Sheet 5 [Scheduling]

u can SKIP 17.4 17.5

17.4

* check final out for T_1 , T_2 for $N=2$, $M=2$

T_1:
r(x)
x = x-N
w(x)
r(Y)
Y = Y+N
w(Y)

T_2:
r(x)
x = x+M
w(x)

for a serializable schedule → x) final = (X-4) , Y) final = (Y+2)

* if we used T_2' instead of T_2:

T_2':
r(x)
x = x+M
if x > 90 ; exit
else ; w(x)

for a serializable schedule → same outcome if x) initial ≤ 88 (implied consistency rule)
↓
same value of X as not updated if x > 90.

17.5] try using T_1' instead of T_1:

T_1':
r(x)
x = x-N
w(x)
r(Y)
Y = Y+N
if Y > 90 ; exit
w(Y)

for a serializable schedule → same outcome if x) initial ≤ 88, Y) initial ≤ 88
↳ obeys an implied consistency rule that: x < 90 & Y < 90.

17.6 *After Adding the Commit operation

T_1: r(x), X=X-N, w(x), r(Y), Y=Y+N, w(Y), Commit T_1

T_2: r(x), X=X+M, w(X), Commit T_2

* Transactions
  - T_1 : $r_1(X), w_1(X), r_1(Y), w_1(Y), c_1$;
  - T_2 : $r_2(X), w_2(X), c_2$;

} Short hand Notation

* No of Possible Schedules $= \frac{\lfloor \sum_{i}^{M} n_i \rfloor}{\prod_{i}^{M} \lfloor n_i \rfloor}$

M → no transac-
n_i → no operations in transaction (i) if all Ops in one transc

e.g. $m=2, n_1=5, n_2=3$ → no sch $= \frac{\lfloor 5+3 \rfloor}{\lfloor 5 \rfloor \lfloor 3 \rfloor} = 56$

Possible comb^n of each sch

(Not guaranteed to be Serializable or valid. Just, Combinations)

* Possible Schedules:
  - S_1 : $r_1(x), w_1(x), r_1(Y), w_1(Y), c_1, r_2(x), w_2(x), c_2$
  - S_2 : _____ , $r_2(x), c_1, w_2(x), c_2$
  - S_3 : _____ , $r_2(x), w_1(Y), c_1, r_2(x), w_2(x), c_2$
  - ⋮

* Note that → no of serial Schedules $= \lfloor M \rfloor$ ← no transc

17.7 for → $T_1: r_1(X), w_1(X), r_1(Y), w_1(Y)$ } n = 4] m = 2
→ $T_2: r_2(X), w_2(X)$ } n = 2

$$\text{No. sch} = \frac{(2+4)!}{2!\,4!} = \frac{6 \times 5}{2} = 15$$

$S_1: r_1(X), w_1(X), r_1(Y), w_1(Y), r_2(X), w_2(X)$
$S_{15}: r_2(X), w_2(X), r_1(X), w_1(X), r_1(Y), w_1(Y)$
} see more schedules in the model answer!

$S_2: r_1(X), w_1(X), r_1(Y), r_2(X), w_1(Y), w_2(X)$ → Conflict Serializ

⋮

$S_6: r_1(X), w_1(X), r_2(X), w_2(X), r_1(Y), w_1(Y)$ → Conflict Serializ

* هتلاقيهم كلهم مفيهمش مشاكل وكاملين زي $T_1 \to T_2$ أو $T_2 \to T_1$
ولو رسمت الجراف مش هتلاقي cycles

$S_7: r_1(X), r_2(X), w_1(X), r_1(Y), w_1(Y), w_2(X)$ → Not Conflict Serializ
(WAR, WAW)

$S_{14}: r_2(X), r_1(X), w_2(X), w_1(X), r_1(Y), w_1(Y)$ → Not Conflict Ser
(WAR, WAW)

* هتلاقيهم كلهم فيهم cycle في الجراف، طبعاً مش هترسم الجراف، بس هتلاقي بالعين في Conflict من T_1 لـ T_2
و Conflict من T_2 لـ T_1

* خد بالك عاوزين نطلع [WAR, RAW, WAW]، أسهل حاجة
دوري على w بعديها ابحث لورا لحد ما تلاقي Conflict،
لو ملقتش، امسك r بعديها ارجع.
لو فضل r في الآخر هتبقى RAR ودي مش Conflict!

17.18 For 3 transactions, we have $3! = 6$ serial schedules

The Serial Schedules:
- $T_1\ T_2\ T_3$
- $T_1\ T_3\ T_2$
- $T_2\ T_1\ T_3$
- $T_3\ T_2\ T_1$
- $T_3\ T_1\ T_2$
- $T_2\ T_3\ T_1$

* كل T_i لها ترتيب في كل Position !

$S_1(T_1\ T_2\ T_3) = r_1(X), w_1(X), r_1(Y), w_1(Y), r_2(Z), r_2(Y), w_2(Y)$
$r_2(X), w_2(X), r_3(Y), r_3(Z), w_3(Y), w_3(Z)$

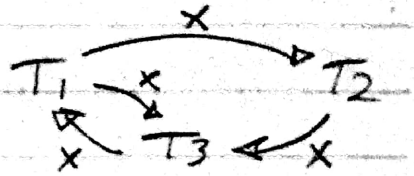
17.20

* there's no need for an explicit "Begin" statement, since transaction initiation is done implicitly when SQL statement are encountered
* on the other hand, the transaction could end in two ways either successfully using "commit" or unsuccessfully using "Abort" or "Rollback".
* An explicit end statement is essential to identify the right way of ending a transaction!

17.22 We have 3-transc; T_1, T_2, T_3

a) $S_1: r_1(x), r_3(x), w_1(x), r_2(x), w_3(x)$

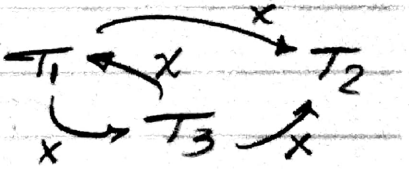
RAW, WAW, WAR, WAR



∴ Cycle → Not Serializable

b) $S_2: r_1(x), r_3(x), w_3(x), w_1(x), r_2(x)$

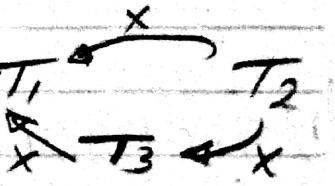
RAW, RAW, RAW, WAW



∴ Cycle → Not Serializable

c) $S_3: r_3(x), r_2(x), w_3(x), r_1(x), w_1(x)$

RAW, WAR, WAR



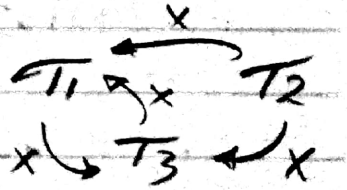
∴ No Cycle → Serializable

and Equiv to → $T_2 \to T_3 \to T_1$

$r_2(x), r_3(x), w_3(x), r_1(x), w_1(x)$

d) $S_4: r_3(x), r_2(x), r_1(x), w_3(x), w_1(x)$

WAR, WAR, WAW



∴ Cycle → Not Serializable

سؤال زي ده تحله ازاي ؟

$T_i \xrightarrow{x} T_j$

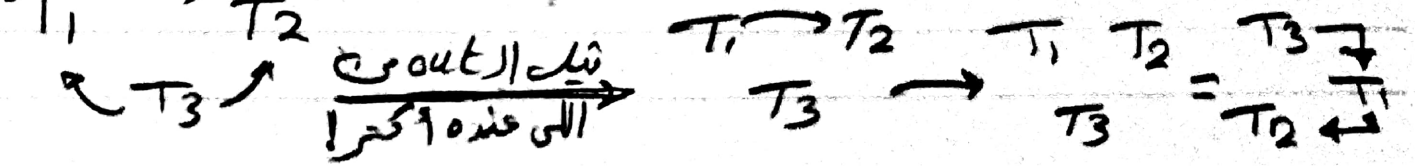
$\begin{cases} w_i(x) \to r_j(x) \text{ [RAW]} \\ r_i(x) \to w_j(x) \text{ [WAR]} \\ w_i(x) \to w_j(x) \text{ [WAW]} \end{cases}$

١) هتشوف ورا ايه Rules دي وتربط!

٢) لو الجراف فيه Cycle يبقى مش Serializable

٣) لو مفيش ، يبقى Serializable و Equivalent لـ SerialSch

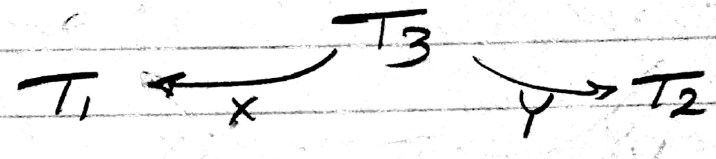
حسب الـ TopologicalSort بتاعي



17.23 $T_1 = r_1(x), r_1(z), w_1(x)$, $T_3 = r_3(x), r_3(y), w_3(y)$
$T_2 = r_2(z), r_2(y), w_2(z), w_2(y)$

a) $S_1 : r_1(x), r_2(z), r_1(z), r_3(x), r_3(y), w_1(x), w_3(y), r_2(y), w_2(z), w_2(y)$

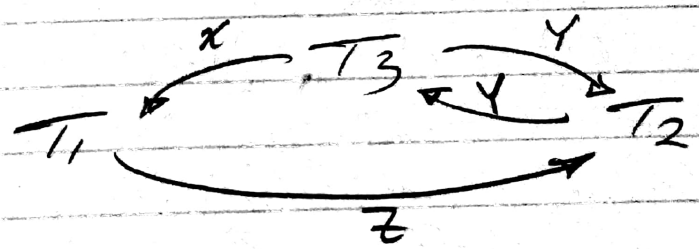
WAR WAW



Conflict Serial
$T_3 \rightarrow T_1 \rightarrow T_2$
or $T_2 \rightarrow T_3$

b) $S_2 : r_1(x), r_2(z), r_3(x), r_1(z), r_2(y), r_3(y), w_1(x), w_2(z), w_3(y), w_2(y)$

WAR WAR WAR WAW



cycle → Not Serializable