

# CM4 Implémentation BDD

Robin VAN DE MERGHEL

2023-26-04

## Rappel du cours précédent

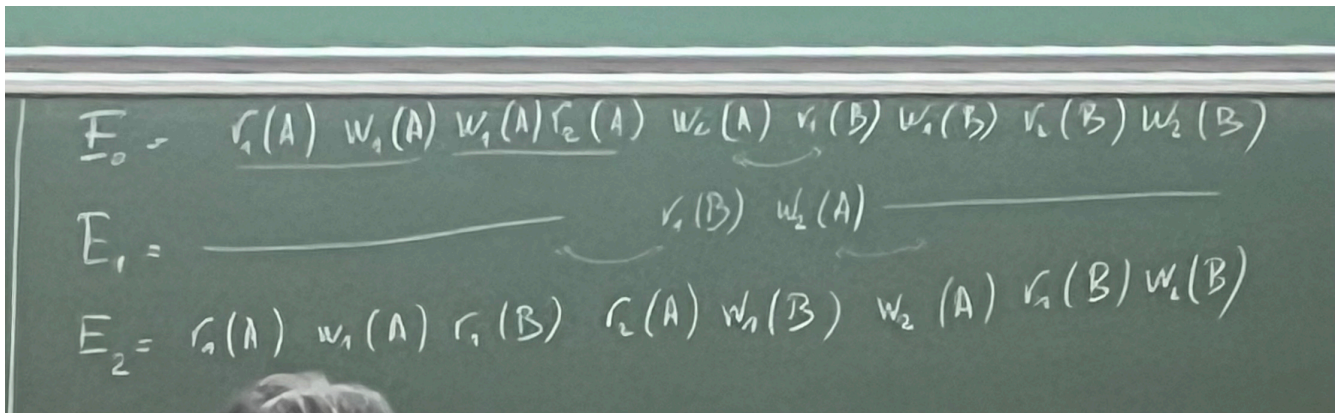
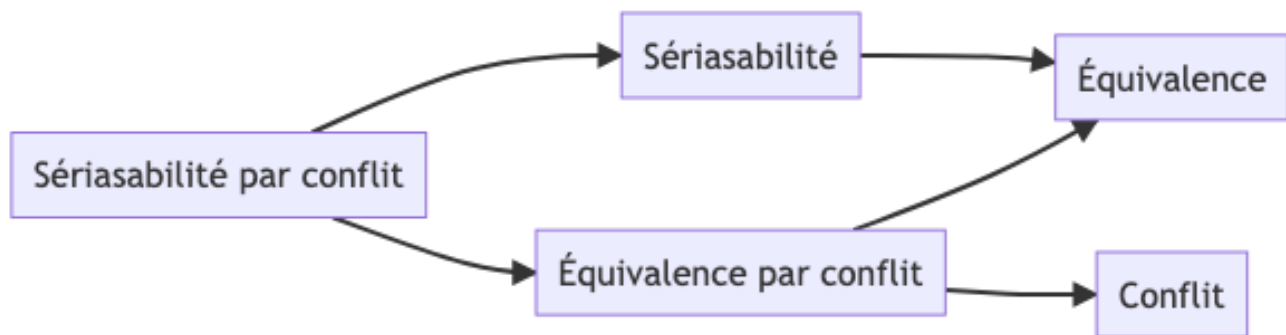


Figure 1: Image Avec l'équation au dessus

On a montré par inversions successives que  $E_0$  est équivalente par conflit à  $(T_1, T_2)$ . Cela constitue une preuve de la sèriabilité par conflit de  $E_0$ , donc  $E_0$  est sèriable.

## Graphe de précédence

On utilise le graphe de précédence pour décider si une exécution est sèrialisable par conflit.

Pour une exécution donnée, on organise ses transactions vis à vis d'un ordre de précédence.

### Définition

On dit que  $T_i$  précède  $T_j$  dans une exécution  $E_1$  noté  $T_i \prec_{E_1} T_j$  s'il y a deux opérations  $O_i$  dans  $T_i$  et  $O_j$  dans  $T_j$  dans  $T_j$  telles que :

1.  $O_i$  précède  $O_j$  dans  $E_1$ .

2.  $O_i$  et  $O_j$  s'applique sur le même élément de la base de données.
3.  $O_i$  et  $O_j$  sont des opérations d'écriture.

### Exemple

$$E = W_2(X) R_1(X) R_3(X) W_1(X) W_2(Y) R_3(Y) R_2(Z) R_3(Z)$$

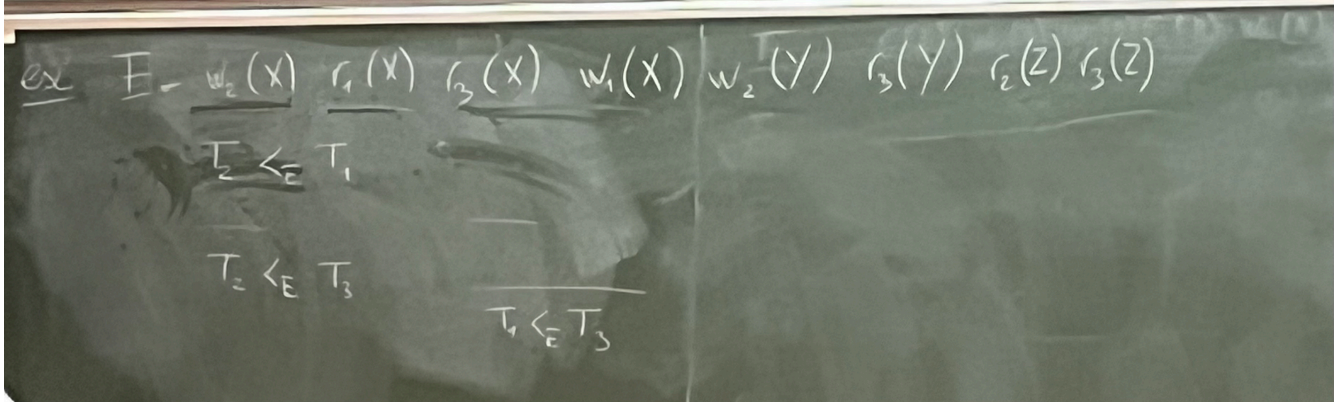


Figure 2: Image Avec l'équation au dessus

**Remarque :**  $T_i \prec_E T_j$  à cause de  $O_i$  et  $O_j$ , alors on sait que l'ordre  $O_i$  et  $O_j$  ne pourra pas être inversé dans les exécutions équivalentes par conflit à  $E$ .

De plus, si  $E$  est sérialisable par conflit, dans toute exécution en série équivalente par conflit à  $E$ ,  $T_i$  apparaîtra nécessairement avant  $T_j$ .

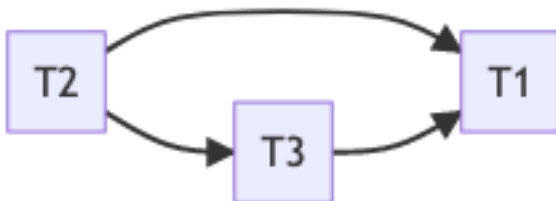
### Définition

Le graphe de précédence d'une exécution  $E$  est un graphe orienté où :

- les noeuds du graphe sont les transactions de  $E$
- il y a une arête de  $T_i$  vers  $T_j$  si  $T_i \prec_E T_j$

### Exemple

Le graphe de précédence du dernier exemple :



### Proposition

Une exécution est sérialisable par conflit si et seulement si son graphe de précédence ne contient pas de cycle (acyclique).

### Exemple

$$E' = R_2(A) R_1(B) W_2(A) R_2(B) R_3(A) W_1(B) W_3(A) W_2(B)$$

On obtient comme données :

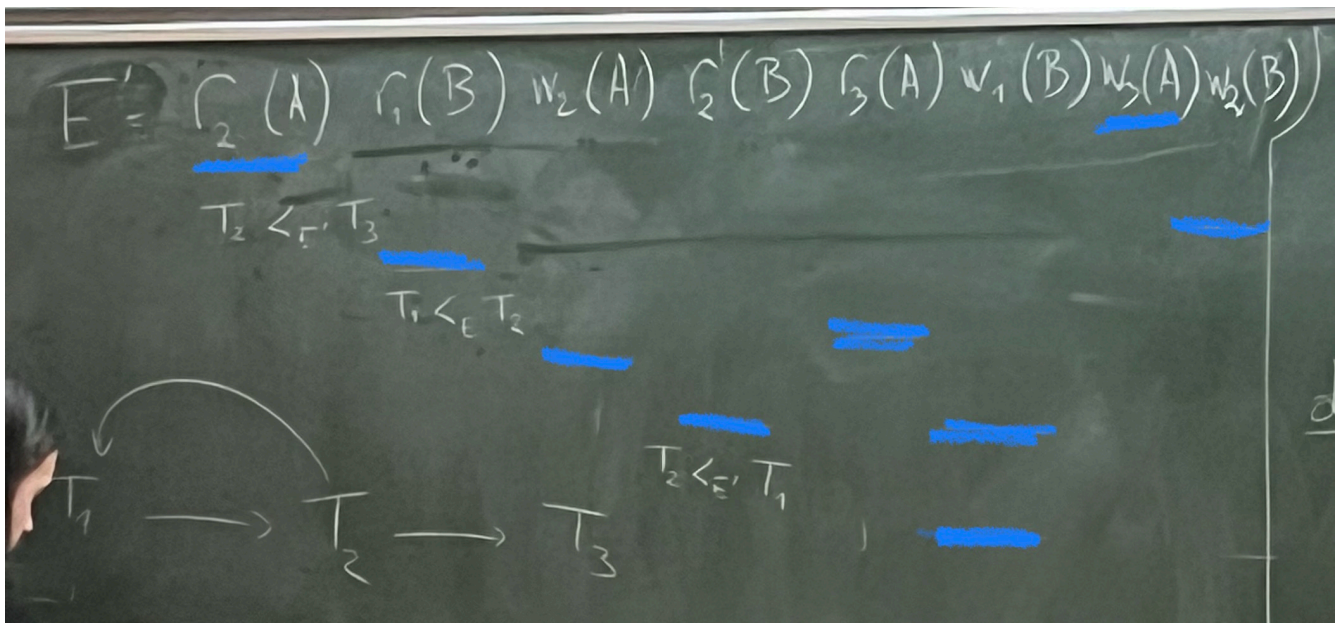


Figure 3: Exemple 2

- $T_2 \prec_{E'} T_3$
- $T_1 \prec_{E'} T_2$
- $T_2 \prec_{E'} T_1$

On obtient le graphe suivant :

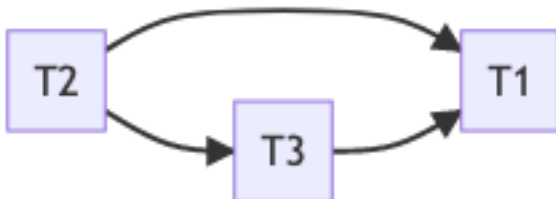


### Proposition

Si  $E$  est s  rialisable par conflit, alors  $E$  est   quivalente par conflit    toute ex  cution en s  rie des transactions de  $E$  o    $T_i$  appara  t avant  $T_j$  d  s qu'il existe une arr  te de  $T_i$  vers  $T_j$  dans le graphe de pr  c  dence de  $E$ .

### Exemple

Le graphe de pr  c  dence de l'exemple avec  $E$  :



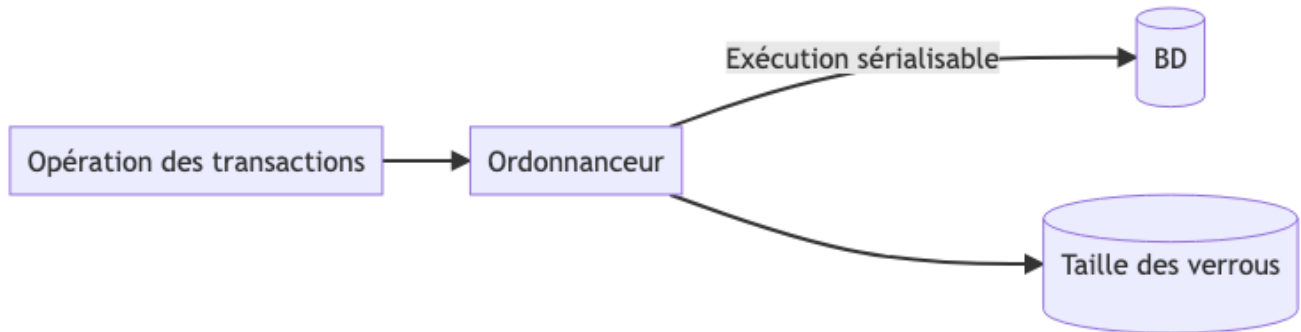
On sait que  $E$  est   quivalente par conflit     $(T_2, T_3, T_1)$ .

### Assurer la s  riabilit      l'aide de verrous

En pratique : les op  rations de chaque transaction arrivent dans un certain ordre au cours du temps. On ne cherche plus    savoir si l'ex  cution dans cet ordre est s  rialisable, mais on cherche    diff  rer certaines de ces op  rations (un

minimum) pour conserver la s rialisabilit .

Pour ce faire, les transactions vont poser des verrous sur les  l ments de la base de donn es.



## Verrous

On consid re deux types de verrous :

- **Verrou partag ** (shared lock) : n cessaire pour lire un  l ment
- **Verrou exclusif** (exclusive lock) : n cessaire pour  crire un  l ment

et des op rations correspondantes :

- $s_i(x)$  : la transaction  $T_i$  obtient un verrou partag  sur l' l ment  $x$
- $x_i(x)$  : la transaction  $T_i$  obtient un verrou exclusif sur l' l ment  $x$
- $u_i(x)$  : la transaction  $T_i$  rel che ses verrous sur les  l ments  $x$

Dans une ex cution, on dit que  $T_i$  a un verrou partag  ou exclusif entre  $s_i(x)$  (ou  $x_i(x)$ ) et la premi re op ration  $u_i(x)$  qui suit.

## Exemple

$$s_1(A) \ r_1(A) \ u_1(A) \ s_1(B) \ r_1(B) \ x_1(A) \ w_1(A) \ u_1(A) \ u_1(B)$$

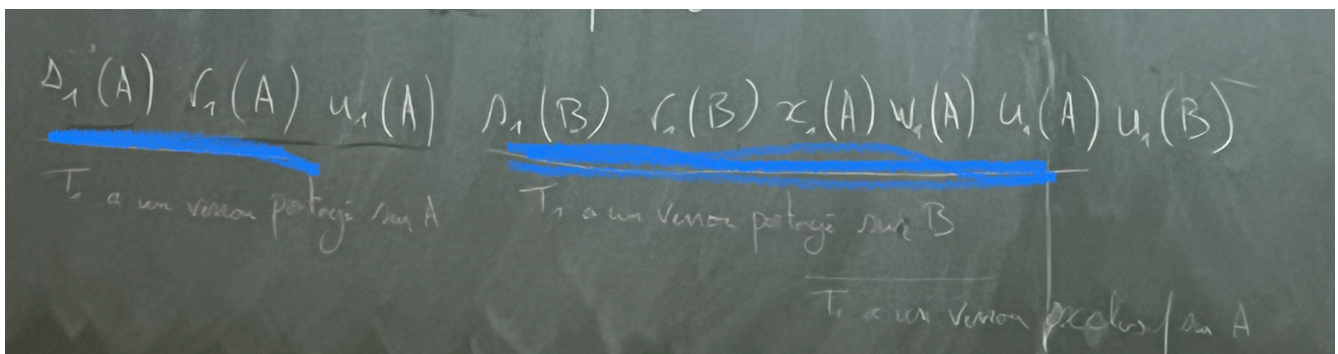


Figure 4: Exemple 3

Pour assurer le bon usage des verrous, on impose que :

- Sur chaque transaction  $T_i$ 
  - Une lecture  $r_i(x)$  ne peut  tre ex cut e que si  $T_i$  a un verrou partag  ou exclusif sur  $x$ .
  - Une  criture  $w_i(x)$  ne peut  tre ex cut e que si  $T_i$  a un verrou exclusif sur  $x$ .
  - Tous les verrous qui sont obtenus sont lib r s avant la fin de la transaction.
- sur les ex cutions,   chaque instant, un  l ment de la base peut- tre
  - Soit v rouill e de mani re exclusive par une unique transaction  $T_i$ .
  - Soit v rouill e de mani re partag e par plusieurs transactions  $T_i$ .
  - Mais pas les deux   la fois.