DEFINIZIONI

1 DEF CHIAVE / VINCOLO DI CHIAVE

DATO UNO SCHEMA DI RELAZIONE

$$R(A_1, A_2, \dots, A_n)$$

DEFINIAMO UNA **CHIAVE** UN SOTTOINSIEME K NON VUOTO MINIMALE (RISPETTO ALLA PROPRIETÀ IDENTIFICANTE) DI $A_1, \dots, A_n$

OVVERO NON ESISTE $K' \subset K$ IN GRADO DI IDENTIFICARE UN'ISTANZA DELLA RELAZIONE

UN **VINCOLO DI CHIAVE** É UN ESPRESSIONE DEL TIPO $K: \{B_1, \dots, B_x\} \subseteq \{A_1, \dots A_n\}$

TALE CHE

1. $\forall$ ISTANZA r DI R VIENE SODDISFATTO TALE VINCOLO

$$\forall t_1 \in r, \nexists t_2 \in r : t_1[B_1] = t_2[B_1] \wedge \dots \wedge t_1[B_x] = t_2[B_x]$$

É DEFINITO SULLO SCHEMA MA DEVE ESSERE SODDISFATTO DALL'ISTANZA (IN OGNI SCHEMA R SONO AMMESSE PIÙ CHIAVI MA SOLO UNA É CHIAVE PRIMARIA. LE ALTRE SONO INDICATE CON LA PAROLA UNIQUE)

VINCOLO CHIAVE ESTERNA

DATO UN DATABASE D, CONTENENTI DUE SCHEMI DI RELAZIONE

$R1(A_1, \dots, A_n)$ $R_2(B_1, \dots, B_m)$

UNA **CHIAVE ESTERNA** (PER LO SCHEMA DI RELAZIONE $R1$) É UN INSIEME DI ATTRIBUTI CHE FA RIFERIMENTO ALLA CHIAVE PRIMARIA DI UNO SCHEMA DI RELAZIONE R2.

UN **VINCOLO DI CHIAVE** ESTERNA É UN ESPRESSIONE DEL TIPO

$$R1[C_1, \dots, C_K] \subseteq_{FK} R2[D1, \dots, D_K] \quad \text{CON} \quad K_2 = \{D1, \dots, D_K\}$$

CHIAVE DI R2

DOVE $\{C_1, \dots, C_K\} \subseteq \{A_1, \dots, A_n\}$ E $\{D_1, \dots, D_K\} \subseteq \{B_1, \dots, B_m\}$

UN ISTANZA d DI D, CON $r1, r2$ ISTANZE DI R_1, R_2, SODDISFA TALE VINCOLO SE

$$\forall t_1 \in r1, \exists t_2 \in r2 : t_1[C1] = t_2[D1] \wedge \dots \wedge t_1[C_K] = t_2[D_K]$$

DIPENDENZA FUNZIONALE / VINCOLO

DATO UNO SCHEMA DI RELAZIONE

$$R_1(A_1, A_2, \dots, A_n)$$

UNA DIPENDENZA FUNZIONALE, SU TALE SCHEMA, É UN ESPRESSIONE DEL TIPO
$X \to Y$ OVE $X, Y \subseteq \{A_1, A_2 \dots, A_n\}$

DEFINITA UN ISTANZA r DI R, É CONSISTENTE RISPETTO A $X \to Y$, SE DEFINITO

$$X = \{x_1, \dots, x_m\} \quad e \quad Y = \{y_1, \dots, y_K\}$$

VALE CHE

$$\forall t_1, t_2 \in r_1 : t_1[x_1] = t_2[x_1] \wedge \dots \wedge t_1[x_m] = t_2[x_m]$$

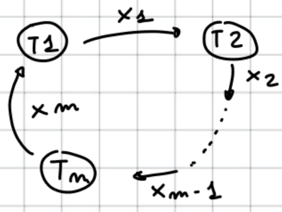
$$\underset{\text{(ALLORA)}}{\Rightarrow} t_1[y_1] = t_2[y_1] \wedge \dots \wedge t_1[y_{K2}] = t_2[y_{K2}]$$

PER OGNI COPPIA DI (TUPLE) $t_1, t_2 \in r$, AVENTI GLI STESSI VALORI SUGLI ATTRIBUTI DI X, ALLORA HANNO GLI STESSI VALORI ANCHE SUGLI ATTRIBUTI DI Y.

Dimostrazione 2PL => serializzabilità

Dato un insieme di n transazioni $T_1, T_2, \dots T_n$ che aderiscono al protocollo 2PL e definito uno schedule S elaborato secondo la corretta semantica dei lucchetti si supponga per assurdo che S non sia conflict-serializable $\rightarrow$ allora si avrà un grafo delle precedenze ciclico (contiene almeno un ciclo)

Vi saranno dunque m transazioni che formano un ciclo ove $1 < m \leq n$
Permutiamo l'insieme delle n transazioni affinché le prime m transazioni siano quelle coinvolte nel ciclo $(T_1, \dots, T_m)$ (senza perdità di generalità)



Si osserva quindi che

$$lock^{T_1}(x_1) < lock^{T_2}(x_1)$$
$$unlock^{T_1}(x_1) < lock^{T_2}(x_1)$$
$$\vdots$$
$$unlock^{T_{m-1}}(x_{m-1}) < lock^{T_m}(x_{m-1})$$
$$unlock^{T_m}(x_m) < lock^{T_1}(x_m)$$

Dalle relazioni ottenute si ottiene $unlock^{T_1}(x_1) < lock^{T_1}(x_m)$
cioè che $unlock^{T_m}(x_m) < lock^{T_1}(x_m)$

Ma è un assurdo poiché, per 2PL, lo schedule aderisce al 2PL protocol

=> il grafo delle precedenze non è ciclico è dunque lo schedule è conflict-serializable