Données Réparties

Thibault BERNARD

thibault.bernard@univ-reims.fr

Sommaire

- Introduction
- Gestion de la concurrence
- Reprise après panne
- Gestion des données dupliquées

Sommaire

- **Introduction**
- Gestion de la concurrence
- Reprise après panne
- Gestion des données dupliquées

Objectifs / Avantages de la répartition

- Augmentation des performances
- Fiabilité / Disponibilité
- Evolution / Extensibilité
- Partage
- Contrôle partagé
- Hétérogénéité

Problèmes / Inconvénients

- Mise en œuvre : très complexe
- Problème du contrôle distribué
- Support de communication : très lent

Architectures

- Centralisées
- Réparties

IMPORTANT : Transparence pour l'utilisateur:

- Localisation
- Partitionnement
- Duplication

Conception d'une base de données

- Fragmentation
- Schéma de fragmentation
- Schéma d'allocation
- Autres techniques

Fragmentation

- Par relations
- Horizontale : restriction / union
- Horizontale dérivée : jointure / union
- Verticale : projection / jointure
- Mixte : combinaison de jointure projection et restriction / combinaison de jointure et union

Exemple : Relation Vol

VolNum	IDCompagnie	HeureDépart	HeureArrivée	Départ	Arrivée
CX260	CX	13h10	8h00	Paris CDG	Hong-Kong
AF125	AF	13h15	17h45	Paris CDG	Mexico
AM437	AM	18h40	19h50	Mexico	Guadalajara
CA179	CA	13h20	17h00	Paris CDG	Houston
AA286	AA	18h00	21h00	Boston	Houston
AF132	AF	18h25	14h00	Paris CDG	Tokyo

Exemple : Relation Compagnie

ID Compagnie	Nom	Siege social	Pays
AF	Air France	Paris	France
CX	Cathay Pacific	Hong-Kong	Chine
AM	AeroMexico	Mexico City	Mexico
AA	American Airlines	New York	United States of America
CA	Continental Airlines	Houston	United States of America

Exercices

Ecrire un ou plusieurs scenarii de répartition des données mettant en œuvre les différentes fragmentations définies précèdemment.

Quelle est LA contrainte à prendre en compte dans le schéma de fragmentation et d'allocation? Pourquoi?

Autres Techniques

- Duplication
- Allocation Dynamique
- Fragmentation Dynamique
- Clichés

Sommaire

- Introduction
- **Gestion de la concurrence**
- Reprise après panne
- Gestion des données dupliquées

Gestion de la concurence

Définition : Une transaction est une unité de traitement séquentielle cohérente et fiable, constituée d'une suite d'opérations à réaliser sur la base de données. Appliquée à une base de données cohérente, une transaction restitue une base de données cohérente.

Opérations sur les transactions

- Start
- Commit
- Abort
- Lecture / Ecriture

Une transaction est active si elle a effectué une opération start mais pas l'opération commit ou abort.

Propriétés des Transactions

- Atomicité : Toutes les opérations sont effectuées ou aucune d'entre elles.
- Cohérence : Les transactions doivent préserver la cohérence de la base de données.
- Isolation : Les maj effectuées par une transaction ne sont visibles qu'une fois celle ci validée.
- Durabilité : Les maj sont définitives.

Les Problèmes

Soient 2 transactions : T1 et T2

T1 : x<- Lire(X)

x<-x+10

Ecrire(X,x)

T2 : y<- Lire(X)

y<-y+20

Ecrire(X,y)

Perte de mise à jour

T1 : x<- Lire(X)

T1 : x<- x+10

T2 : y<- Lire(X)

T2 : y<- y+20

T1 : Ecrire(X,x)

T2 : Ecrire(X,y)

Création d'incohérence

T1

x <- Lire(a)

x <- x+1

Ecrire(a,x)

y <- Lire(b)

y <- y+1

Ecrire(b,y)

T2

z <- Lire(a)

z <- z*2

Ecrire(a,z)

t <- Lire(b)

t <- t*2

Ecrire(b,t)

Création d'incohérence

Soit l'execution suivante:

x<- Lire(a)

x<-x+1

Ecrire(a,x)

Execution complete de T2

y <- Lire(b)

y <- y+1

Ecrire(b,y)

Lecture non reproductible

T1:

x<-Lire(A)

Afficher(x)

x<-Lire(A)

Afficher(x)

T2:

y<-Lire(A)

y<-y*2

Ecrire(A,y)

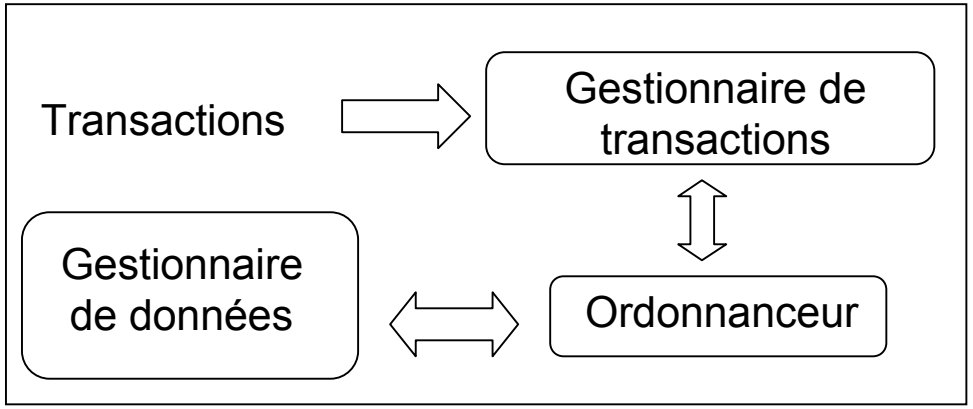
Gestion des Transactions

Gestionnaire de transaction : réaliser les exécutions des transactions.

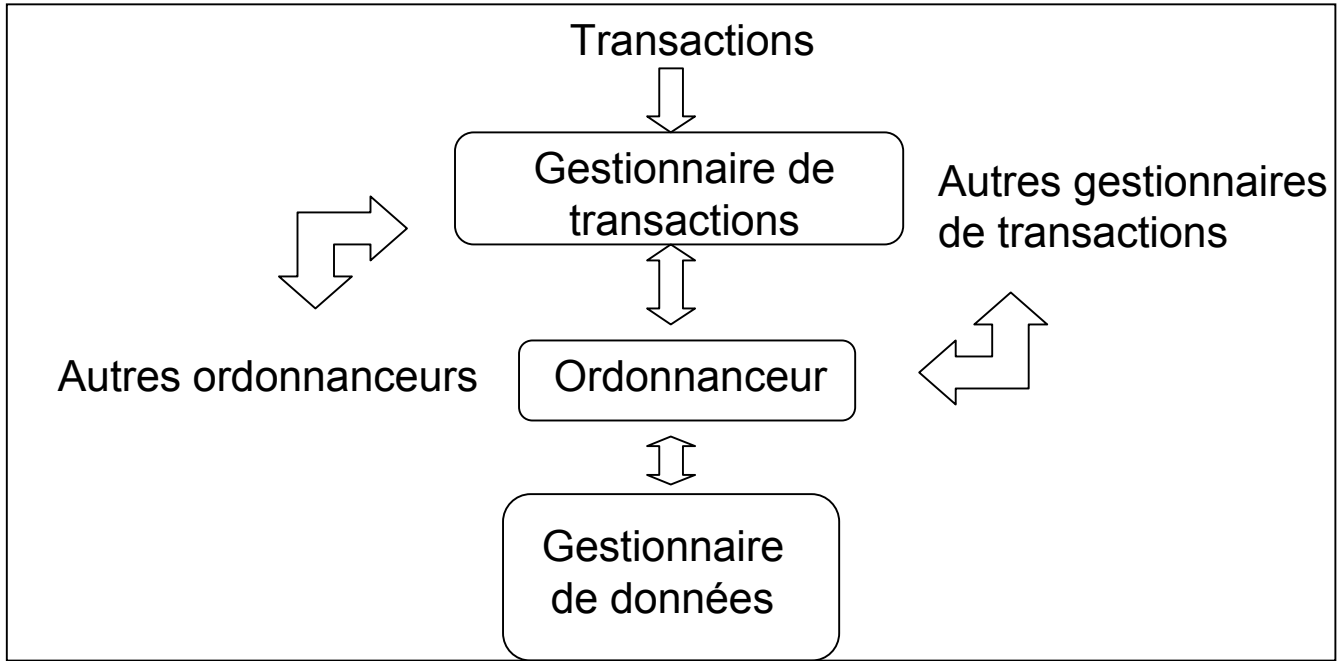
Ordonnanceur : contrôler l'ordre des opérations des différentes transactions.

Architecture

- Centralisée



- Répartie



Que fait l'ordonnanceur?

Lors de la réception d'une opération d'une transaction, il peut :

- L'executer
- La rejeter
- La mettre en attente

Sériabilisité

Définition : Une exécution d'un ensemble de transaction est dite sérialisable si elle donne pour chaque transation participante, le même resultat que l'exécution en série de ces mêmes transactions.

Conflits d'opérations

Sur 2 ou plusieurs transactions concurentes, les conflits ont lieu sur des opérations:

- Lecture - Ecriture
- Ecriture - Ecriture

Il faut donc connaître l'ordonnancement des opérations Lire et Ecrire sur une donnée.

Graphe de Précédence

G=(V,E) où V représente l'ensemble des transactions et E les relations de précédance par rapport à une donnée : un arc de i à j avec le label a indique que la Transaction Ti accede a avant Tj

Un graphe de précédence sans cycle caractérise une exécution sérialisable.

Exercices

Soit l'execution suivante :

T1 Lire(A)

T2 Ecrire(A)

T2 Lire(B)

T2 Ecrire(B)

T3 Lire(A)

T1 Ecrire(B)

Tracer le graphe de précédence associé

Exercices (suite)

Ecrire une exécution de 3 transactions qui soit sérialisable. Vérifier à l'aide d'un graphe de précédence.

Vérouillage

Afin d'ordonner les opérations des différentes transactions, il est possible d'utiliser des verrous sur les données.

2 types de verrous : Lecture Ecriture

Un verrou en Ecriture est exclusif!

Exercice

Soient les 3 transactions suivantes :

T1: Ecrire(X) Ecrire(Y)
T2: Ecrire(Y) Ecrire(Z) Ecrire(X)
T3: Ecrire(X) Ecrire(Z)

Exercice (suite)

Lock(1,X,E)
Lock(2,Y,E)
Lock(2,Z,E)
Unlock(2,Y)
Lock(1,Y,E)
Unlock(1,X)
Lock(2,X,E)
Unlock(2,Z)
Unlock(2,X)
Lock(3,X,E)
Lock(3,Z,E)
Unlock(3,X)
Unlock(3,Z)

Tracer le graphe de précédence!

Verrouillage à 2 phases

La technique du verrouillage à 2 phases consiste pour une transaction à ne plus poser de verrous après en avoir relacher un.

Verrouillage à 2 phases => sérialisable

Exercice

Proposer une exécution respectant le verouillage à 2 phases dans les 3 transactions précédantes. Tracer le graphe de précédence.

Le problème d'interblocage

Soit l'exécution suivante (respectant la technique du verouillage à 2 phases) :

Lock(T1,X,E)

Lock(T2,Y,L)

Lock(T1,Y,E)

Lock(T2,X,L)

Unlock(T1,X)

Unlock(T2,X)

Unlock(T1,Y)

Unlock(T2,Y)

Que se passe-t-il?

Propriété du contrôle de concurrence

Sureté : Toute exécution concurente est sérialisable.

Vivacité : Toute transaction se termine en temps fini.

Graphe des attentes

Outil pour analyser les situations d'interblocages

Les nœuds du graphe représentent les différentes transactions et un arc entre i et j indique que la transaction i attend un verrou que possède la transaction j.

Detection de l'interblocage

Le graphe des attentes permet de détecter les situations d'interblocage. Celui ci est construit localement au sein de chaque ordonnanceur. En cas d'interblocage ce dernier choisit une transaction à annuler. Ce choix dépend:

- De l'effort investi dans la transaction.
- Du cout de l'annulation de cette transaction.
- Du nombre de circuits qui contient la transaction.

Cadre distribué ?

On distingue alors un site : le détecteur qui fréquemment fait l'union des graphes des attentes locaux à chacun des ordonnanceurs et indique le cas échéant la transition à annuler.

Cadre distribué ?

Oui mais, n'y a-t-il pas des cas où l'on risque de détecter un interblocage sans que celui ci soit effectif?

Autres méthodes pour résoudre l'interblocage

- Timeout
- Construction d'un ordre total
- Estampillage
- Certification

Exemple Estampillage

Soit w(x) la plus grande estampille d'écriture sur x et r(x) la plus grande estampille de lecture sur x.

Algo Lecture

Si est(Ti)<w(x)

Annuler Ti

Sinon

Effectuer lecture

r(x) <- max(r(x),est(Ti))

Algo Ecriture

Si est(Ti)<w(x) ou est(Ti)<r(x)

Annuler Ti

Sinon

Effectuer Ecriture

w(x)<-est(Ti)

Sommaire

- Introduction
- Gestion de la concurrence
- **Reprise après panne**
- Gestion des données dupliquées

2 Problèmes

- ANNULATION EN CASCADE
- ANNULATION DES TRANSACTIONS VALIDEES

Le problème des annulations

Soient 2 transactions T1 et T2 et 2 données x et y valant initialement la valeur 1.

Soit l'exécution suivante :

T1 Lire(x)

T1 x<-x+1

T1 Ecrire(x)

T2 Lire(x)

T2 Lire(y)

T2 y<-y+2

T2 Ecrire(y)

T1 Annulée

Que se passe-t-il?

Principe de recouvrabilité

Le principe de recouvrabilité consiste à ne jamais annuler une transaction validée.

Execution recouvrable

Une exécution est dite recouvrable si pour chaque transaction Ti, la validation de Ti précède la validation de toutes les transactions telles que :

1. Tj lit x après que Ti est écrit dessus
2. Ti n'est pas annulée avant que Tj ne lise x
3. Chaque transaction qui écrit sur x pendant la période entre Ti ecrit sur x et Tj lit sur x, est annulée

Exemple (1)

T1 Lire(x)
T1 x<-x+1
T1 Ecrire(x)
T2 Lire(x)
T2 Lire(y)
T2 y<-y+2
T2 Ecrire(y)
T2 Validée

Non recouvrable !

Exemple (2)

T1 Lire(x)
T1 x<-x+1
T1 Ecrire(x)
T2 Lire(x)
T2 Lire(y)
T2 y<-y+2
T2 Ecrire(y)
T1 annulée

Recouvrable, mais annulation en cascade !

Solution

Une solution aux annulations en cascade est de ne permettre la lecture de valeurs écrites que par des transactions déjà validées. Ceci assure aussi la recouvrabilité d'une exécution.

Types de défaillance

- Panne de transitions (vu précédemment)
- Panne de système ou de site (checkpoint)
- Panne de supports (duplication)
- Panne de communications

Architecture

Le gestionnaire de données est divisé en 2 parties :

- Gestionnaire de reprise (gère les opérations sur les transactions)
- Gestionnaire de cache (gère le transfert mémoire stable et mémoire cache)

2 types de mémoires :

- Mémoire cache
- Mémoire stable

Panne de système

Le principe est d'effectué un journalisation des opérations qui ont été effectuées sur la bases de données.

L'opération restart du GR pourra alors effectuer 2 actions :

- Undo
- Redo

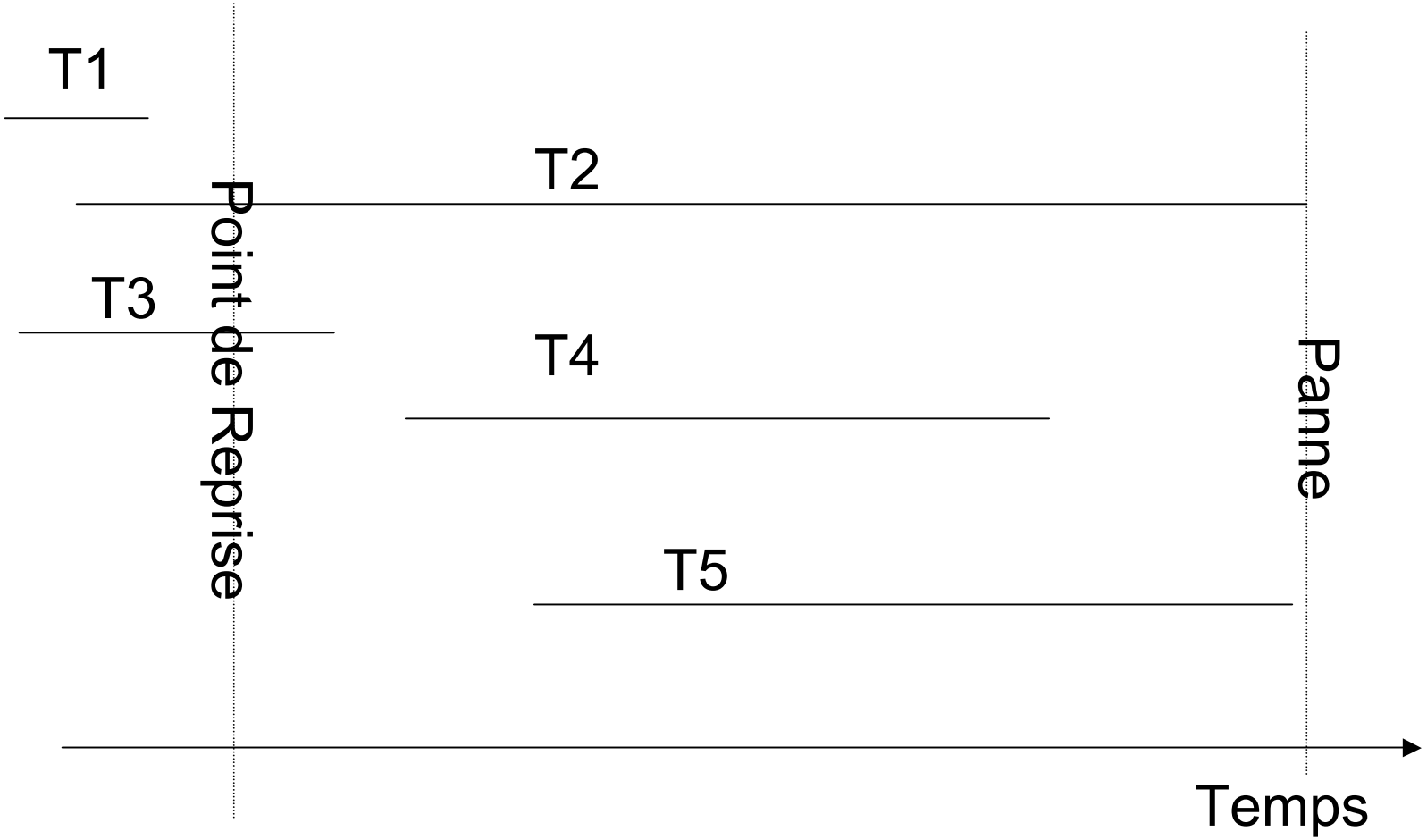
Principe de la reprise

Si l'on trouve dans le journal, une Transaction comprenant un start sans le commit associé, l'action Undo est effectuée sur cette transition.

Dans le cas contraire, c'est-à-dire on trouve un start et un commit, l'action Redo est effectuée sur cette transition.

On utilise des points de reprise afin d'éviter le parcourt intégral du journal.

Exemple



Panne de support

Principe analogue à la panne de site avec duplication de la base de données sur d'autres supports.

Cas distribué

Chaque transaction appartient à un seul site. Néanmoins celle ci peut accéder aux données d'autres sites.

Il convient donc à chacun des sites de s'accorder sur l'opération finalisant l'exécution de la transaction : Problème de consensus distribué.

Consensus Distribué

Le système est composé comme suit :

- Un coordinateur (le site propriétaire de la transaction)
- Des participants (les sites hébergeants des données lues ou écrites par la transaction)

Validation à 2 phases

- Le coordinateur demande à chaque site de s'accorder sur commit ou abort.
- Chaque site répond au coordinateur sur commit ou abort. Si le choix est abort le site annule tout de suite la transaction.
- Le site collectionne les réponses des sites. Si il y a aucune reponses abort, commit est décidé. Sinon abort est décidé. Le coordinateur relaie la décision aux sites.

Cas de panne temporaire de site

Un site venant de subir une défaillance s'informe auprès d'autres sites pour connaître la décision qui avait été prise.

Panne de communication (bref résumé)

- Pas de réponse d'un site (détection par timeouts) : abort est décidé (par le coordinateur et le site ne communiquant pas)
- Un site ne reçoit pas la réponse du coordinateur. Il tente de s'informer auprès d'autres sites.

Sommaire

- Introduction
- Gestion de la concurrence
- Reprise après panne
- **Gestion des données dupliquées**

Gestion de données dupliquées

- Assurer la cohérence d'une donnée sur chacun des sites la contenant.
- L'utilisateur ne doit pas voir la donnée comme dupliquée.
- Problème des lecteurs - redacteurs

Le problème

- Une action d’écriture exclut toute autre action.
- Une action de lecture peut être combinée avec d’autres actions d’écriture.

Solution

Mise en place d'un système de quorum :

- Tout accès en lecture nécessite r permissions.
- Tout acces en écriture nécessite w permissions.

Avec N le nombre total de permissions (nombre d'instance de la donnée), r+w>N et 2*w>N