due insiemi. Quindi, se devo costruire UNDO/REDO non mi serve scorrere tutto il file di log, ma solo arrivare all'ultimo checkpoint, partendo appunto dal basso)

All'interno di UNDO andremo ad inserire:

*Handwritten:* T2 T3 T4 | Commit → UNDO

*Handwritten:* D/A/I/U → UNDO

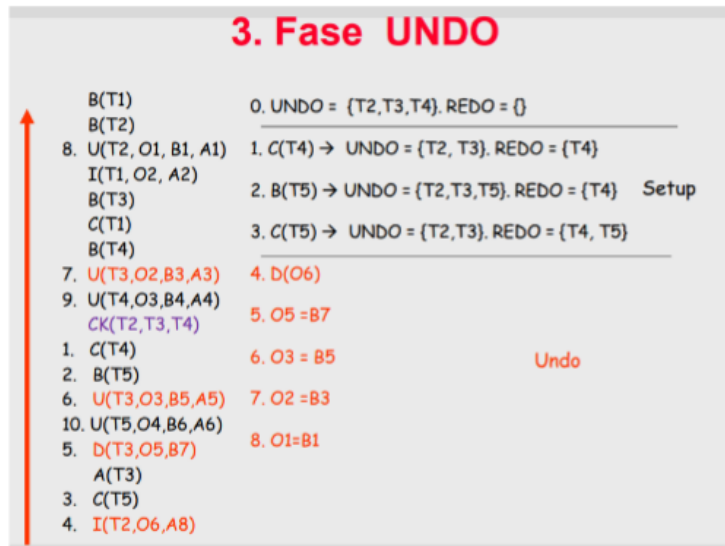
- inizialmente T2 e T3, dato che su di esse pone controllo il check
- si ha il commit di T4 in C(T4) e la transazione viene posta in REDO
- qualsiasi altra operazione (D/A/I/U) prevede che la transazione rimanga in UNDO; è il caso di T5 che inizia (B/BEGIN), di T3 che aggiorna l'oggetto O3 in B5, di T5 che aggiorna l'oggetto O4 in B6, la cancellazione dell'oggetto O5 in B7 e l'abort di A3 (che fanno in modo T2,T3,T5 rimangano in UNDO)
- si ha il commit di T5 e alla fine si rimane con:  
C(T5) → UNDO = {T2,T3}. REDO = {T4, T5}

## 2. Costruzione degli insiemi UNDO e REDO

- |                    |  |
|--------------------|--|
| B(T1)              | 0. UNDO = {T2,T3,T4}. REDO = {}                    |
| B(T2)              |  |
| 8. U(T2,O1,B1,A1)  | 1. C(T4) → UNDO = {T2, T3}. REDO = {T4}            |
| I(T1,O2,A2)        |  |
| B(T3)              | 2. B(T5) → UNDO = {T2,T3,T5}. REDO = {T4}    Setup |
| C(T1)              |  |
| B(T4)              | 3. C(T5) → UNDO = {T2,T3}. REDO = {T4, T5}         |
| 7. U(T3,O2,B3,A3)  |  |
| 9. U(T4,O3,B4,A4)  |  |
| CK(T2,T3,T4)       |  |
| 1. C(T4)           |  |
| 2. B(T5)           |  |
| 6. U(T3,O3,B5,A5)  |  |
| 10. U(T5,O4,B6,A6) |  |
| 5. D(T3,O5,B7)     |  |
| A(T3)              |  |
| 3. C(T5)           |  |
| 4. I(T2,O6,A8)     |  |

3) Dato l'insieme degli UNDO, si ripercorre dalla fine all'inizio tutto il log, rifacendo le operazioni che riguardano T2 e T3. Quindi:

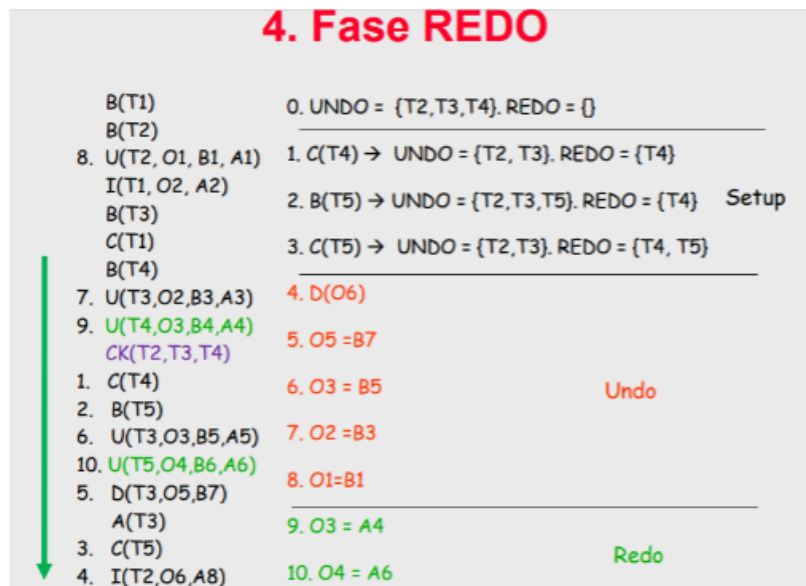
- si ha un INSERT di T2 su O6; questo dovrà essere cancellato (quindi quando si ha un INSERT si ha la cancellazione, per permettere di rifarla → D(O6)
- si ha il DELETE di O5 in B7 e dunque → O5 = B7 (che era lo stato precedente, ma è la convenzione che si adotta)
- si ha un UPDATE DI T3 su O3 e → O3=B5
- prima del CHECK si ha un UPDATE di T3 su O2 e → O2=B3
- finalmente, si ha un UPDATE di T2 su O1 in B1 → O1=B1



Sull'insieme si ha da considerare T4 e T5

Si nota che le operazioni che li riguardano sono:

- U(T4, O3, B4, A4) e quindi porta a dire O3 = B4
- U(T5, O4, B6, A6) e quindi porta a dire O4 = A6



Sempre seguendo l'ordine, tiene traccia di quali operazioni rifare.

Se il DB è scritto in Modalità Immediata (immediatamente dopo il log), il REDO non è necessario.

Invece se il DB è scritto in Modalità Differita (solo dopo un commit), l'UNDO non è necessario.

È possibile venga fatto l'UNDO oppure il REDO di operazioni con effetto nullo nel database, come se fosse fatto una sola volta (ad esempio:  $\text{undo}(\text{undo}(A)) = \text{undo}(A)$      $\text{redo}(\text{redo}(A)) = \text{redo}(A)$ )

In merito ai guasti abbiamo:

- *guasti di sistema (soft)*, intendendo errori di programma, crash di sistema, caduta di tensione; si opera la ripresa a caldo (*warm restart*), ripetendo le transazioni
- *guasti di dispositivo (hard)*, intendendo guasti su dispositivi di memoria secondaria; tuttavia si ha ridondanza su varie memorie stabili del log, non perdendo la memoria stabile

La struttura dei guasti, oltre al checkpoint, periodicamente esegue dei *dump* (copie del database, ad esempio su nastro) di tutto il contenuto del DB. Grazie a questo evita di dover ogni volta ricominciare.

TRANSITION  $\rightarrow$  ACD

S<sub>1</sub>: R<sub>1</sub>(x) R<sub>2</sub>(z) W<sub>1</sub>(x)  
SCHEDULES W<sub>2</sub>(z)

R/W = READ / WRITE

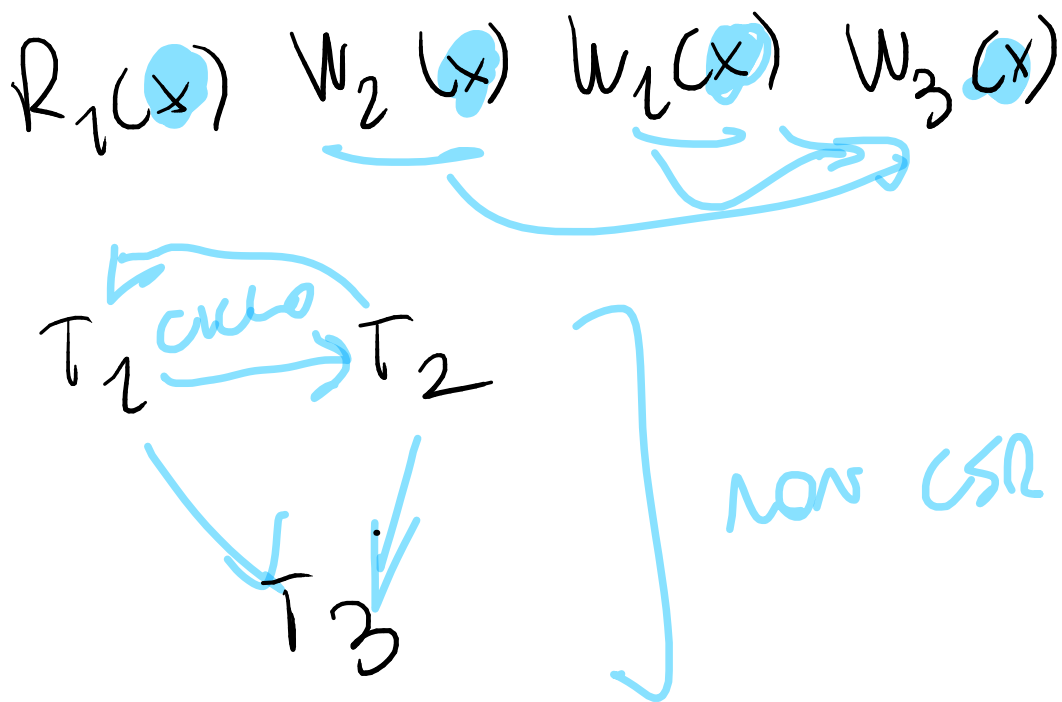
VSQ  $\rightarrow$  R<sub>1</sub>(x) W<sub>1</sub>(x) R<sub>2</sub>(x)  
                    R            R  
                    W<sub>2</sub>(x)

- STRESS WRITES FIRST

- WGGI  $\rightarrow$  A

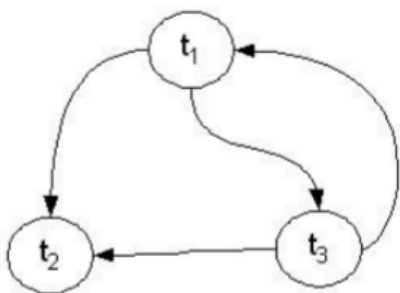
W<sub>1</sub>(x) R<sub>1</sub>(x) R<sub>2</sub>(x)  
          R          R

CSQ  $\Rightarrow$  TRANS. DIVERS  
SCHEDULES  $\rightarrow$  WR  
STRESS  $\rightarrow$  A



Dato lo schedule  $S = r_1(x) \ r_1(t) \ r_2(z) \ w_3(x) \ w_1(x) \ r_1(y) \ w_3(t) \ w_2(x) \ w_1(y)$ , indicare se  $S$  è view e/o conflict-serIALIZZABILE, motivando la risposta. Se  $S$  è view e/o conflict-serIALIZZABILE, indicare uno schedule seriale che è view e/o conflict-equivalente, motivando la risposta.

Il grafo dei conflitti è come segue:



Siccome il grafo contiene un ciclo, lo schedule non è conflict-serIALIZZABILE.

Tuttavia è view-serIALIZZABILE, giacché view-equivalente al seguente schedule seriale:

$S_2 = r_1(x) \ r_1(t) \ w_1(x) \ r_1(y) \ w_1(y) \ w_3(x) \ w_3(t) \ r_2(z) \ w_2(x)$

Infatti, hanno le stesse relazioni leggi-da e le stesse scritture finali. In particolare, la relazione leggi-data è vuota, mentre le scritture finali sono le seguenti sia per  $S$  che per  $S_2$ :  $w_2(x)$ ,  $w_1(y)$  e  $w_3(t)$