Reprise sur panne

Chapitre 4

Hmida HMIDA

Avril 2022

DSI2 - ISET Bizerte
D'après le cours de Philippe Rigaux

PLAN

1. Introduction
2. Buffer et disque
3. Reprise après une panne mémoire
4. Panne de disque

Introduction

DÉFINITIONS ET OBJECTIF

Panne :

- Dysfonctionnement d'un élément du système
- Causes :
  - Annulation de transaction (Exception, Rollback)
  - Coupure de courant
  - Secteur défectueux
- 2 Familles :
  - Panne RAM : Buffer de la BD (cache) : Panne légère
  - Panne disque : Panne lourde

Reprise (Recovery) :

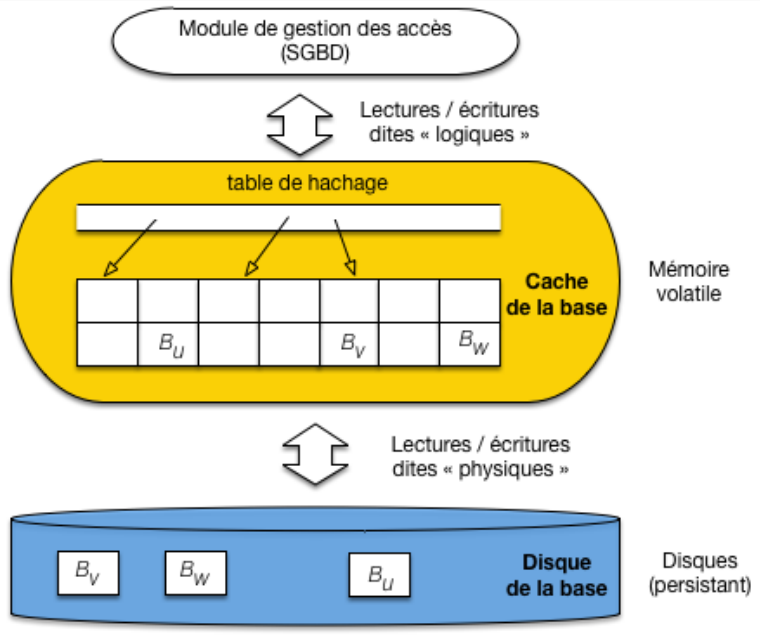
- Objectif : Ramener la BD au dernier **état validé** avant la panne
- Propriétés ACID et pannes
  - Recouvrabilité et Atomicité : Si une transaction n'est pas validée par un commit, elle peut être annulée par un Rollback.
  - Durabilité et atomicité : Les modifications apportées par une transaction validée par un Commit sont permanentes.

ÉTAT DE LA BASE

- État résultant des transactions validées
- Les données modifiées $\rightarrow$ image après
- Les données avant modifications $\rightarrow$ image avant
- **Problème :**
  - Au commit l'image après remplace l'image avant
  - Modification des données en RAM (buffer)
  - Assurer l'atomicité de cette opération

Buffer et disque

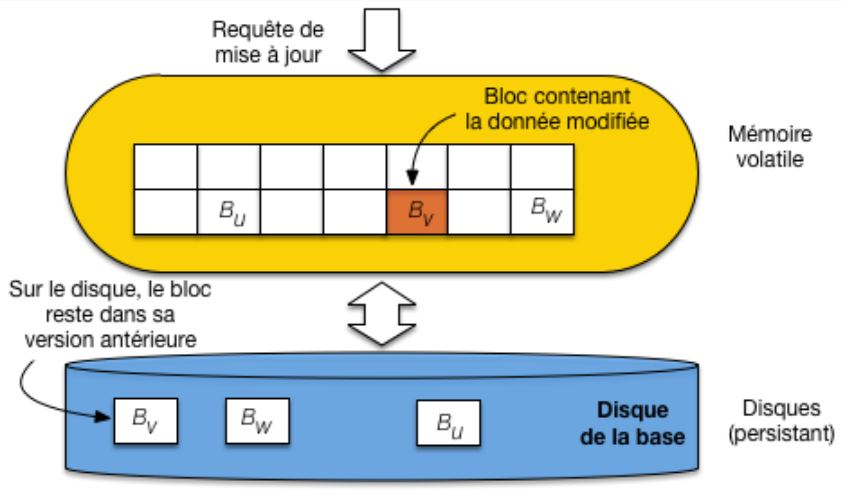
LECTURE AVEC BUFFER



source : cours Philippe Rigaux

- Échange entre RAM et disque
- Toute opération de lecture contient l'adresse du bloc à lire.
- Si le bloc est dans le buffer, le système accède à la donnée dans le bloc, et la retourne.
- Sinon il faut d'abord lire un bloc du disque, et le placer dans le buffer.
- Le SGBD garde en mémoire les blocs après lecture.
- Buffer plein : Least Recently Used, FIFO, ...

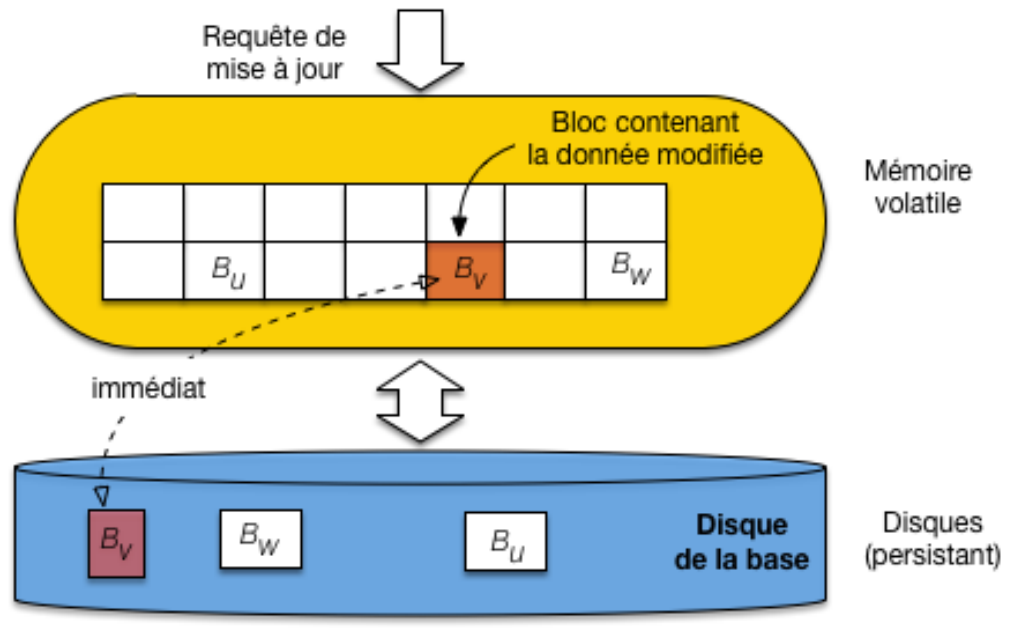
MISE À JOUR AVEC BUFFER



source : cours Philippe Rigaux

- Lire le bloc
- Modifier le bloc dans le buffer
- Marquer le bloc par un bit de consistance
- Synchronisation avec le disque
  - Immédiate
  - Différée
  - Opportuniste
- La synchronisation a un impact sur la reprise

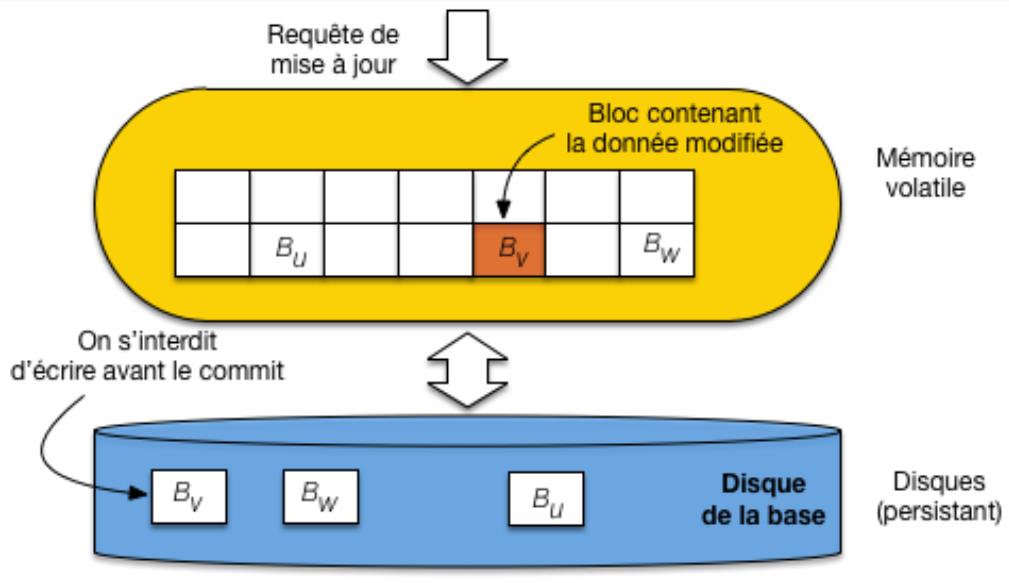
MISE À JOUR IMMÉDIATE



source : cours Philippe Rigaux

- Propager la modification sur le disque immédiatement
- Écraser l'image avant
- Écritures aléatoires : moins performant

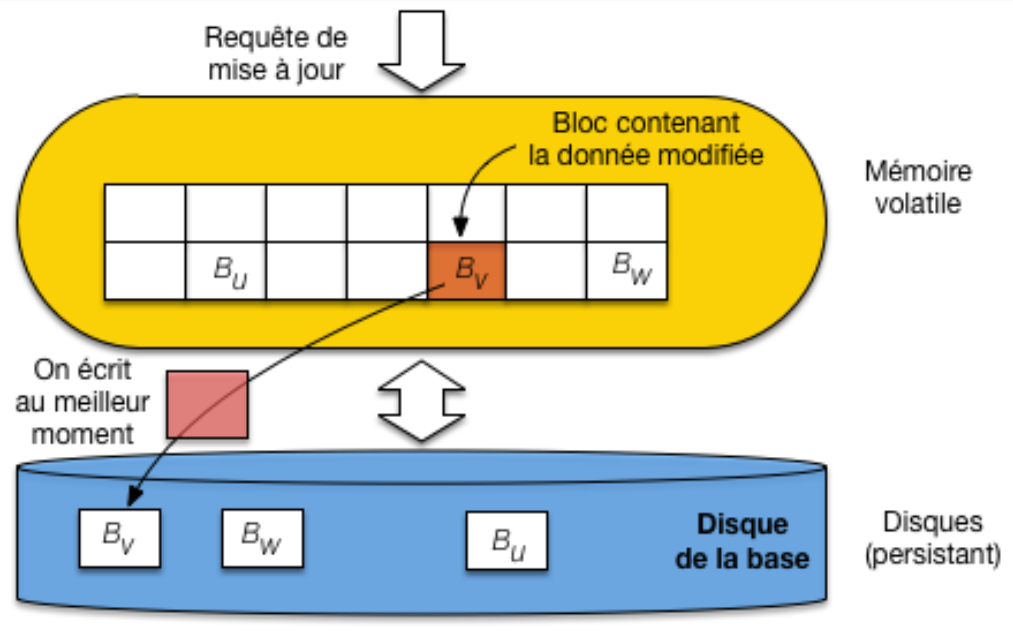
MISE À JOUR DIFFÉRÉE



source : cours Philippe Rigaux

- Attendre le commit
- L'image avant n'est pas modifiée
- Surcharge du buffer

MISE À JOUR OPPORTUNISTE



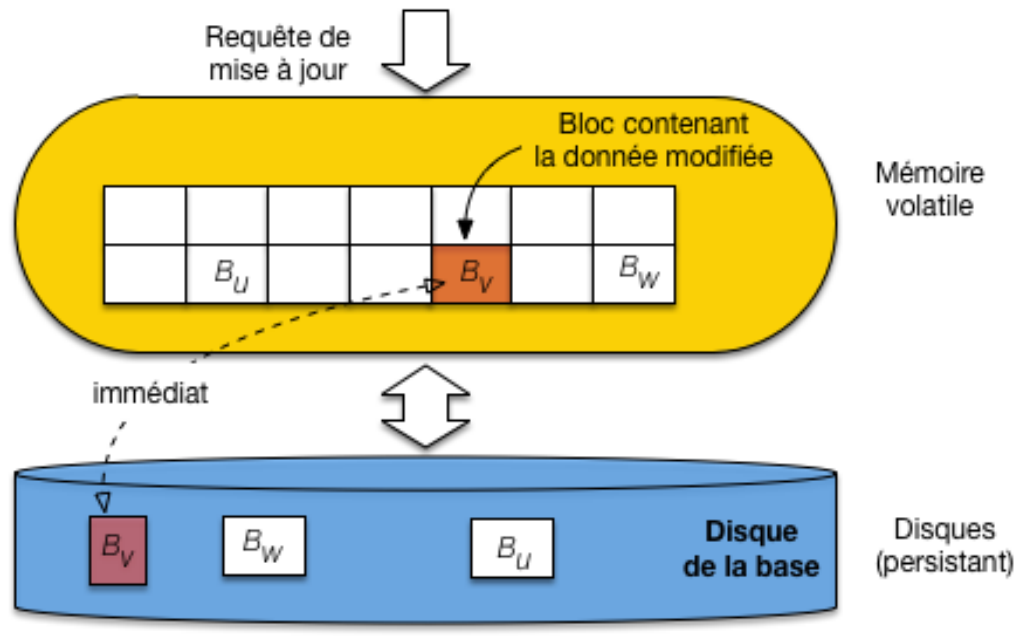
source : cours Philippe Rigaux

- Choisir le **meilleur** moment pour la synchronisation
- Plusieurs écritures successives dans le buffer

Reprise après une panne mémoire

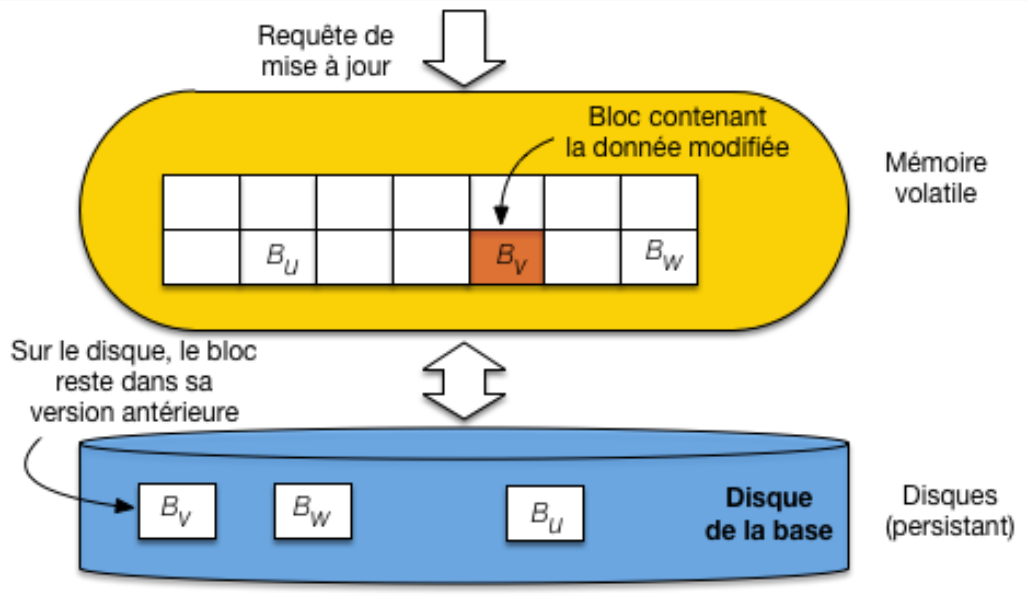
SCÉNARIO

- Une transaction en cours d'exécution
  - L'image avant est sur le disque
  - L'image après est dans le buffer
  - Commit : écrire les blocs modifiés à partir du buffer vers le disque
  - Rollback : supprimer les blocs modifiés du buffer
- Contenu de la RAM $\rightarrow$ buffer $\rightarrow$ l'image après est perdue
- Avec MAJ immédiate/opportuniste :
  - L'image avant est écrasée par l'image après
  - Rollback impossible



SCÉNARIO

- Avec MAJ différée :
  - Panne après quelques écritures de blocs modifiés
  - Rollback et Commit impossibles



- Garantir un Commit : Les données modifiées sont sur le disque → image après
- Garantir un Rollback : Les données avant modifications sont sur le disque → image avant

⇒ Les 2 images doivent être sur disque : log (journal des transactions)

JOURNALISATION

- Enregistrement des opérations d'écritures
  - Log Logique : Enregistre les opérations de haut niveau
  - Log Physique : Enregistre les valeurs écrites
- Fichier séquentiel
- Pas de suppression
- Stratégie :
  - Opportuniste pour la BD
  - Immédiate pour le log
- Événements :
  - Start(T)
  - Write(T, X, old_value, new_value)
  - Commit(T)
  - Rollback(T)
  - Checkpoint

LOG ET COMMIT/ROLLBACK

- Point de commit
  - Forcer l'écriture des modifications dans le log suite à un commit
  - L'état de la base est dans le log
  - Une transaction est validée quand l'instruction commit est écrite dans le log
- Gestion du Rollback
  - Un bloc modifié mais pas encore validé peut être écrit dans la base
  - $\rightarrow$ En cas de panne il faut reprendre l'image avant
  - Write-ahead logging : Un bloc modifié est écrit dans le log avant d'être écrit dans la base
  - le log contient aussi des modifications non validées

ALGORITHMES DE REPRISE

Redo / Undo : Refaire / Défaire

- Conséquences de l'écriture opportuniste
  - Des modifications validées qui ne sont pas encore dans la base
  - Inversement, des modifications non validées qui sont dans la base
- À la reprise, il faut donc
  - Refaire (Redo) les transactions validées
  - Défaire (Undo) les transactions en cours.
- Ces deux opérations sont basées sur le journal :
  - Reconstituer la liste L_V des transactions validées : on trouve un commit dans le log
  - Reconstituer la liste L_A des transactions actives ou annulées : pas de commit dans le log
  - Déterminer l'image après (new_val) et l'image avant (old_val) pour chaque transaction.

ALGORITHMES DE REPRISE

Redo / Undo : Refaire / Défaire

- Undo
  - Pour chaque transaction T de L_A dans l'ordre inverse
  - Pour chaque $write(T, x, old\_val, new\_val)$, on écrit old_val dans x
- Redo
  - Pour chaque transaction T de L_V dans l'ordre d'exécution
  - Pour chaque $write(T, x, old\_val, new\_val)$, on écrit new_val dans x
- Panne de la reprise $\rightarrow$ Recommencer

Checkpoint

- En cas de panne, il faudrait en principe refaire toutes les transactions du journal, depuis l'origine de la création de la base.
- Un checkpoint écrit sur disque tous les blocs modifiés, ce qui garantit que les données validées par commit sont dans la base.

ALGORITHMES DE REPRISE

Typologie des algorithmes de reprise

- Deux paramètres
  - Permettre des modifications non validées dans la base ? (oui : Steal, Non: No-Steal)
  - Forcer l'écriture dans la base des modifications validées au commit ? (oui: Force, non : No-Force)
- Steal : Undo des modifications non validées
- No-Force : Redo des modifications validées
- 4 formes : Steal/Force, Steal/No-Force, No-Steal/Force et No-Steal/No-Force
- Steal/No-Force : le plus performant en mode normal mais le plus coûteux en reprise (nécessite Undo et Redo)

REPRISE DANS ORACLE

- Algorithme Steal / No-Force
- Journalisation
  - Journal d'images après (redo log)
  - Journal d'images avant (undo segment)
- Reprise en 2 étapes :
  - Répétition des écritures (journal d'images après)
  - Annulation des écritures non validées (journal d'images avant)
- Retrouver des versions plus anciennes des données
  - Flashback: réconstitution de la BD à un moment donné du passé
  - `CREATE TABLE nom AS SELECT ... AS OF TIMESTAMP t`

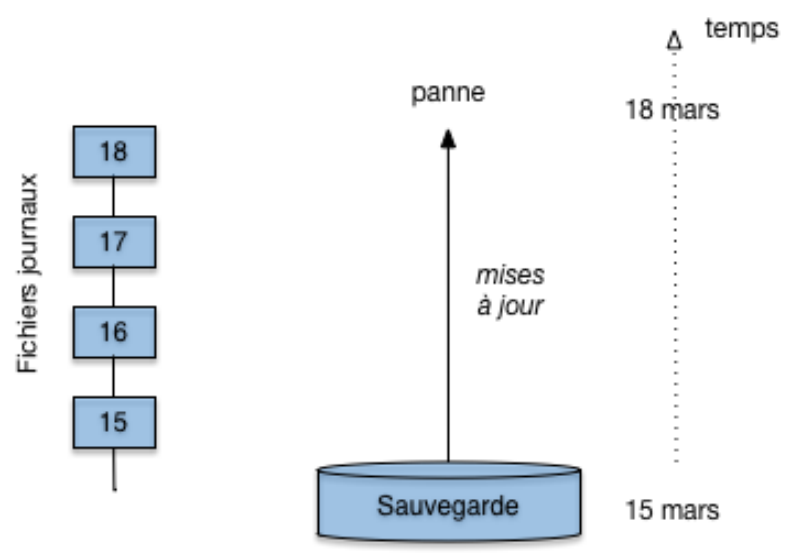
Panne de disque

SCÉNARIO

- Panne la plus grave, car on risque de perdre l'état de la base.
  1. L'état de la base est dans le fichier journal.
  2. L'état de la base est aussi dans le fichier de la base et le buffer.
- Solutions :
  - Mettre le log et la base dans des disques distincts
  - La réplication
  - La sauvegarde

PANNE DU DISQUE DE LA BASE

- Solution de base :
  - On réinstalle un disque pour la base.
  - On ré-exécute toutes les transactions du log.
- Inconvénients :
  - Phase de latence longue au moment du redémarrage.
  - Implique de conserver tous les log depuis l'origine.
- Avec sauvegarde
  - Une sauvegarde S_t est une copie de l'état de la base à un instant t
  - Récupérer S_t
  - Réappliquer le log depuis t

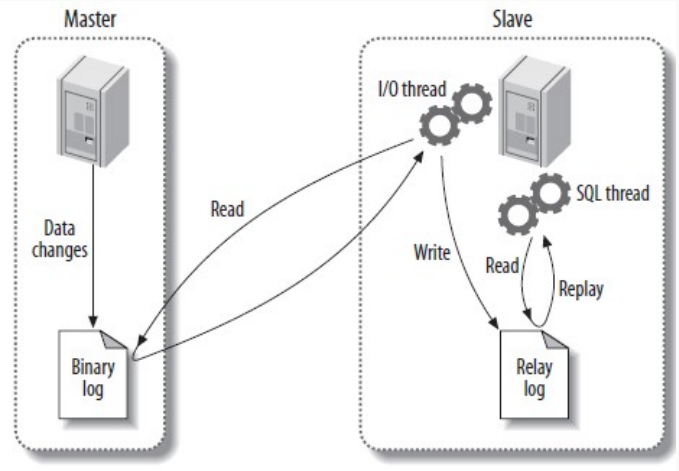


PANNE DU DISQUE DU LOG

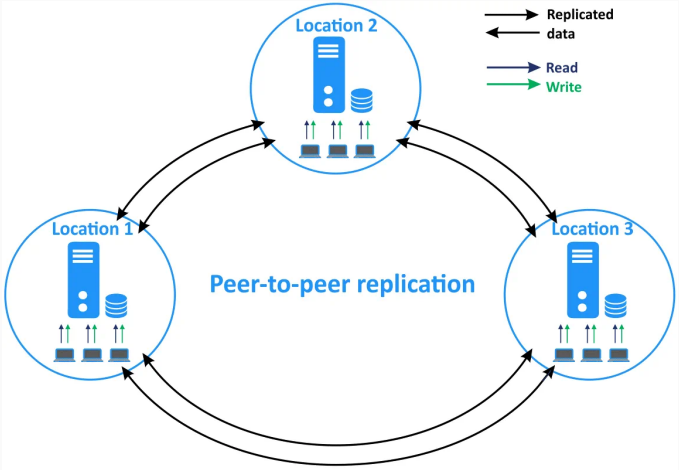
- Rappel : l'état de la base est dans les fichiers et le buffer
- On gère la panne en effectuant un checkpoint
  - on écrit les blocs modifiés sur le disque de la base,
  - on fait les réparations nécessaires et on redémarre.
- Assez fragile : quid en cas de panne électrique et panne du disque de log ?
- Réplication du log

RÉPLICATION

- Technique utilisée dans les systèmes distribués
- Plusieurs copies de la base
  - Facteur de réplication
  - Stratégies de réplication : Synchronisation des copies
    - Log-Based Incremental Replication
    - Key-Based Incremental Replication
    - Full Table Replication
    - Snapshot Replication
    - Transactional Replication
    - ...
- Architectures



Master-Slave



Peer-to-peer (Multi-master)

Questions?