GESTION DE DONNÉES À LARGE ÉCHELLE

Hubert Naacke

Decembre 2011

Plan

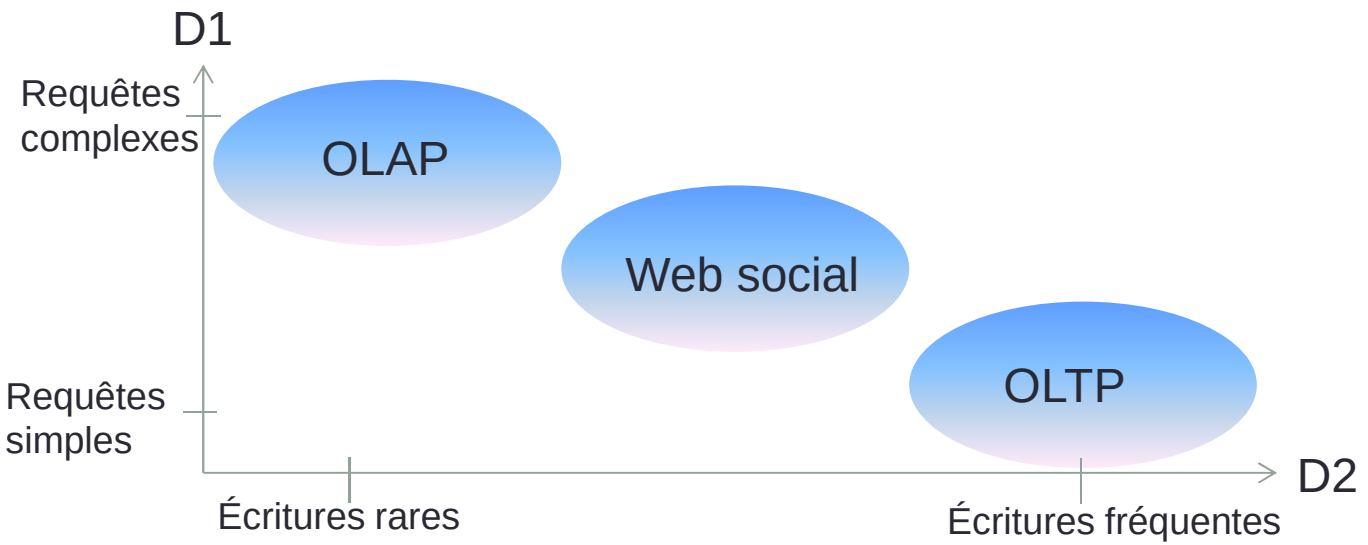
- Gestion de données à large échelle.
  - Introduction
  - Règles de conception
  - Principe de la réplication
  - Etude de cas de Facebook
- Gestion de données dans une infrastructure cloud.
  - Architecture d'un cloud, service de stockage de données,
  - Requêtes, modification de données, cohérence des données
  - (à venir) Etude de cas : Google Megastore, DB Shards

Contexte Applicatif

- Type d'application
  - Application web gérant des masses de données (TO, POctets)
  - Application Web2.0, centrée sur l'utilisateur
    - Gestion du profil de l'utilisateur et de ses contacts
    - Interactions entre utilisateurs : partage, discussion, ...
- Croissance des données
  - La quantité de données augmente
    - Exple : La taille d'une table double tous les 6 mois.
  - Le schéma des données devient plus complexe
- Croissance des traitements
  - Requête d'analyse plus complexe
  - Requêtes plus nombreuses (demande croissante)

Classification des applications

- Classification suivant deux dimensions
- D1: Complexité d'une opération
- D2: Taux d'écriture
  - Nombre d'écritures / (Nombre de lectures et d'écritures)
- Types d'application
  - OLAP: appli décisionnelle
  - OLTP: appli transactionnelle
  - Appli Web 2.0 sociale

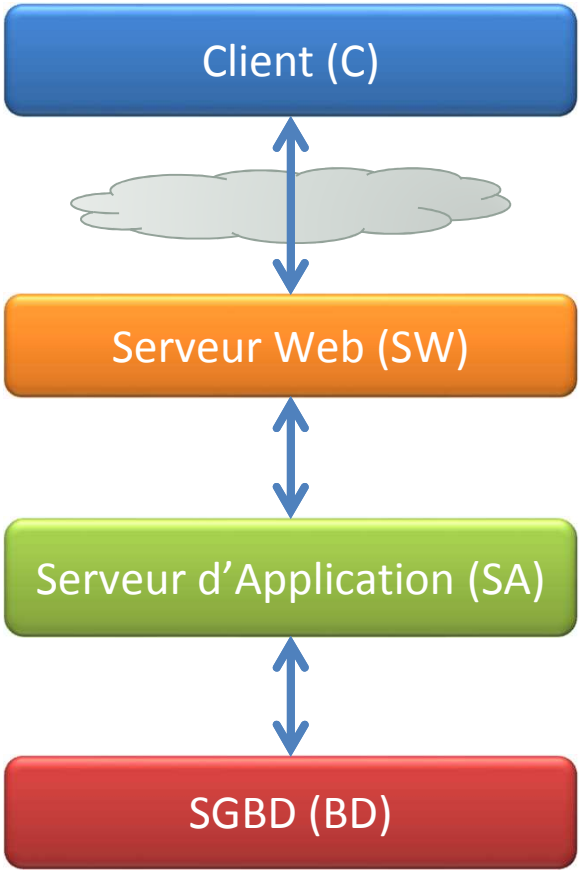


Les applications transactionnelles

- Caractéristiques
  - Le schéma des données est simple.
    - Exple: schéma relationnel TPC (moins de 10 tables)
  - Chaque opération est simple
    - Lit et écrit peu de données
    - La complexité d'une demande (opération) reste constante
- Passage à l' échelle des données
  - La quantité totale de données augmente.
    - Exple: La cardinalité d'une table augmente
- Passage à l'échelle des traitements
  - Le nombre de demandes à traiter augmente.

Architecture d'une application web

- Décomposée en couches

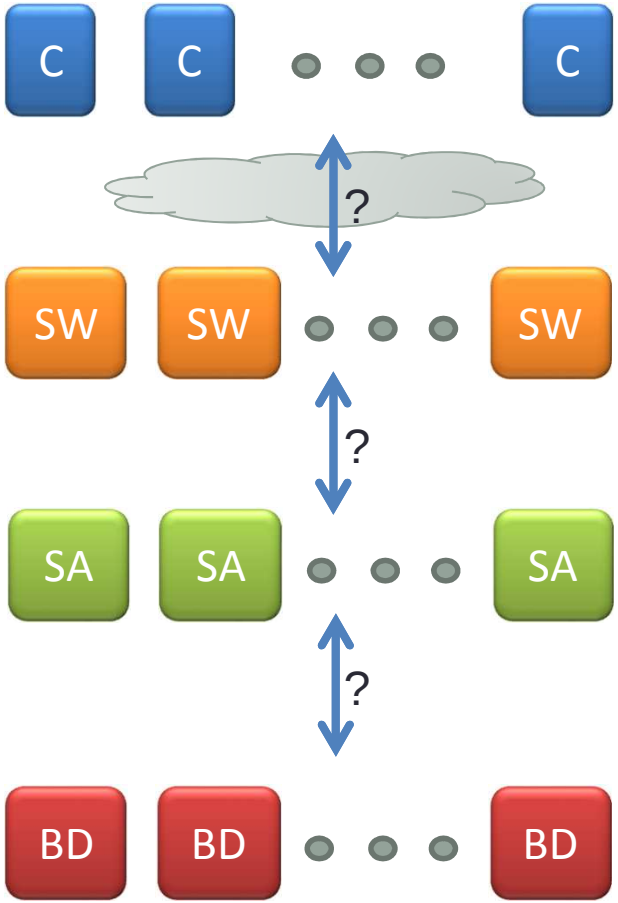


Exigences applicatives

- Qualité de service offerte
  - Disponibilité
    - Répondre à une requête en un temps constant (inférieur à un seuil)
  - Durabilité
    - Pérennité des données à long terme, tolérer les pannes
  - Cohérence
    - Pas de contradiction entre les requêtes
      - Requêtes successives d'un même utilisateur
      - Requêtes simultanées de plusieurs utilisateurs
- Défi du passage à l'échelle
  - Comment offrir une qualité de service constante lorsque les données et la charge augmentent ?

Vers une architecture répartie

- Objectif: décentraliser chaque couche



Problème lié à l'échelle

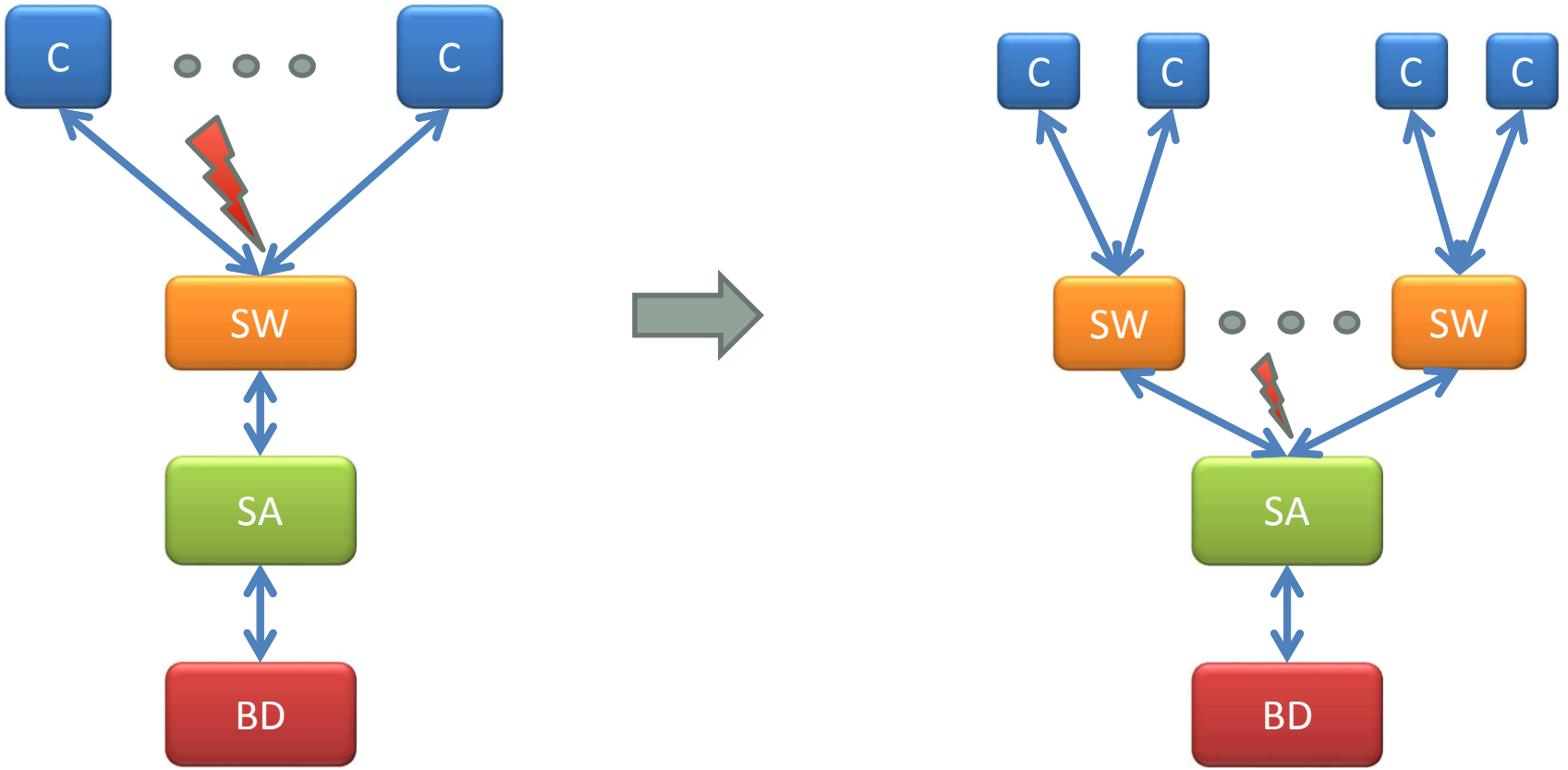
- Système à large échelle
  - Composé de nombreuses ressources interconnectées
  - Communication : échange de messages entre les ressources
  - Probabilité élevée de panne de communication : perte de message
- Tolérer une panne de communication
  - Est une nécessité
- Disponibilité
  - Il existe toujours une ressource prête à répondre à une requête
    - Redondance, réplication
- Cohérence des données
  - Accéder à toutes les répliques d'une donnée, pour les contrôler

- Incompatibilité: Disponibilité/Cohérence
  - Disponibilité: répondre sans connaître toutes les répliques
  - Cohérence: répondre seulement si on connaît toutes les répliques
  - Cf: théoreme CAP : Consistent, Available, tolerate network Partition (panne du réseau)

- Deux solutions envisageables
  - Disponible mais pas entièrement cohérent
  - Entièrement cohérent mais temporairement indisponible
  - Rmq, les deux solutions tiennent compte des éventuelles pannes du réseau

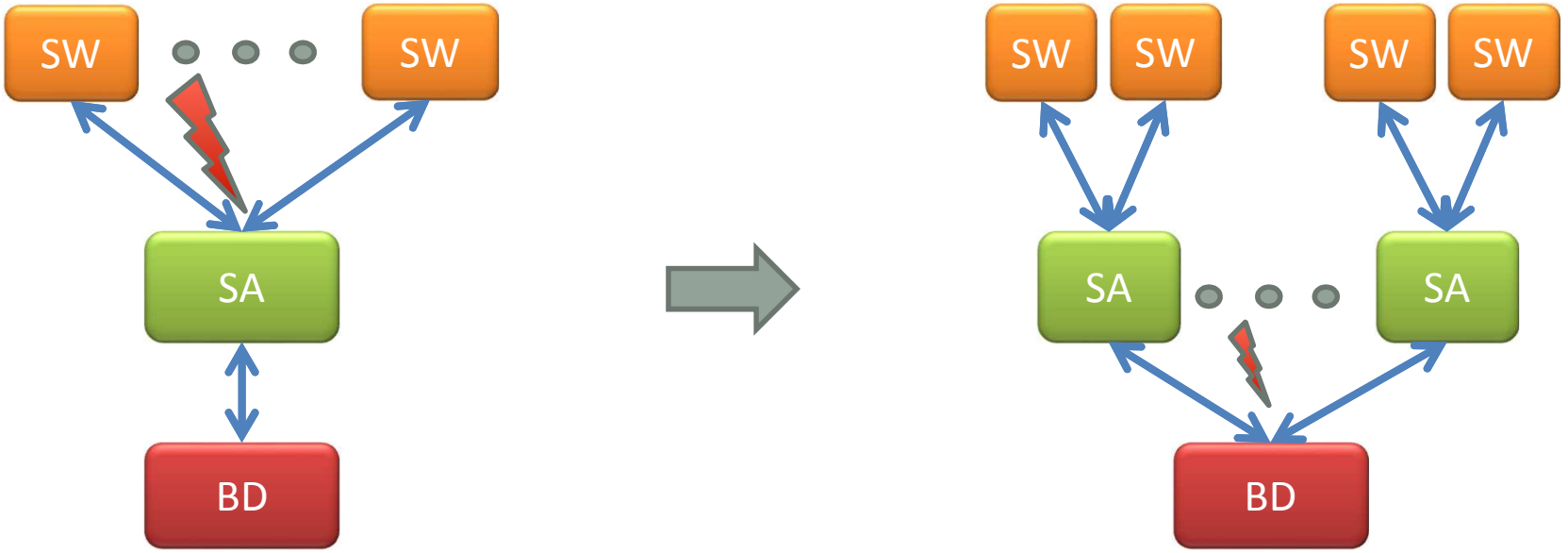
Architecture web décentralisée (1)

- Décentraliser la couche Serveur Web
  - gestion des requêtes http
  - Equilibrage de charge



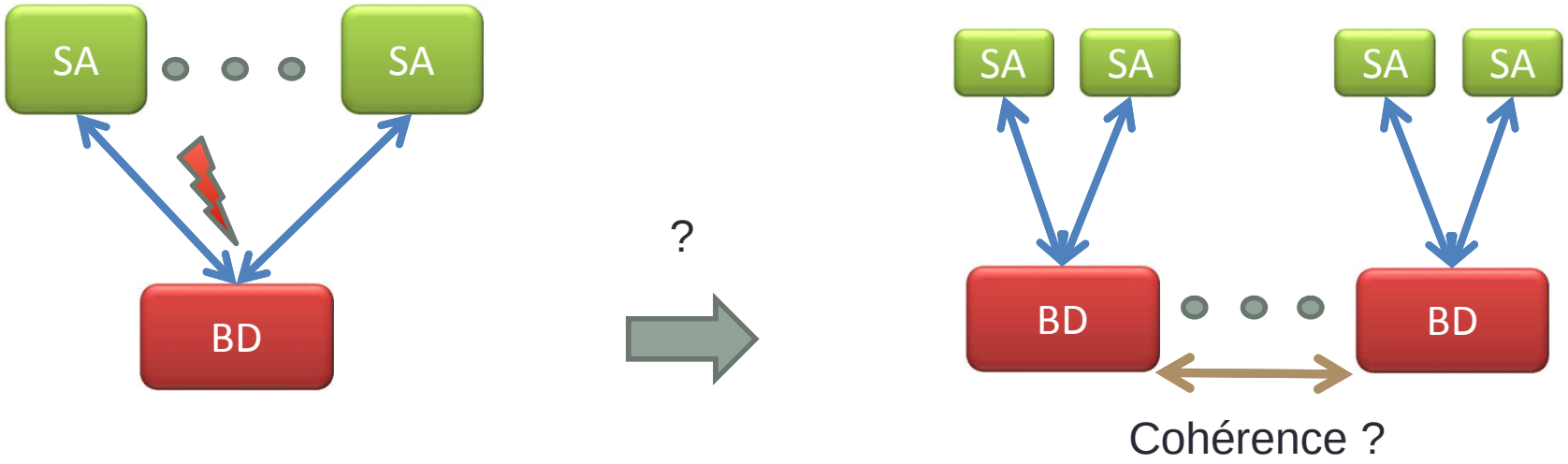
Architecture web décentralisée (2)

- Décentraliser le SA
  - Distribuer la logique applicative, fragmentation fonctionnelle
  - Cloner les fonctions sans état (stateless)

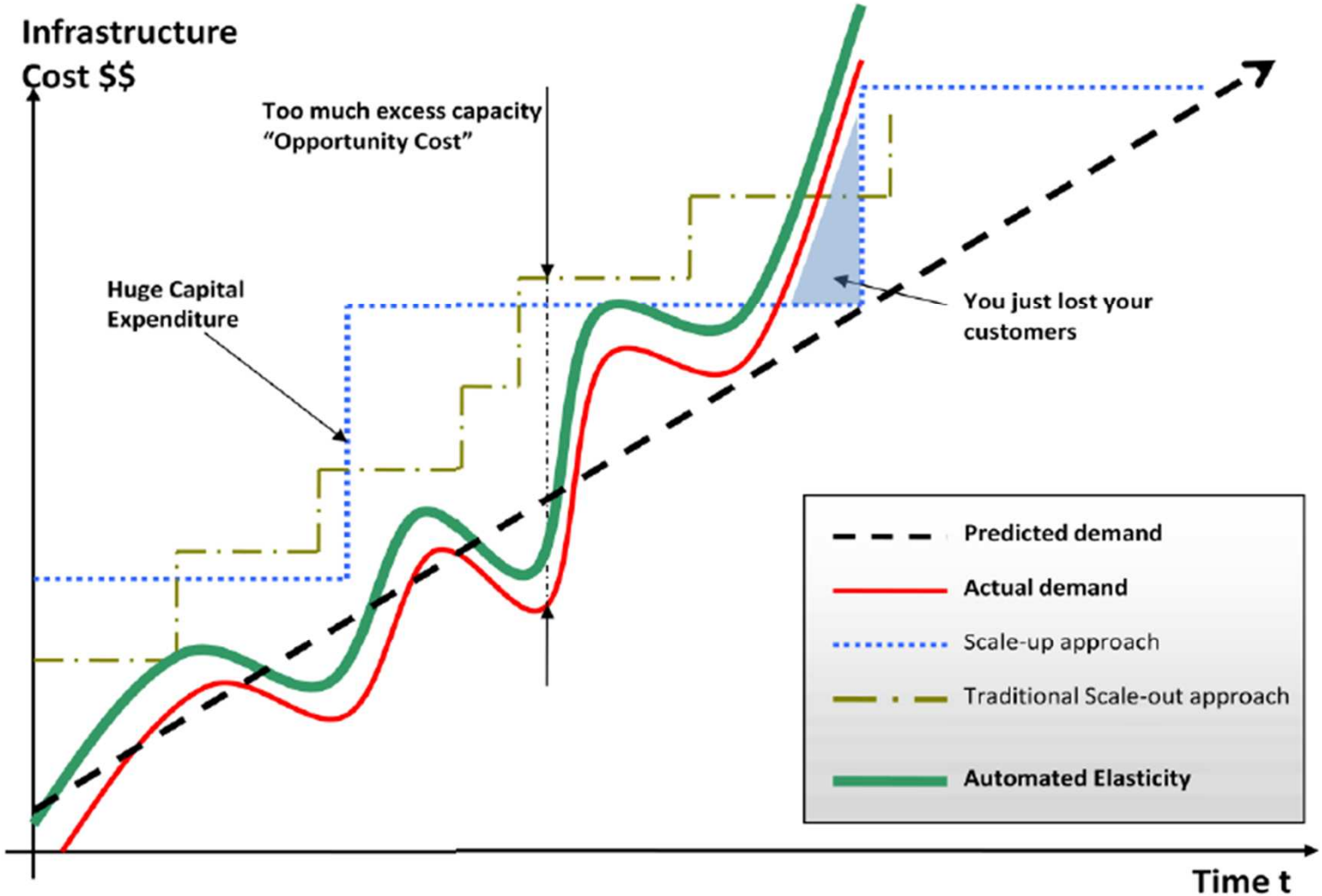


Architecture web décentralisée (3)

- Décentraliser le SGBD ?
  - Fragmenter les données ? Répliquer les données ?
  - Garantir la cohérence des données ?



Elasticité: Exemple d'Amazon AWS



Automated Elasticity + Scalability

Bibliographie

- Ten Rules for Scalable Performance in “Simple Operation”
- Datastores
  - By Michael Stonebraker and Rick Cattell, CACM 2011
- Data Management Challenges in Cloud Computing Infrastructures
  - D. Agrawal, Amr El Abbadi, S. Antony, S. Das, DNIS 2010
- Data Management in the Cloud
  - Amr El Abbadi, keynote BDA 2011