Principes des systèmes de bases de données distribuées

M. Tamer Özsu
Patrick Valduriez

Grandes lignes

- Introduction
- Conception de bases de données distribuées et parallèles
- Contrôle des données distribuées
- Traitement distribué des requêtes
- Traitement distribué des transactions
- Réplication des données
- Intégration des bases de données - Systèmes multi-bases de données
- Systèmes de bases de données parallèles
- Gestion des données de pair à pair
- Traitement des données volumineuses
- NoSQL, NewSQL et Polystores
- Gestion des données sur le Web

Grandes lignes

- NoSQL, NewSQL et Polystores
  - Motivations
  - Systèmes NoSQL
  - Systèmes NewSQL
  - Polystores

Motivations

- Tendances
  - Big data
    - Données non structurées
  - Interconnexion des données
    - Hyperliens, balises, blogs, etc.
  - Très grande évolutivité
    - Taille des données, nombre d'utilisateurs
- Limites des SGBD relationnels (SQL)
  - Nécessité d'un DBA compétent et de schémas bien définis
  - SQL et tuning complexe
  - Difficile de rendre les mises à jour évolutives
    - Les SGBDR parallèles utilisent un disque partagé pour OLTP.
- Le théorème CAP

Le théorème CAP

- Sujet polémique
  - "Une base de données ne peut pas assurer la cohérence ET la disponibilité pendant une partition du réseau"
  - Argument utilisé par NoSQL pour justifier leur manque de propriétés ACID
  - Mais cela n'a rien à voir avec le passage à l'echelle
- Deux points de vue différents
  - Bases de données relationnelles
    - La cohérence est essentielle
      - Transactions ACID
  - Systèmes distribués
    - La disponibilité du service est essentielle
      - Incohérence tolérée par l'utilisateur, par exemple le cache du web.

Qu'est-ce que le théorème de la PAC ?

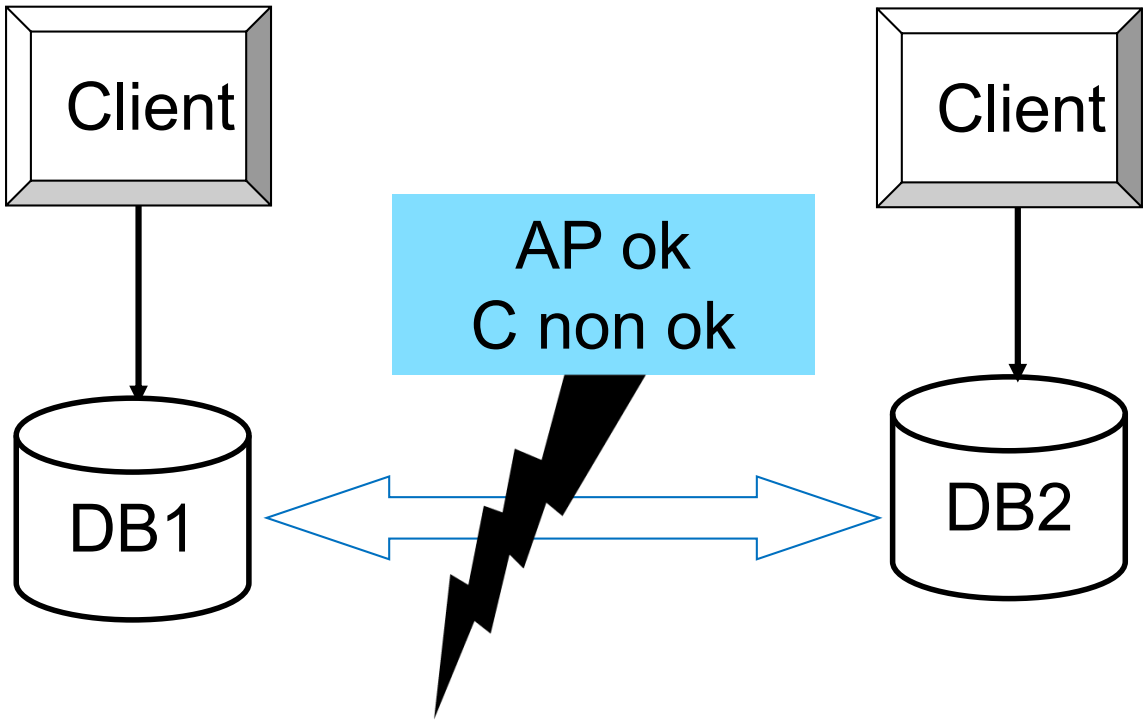
- Les propriétés souhaitables d'un système distribué
  - *Cohérence* : tous les nœuds voient les mêmes valeurs de données au même moment.
  - *Disponibilité* : toutes les demandes reçoivent une réponse
  - *Tolérance de partition* : le système continue de fonctionner en cas de défaillance du réseau.
- Histoire
  - Lors de la conférence PODC 2000, Brewer (UC Berkeley) conjecture qu'on ne peut avoir que deux propriétés en même temps
  - En 2002, Gilbert et Lynch (MIT) prouvent la conjecture, qui devient un théorème

Cohérence forte ou éventuelle

- Cohérence forte (ACID)
  - Tous les nœuds voient les mêmes valeurs de données en même temps.
- Cohérence éventuelle
  - Certains nœuds peuvent voir différentes valeurs de données en même temps.
  - Mais si nous arrêtons d'injecter des mises à jour, le système atteint une forte cohérence

- Illustration avec la réplication symétrique et asynchrone dans les bases de données

Réplication symétrique et asynchrone



Mais nous avons une cohérence éventuelle

- Après la reconnexion (et la résolution des conflits de mise à jour), la cohérence peut être obtenue

Grandes lignes

- NoSQL, NewSQL et Polystores
  - Motivations
  - Systèmes NoSQL
  - Systèmes NewSQL
  - Polystores

NoSQL (Not Only SQL) : définition

- SGBD spécifique : pour les données basées sur le web
  - Modèle de données spécialisé
    - Clé-valeur, tableau, document, graphique
  - Propriétés des SGBD relationnels commerciaux
    - Full SQL, transactions ACID, indépendance des données
  - Pour
    - Simplicité (schéma, API de base)
    - Évolutivité et performance
    - Flexibilité pour le programmeur (intégration avec le langage de programmation)

- NB : SQL n'est qu'un langage et n'a rien à voir avec le débat.

Approches NoSQL

- Caractérisé par le modèle de données, par ordre croissant de complexité :
  1. Clé-valeur : DynamoDB
  2. Tabulaire : Bigtable
  3. Document : MongoDB
  4. Graphique : Neo4J
  5. Multimodèle : OrientDB
- Qu'en est-il du SGBD objet ou du SGBD XML ?
  - Il existait déjà bien avant NoSQL
  - Parfois présenté comme NoSQL
  - Mais pas vraiment évolutif

Stockage des clés et des valeurs

- Modèle de données simple (clé, valeur)
  - Clé = identifiant unique
  - Valeur = un texte, une donnée binaire, une donnée structurée, etc.
- Requêtes simples
  - Put (clé, valeur)
    - Insère une paire (clé, valeur)
  - Valeur = get (key)
    - Retourne la valeur associée à la clé
  - {(clé, valeur)} = get_range (clé1, clé2)
    - Retourne les données dont la clé est dans l'intervalle [clé1, clé2].

Amazon DynamoDB

amazon web services

- Service principal d'AWS pour le stockage de données
  - Par exemple, les listes de produits, les paniers d'achat, les préférences des utilisateurs.
- Modèle de données (clé, valeur structurée)
  - Partitionnement sur la clé et indexes secondaires sur les attributs
  - Requêtes simples sur la clé et les attributs
  - Flexible : aucun schéma à définir (mais inférence automatique)
- Cohérence
  - Lectures cohérentes éventuelles (par défaut)
  - Des lectures cohérentes (Strong consistency)
  - Mises à jour atomiques avec compteurs atomiques
- Haute disponibilité et tolérance aux pannes
  - Réplication synchrone entre les centres de données
- Intégration avec d'autres services AWS
  - Contrôle d'identité et accès
  - MapReduce
  - Entrepôt de données Redshift

DynamoDB - modèle de données

DynamoDB

amazon web services

- Tableau (éléments)
- Elément (clé, attributs)
  - 2 types de clés primaires (uniques)
    - Hash (1 attribut)
    - Hash & range (2 attributs)
  - Attributs de la forme "name" : "value" (nom)
    - Type de valeur : scalaire, ensemble, ou JSON
- API Java avec méthodes
  - Ajouter, mettre à jour, supprimer un élément
  - GetItem : renvoie un élément par clé primaire dans une table
  - BatchGetItem : renvoie les éléments de la même clé primaire dans plusieurs tables
  - Scan : renvoie tous les éléments
  - Requête
    - Intervalle sur la clé de et l'intervalle de hachage
    - Accès sur l'attribut indexé

Table: Forum_Thread

Forum	Subject	Date of last post	Tags
"S3"	"abc"	"2017 . . . "	"a" "b"
"S3"	"acd"	"2017 . . . "	"c"
"S3"	"cbd"	"2017 . . . "	"d" "e"
"RDS"	"xyz"	"2017 . . . "	"f"
"EC2"	"abc"	"2017 . . . "	"a" "e"
"EC2"	"xyz"	"2017 . . . "	"f"

Hash key — Range key

GetItem (Forum="EC2", Subject="xyz")

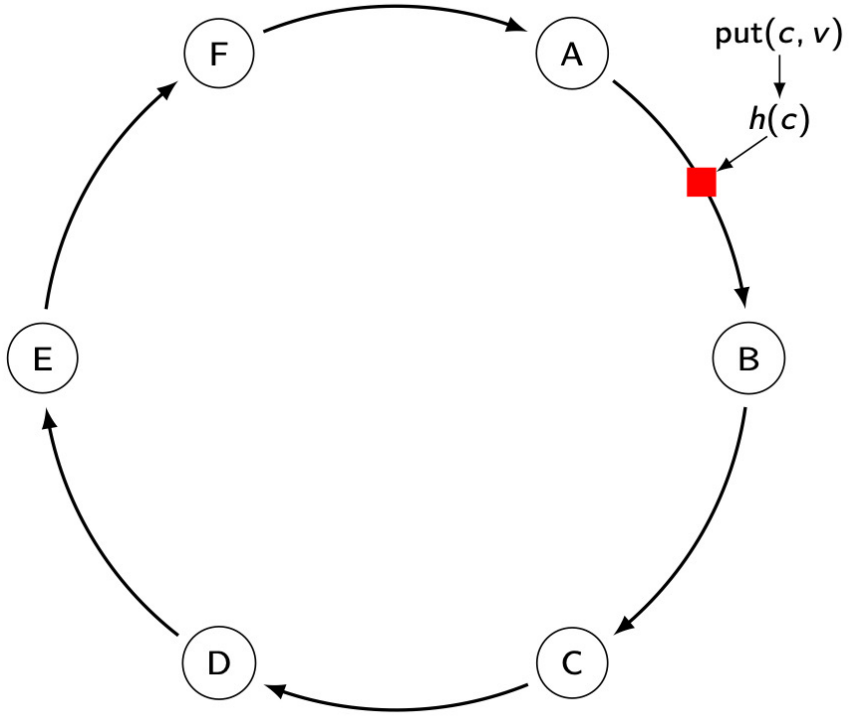
Requête (Forum="S3", Sujet > "ac")

DynamoDB - partitionnement des données

DynamoDB

amazon web services

- Hachage cohérent : l'intervalle des valeurs de hachage est traité comme un anneau.
- Avantage : si un nœud tombe en panne, son successeur reprend ses données.
  - Aucun impact sur les autres nœuds
- Les données sont répliquées sur les nœuds suivants



Le nœud B est responsable de l'intervalle de valeurs de hachage (A,B]. Ainsi, l'élément (c,v) est attribué au noeud B