C018SA-W1-S3

SEMAINE 1 : Transactions et concurrence

1. Introduction : les transactions
2. Les problèmes
3. **Sérialisabilité**
4. Estampillage
5. Verrouillage à 2 phases
6. Degrés d'isolation dans les SGBD
7. Verrouillage hiérarchique

Objectifs du contrôle de concurrence

Une transaction = une séquence d'opérations.

Transactions sérielles =
exécuter toute la transaction T_1 puis T_2 etc…
= pas de problème !

La **sérialisabilité** permet de répondre au problème de **l'isolation** dans le cas de transactions qui s'enchevètrent

Il faut permettre :

- La meilleure fluidité possible
- La meilleure performance possible
- D'éviter les blocages

Formalisation : opérations de lecture et d'écriture

- On travaille au grain du nuplet, noté x_i
  - x_0 = (0, « Alice », 1000), x_1 = (1, « Bob », 750) et x_2 = (2, « Charlie », 100)
- Une requête lisant une valeur d'un nuplet donné x_i correspond à une opération de **lecture** sur ce nuplet, notée $R(x_i)$.
  - **SELECT SOLDE FROM COMPTE WHERE NC = 0** est une opération de lecture de x_0, notée **$R(x_0)$**
- Une requête modifiant une valeur d'un nuplet donné x_i correspond à une opération d'**écriture** sur ce nuplet, notée $W(x_i)$.
  - **UPDATE COMPTE SET SOLDE = SOLDE + 100 WHERE NC = 1** est une opération d'écriture de x_1, notée **$W(x_1)$**

Opérations conflictuelles entre transactions

- Deux opérations a et a’, exécutées respectivement par deux transactions différentes T et T' sont dites **conflictuelles** si l'exécution des deux séquences a;a’ (a puis a’) et a’;a (a’ puis a) est *susceptible* de conduire à des résultats différents.

Exemple d'opérations conflictuelles

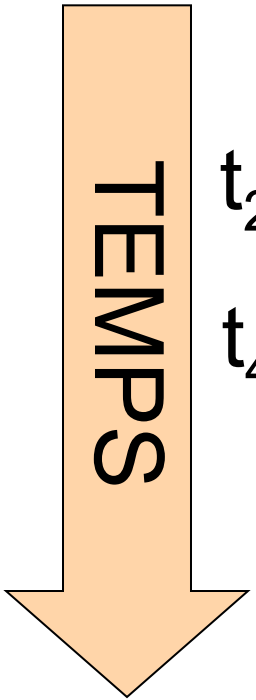
T_1 : transférer le solde de Bob à Alice

T_2 : transférer le solde de Charlie à Alice

t_1 SELECT SOLDE INTO S1
WHERE NC = 1

t_3 UPDATE COMPTE
SET SOLDE = S1
WHERE NC = 0

…



t_2 SELECT SOLDE INTO S2
WHERE NC = 2

t_4 UPDATE COMPTE
SET SOLDE = S2
WHERE NC = 0

…

Opérations conflictuelles

	R(X)	W(X)
R(X)	NON	OUI
W(X)	OUI	OUI

Transactions sérielles =
exécuter toute la transaction T_1 puis T_2 etc…

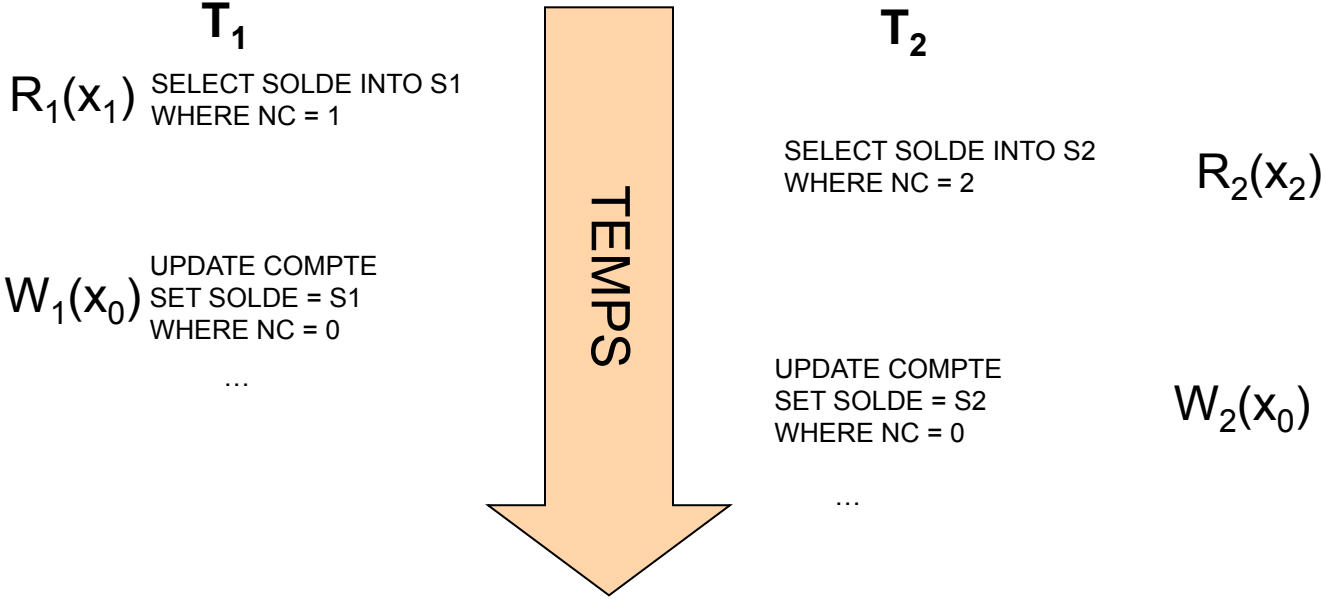
Pour garantir l'isolation des transactions, il suffit de pouvoir exécuter intégralement une transaction, puis une autre, etc.

Est-ce possible ? Et si oui, dans quel ordre faut-il le faire ?

Solution : avoir (un ordonnancement) des transactions SERIALISABLE(S)
⇔ équivalent à des transactions sérielles

Ordonnancement

Définition : Un *ordonnancement* d'opérations de plusieurs transactions est composé d'opérations atomiques de type R ou W sur des données.



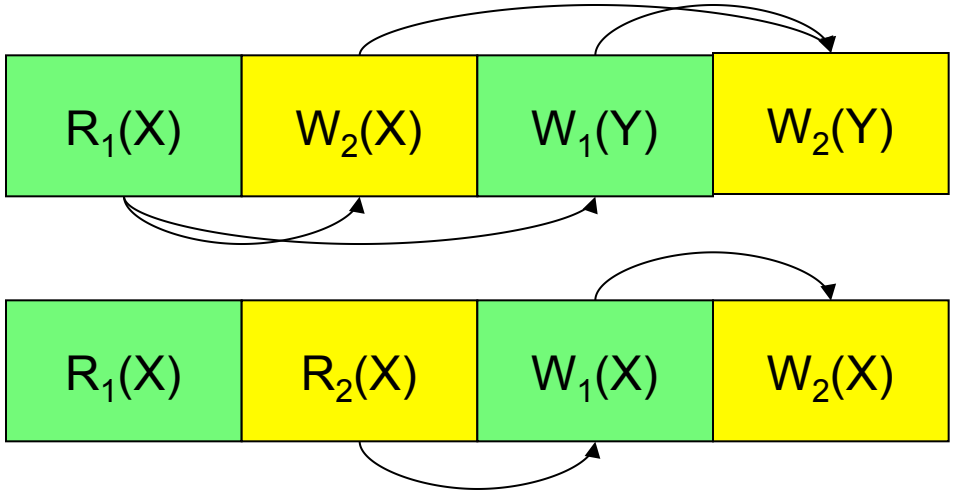
$$\Omega = R_1(x_1);R_2(x_2);W_1(x_0);W_2(x_0)$$

Ordonnancement

- **Définition :** Un ordonnancement A est *équivalent* à un ordonnancement A' **ssi** le résultat produit et observé est le même.
- **Proposition : Si** A' est un ordonnancement produit à partir d'un ordonnancement A en ne permutant que des opérations non conflictuelles **alors** A et A' sont équivalentes.

Sérialisabilité : Définition Informelle

Un ordonnancement (d'opérations atomiques) est **sérialisable** si et seulement si tous les conflits sont « dans le même sens »



SERIALISABLE

PROBLEME

Sérialisabilité : Définition équivalente

Un ordonnancement est sérialisable

si et seulement si :

Il peut être transformé en un ordonnancement série par permutations successives d'opérations ne constituant pas une paire d'opérations conflictuelles.

Une telle transformation, si elle est possible, fournit effectivement un ordonnancement série équivalent.

Détection de la non sérialisabilité : graphe de précédence 1/2

Graphe de précédence de l'exécution → sérialisable ou non

Sur chaque objet, deux actions non permutables impliquent une relation de précédence entre les transactions correspondantes

Définition de la précédence :

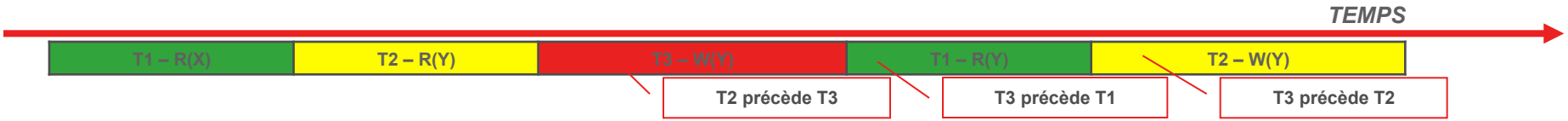
Si :
- T_1 applique une action A_1 avant que T_2 applique A_2
- A_1 et A_2 sont non permutables

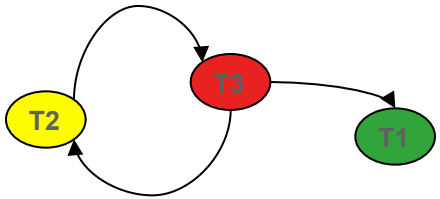
Alors :
- T_1 précède T_2
- ajouter un arc de précédence de T_1 vers T_2

Détection de la non sérialisabilité : graphe de précédence 2/2

Décision : Graphe de précédence sans boucle → **exécution sérialisable** (il suffit de suivre les arcs)

Coût = O(*nombre d'opérations*)





Que faire si des transactions ne sont pas sérialisables ?

Sérialisabilité : Définition équivalente

Un ordonnancement est sérialisable ssi :

Le graphe de précédence est sans circuit

Rendre l’exécution sérialialisable

- Protocole d’estampillage : **stratégie curative**
- Protocole de verrouillage : **stratégie préventive**