Structure de données dans le système noSQL

Presented by: Doussiet Lou, Vidor Fabien, Corin Gaetan

Sommaire

1. Introduction

2. Comparaison de 3 moteurs

3. Choix du moteur sélectionné

4. Modèle CAP

5. Architecture du moteur selectionné

6. Contexte de sauvegarde

7. Gestion de la sauvegarde

8. Sauvegarde Local vs Externalisé

9. Démarrage consistant et automatique

Introduction

- Entreprise IADATA est une entreprise qui souhaite évoluer son système d'information en comparant différents systèmes qui répondent le mieux à leur problématique

- Elle nous a mandaté pour que, en tant que expert technique, nous puissions adapté le SI à leurs problématique métier et leurs attente en disponibilité, cohérence et besoin en partition

Comparaison des 3 moteurs

Neo4j	Cassandra	MongoDB
+ Fait en langage Java, rapidité de requête sur relations complexes + Langage de requête très performant, optimisé pour requête les plus courantes + Modularité dans la hiérarchisation des données - Mauvaise maturité pour export massif de données - Difficulté de sauvegarde et restauration dans un contexte d'automatisation	+ Disponibilité continu, même en cas de panne + Optimisé pour la surveillance des données grâce à des métriques et logs pertinents + Ajout de noeuds à chaud parfaitement scalable - Peu adapté aux relations complexes - Peu adapté aux évolutions fréquentes du schémas	+ Modèle Maître-esclave performant face aux pannes et répartition des charges + Système orienté document, aide à la modularité des éléments à intégrer dans la BDD + La scalabilité horizontale permet de gérer un grand nombre de données sur plusieurs machines avec une performance remarquable - Mal adapté aux relations complexes - Le modèle favorise la consistance au détriment de la disponibilité immédiate

Choix du moteur

MongoDB :

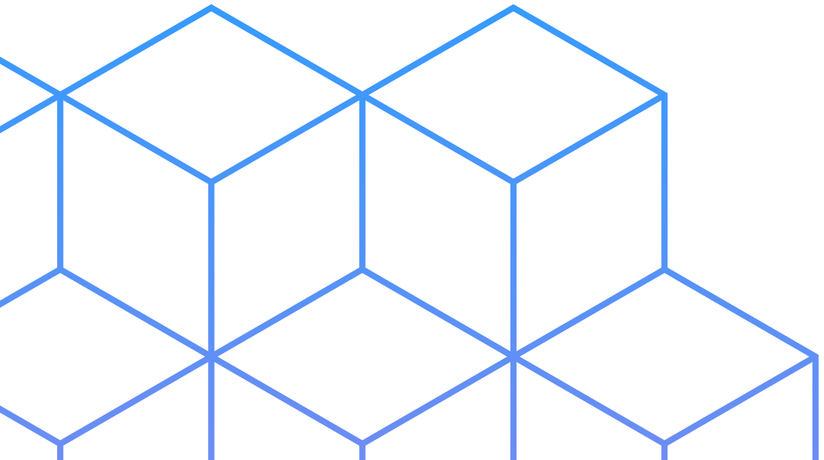
- Exportation/restauration de données sans perte de performance
- Adapté à des évolutions futures des données
- Import facile dans d’autres environnements (test workflow)

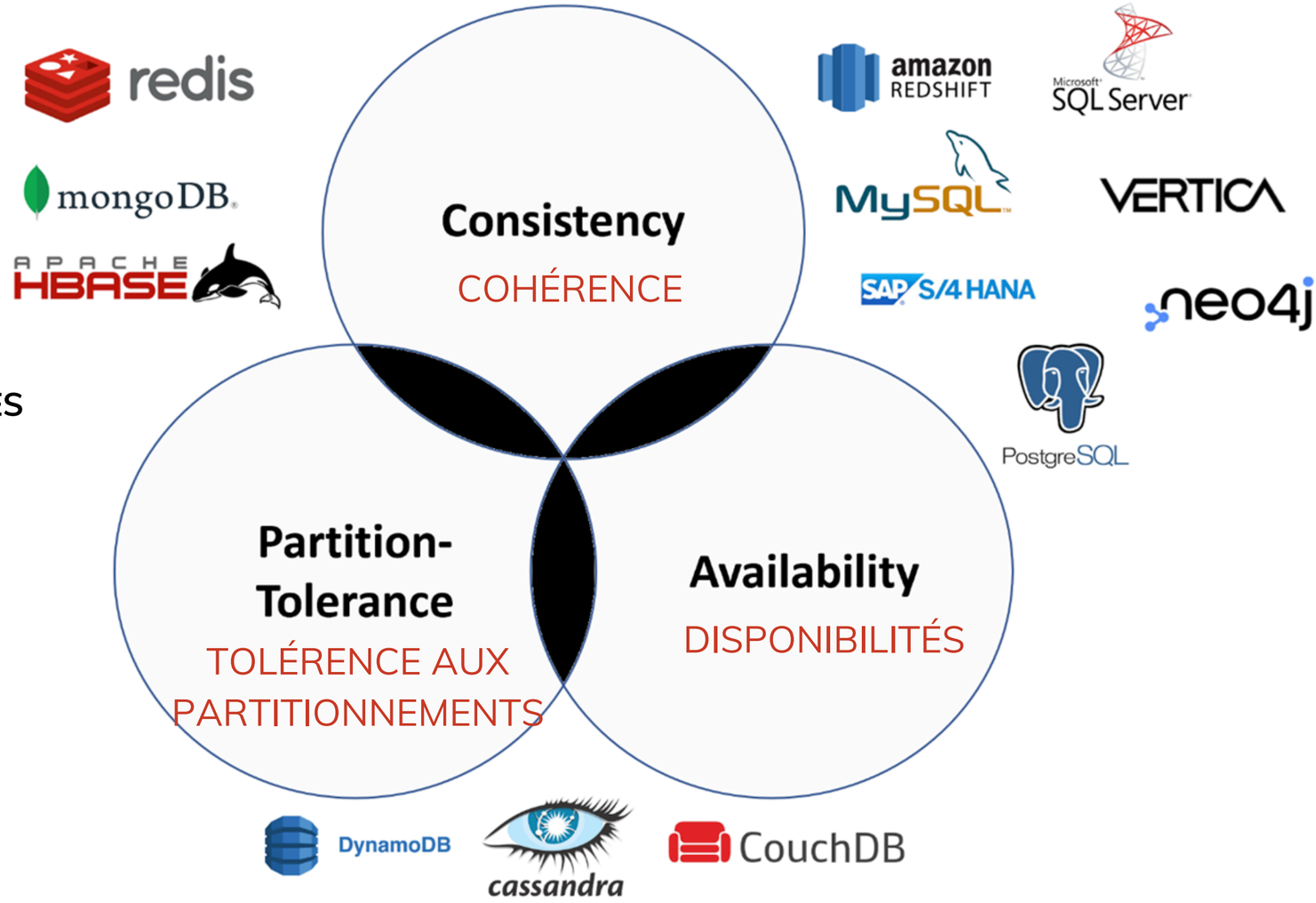
Modèle CAP

MongoDB est **DISPONIBLE**
(toutes les requêtes sont répondu, peut importe la situation)

MongoDB est **TOLÉRENT AUX PARTITIONS**
(même lors de dissociation des données entre les nœuds,
mongoDB continue a offrir la possibilités de lire et écrire)

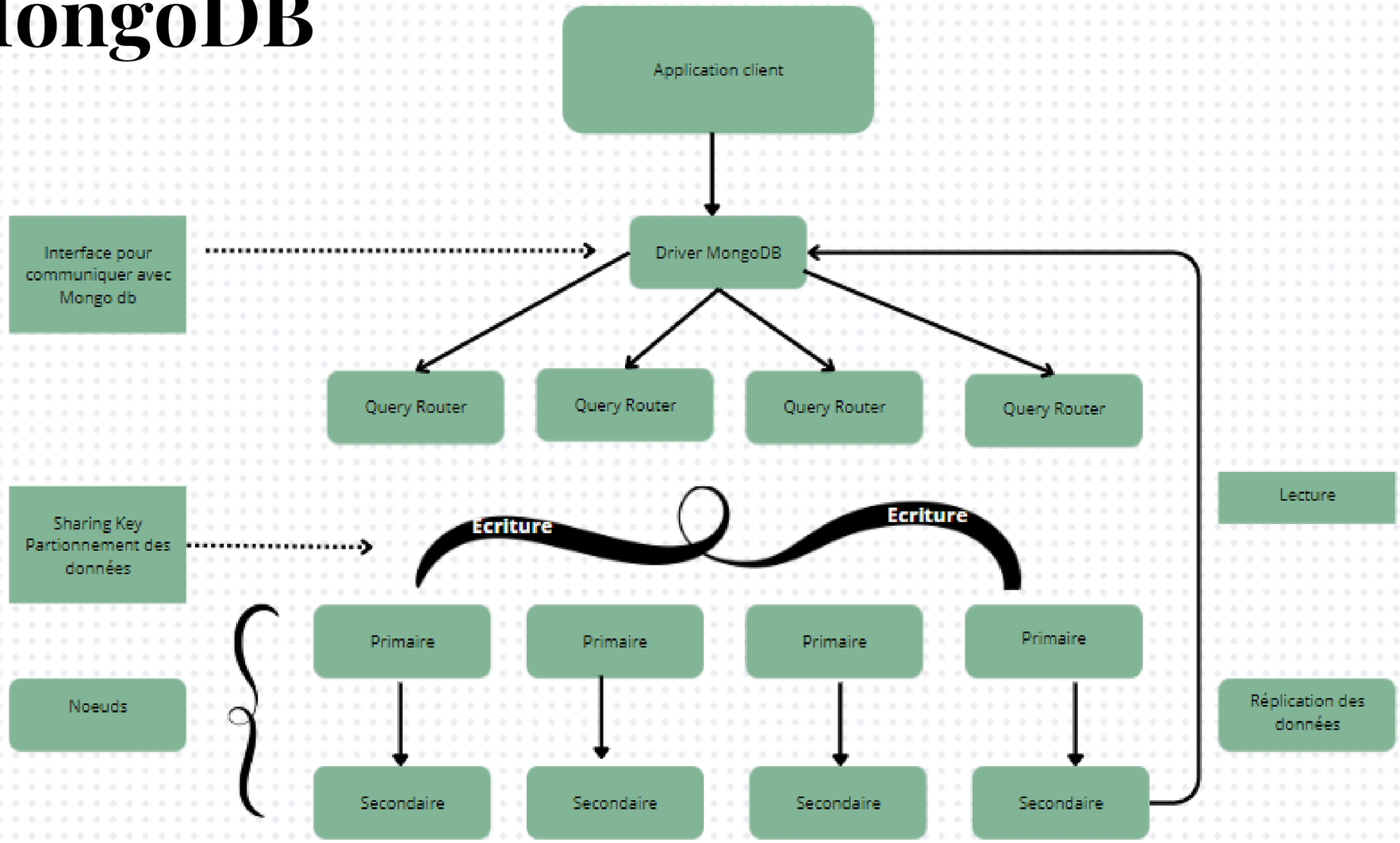
Cependant, MongoDB a un coût en **COHÉRENCE DES DONNÉES**
(lorsqu'une partition des données se produit,
les données, qui reste disponible, ne sont pas cohérentes
jusqu'à ce que la partition soit résolue)





Architecture du moteur sélectionné

MongoDB

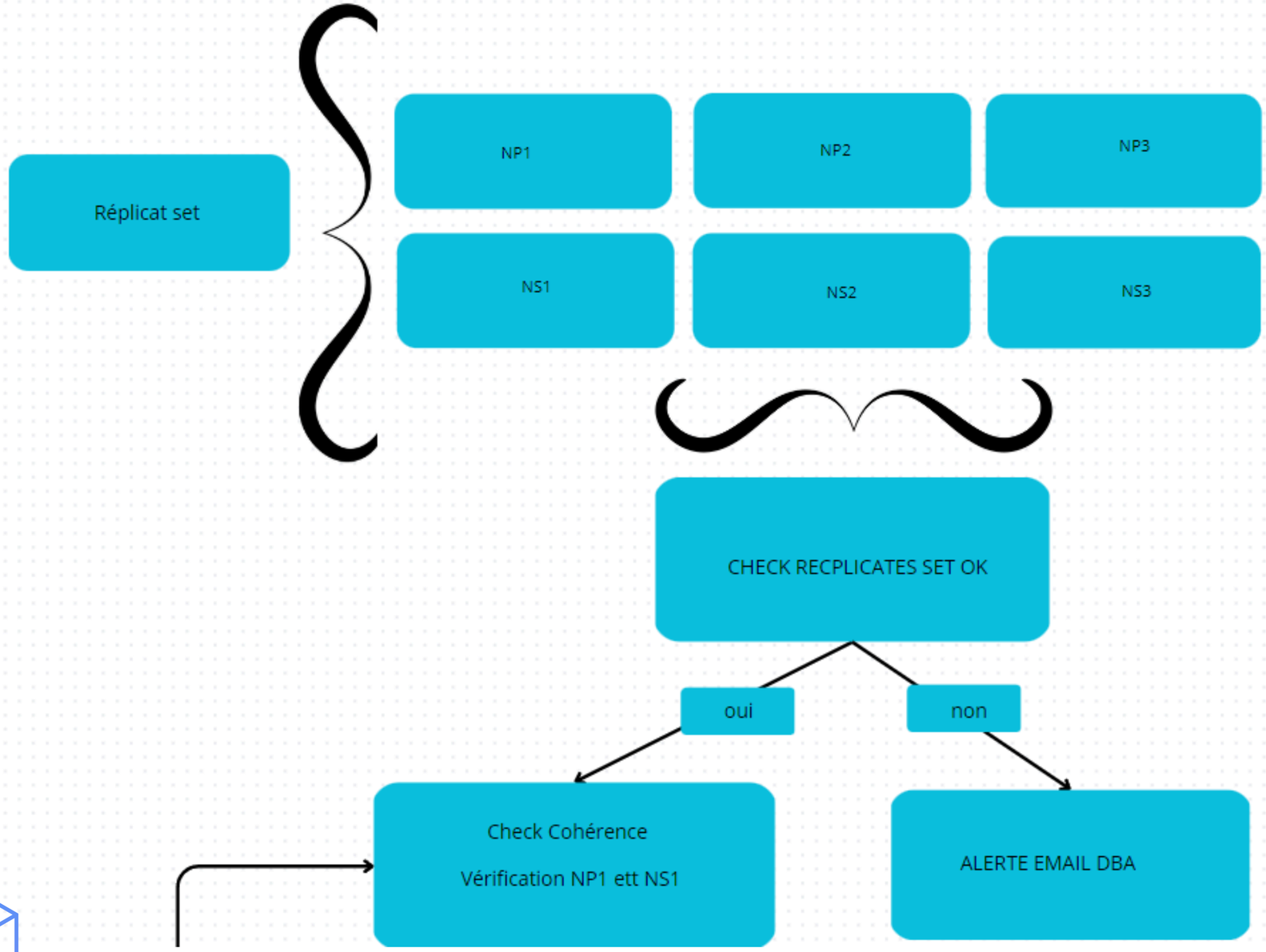


Contexte de sauvegarde

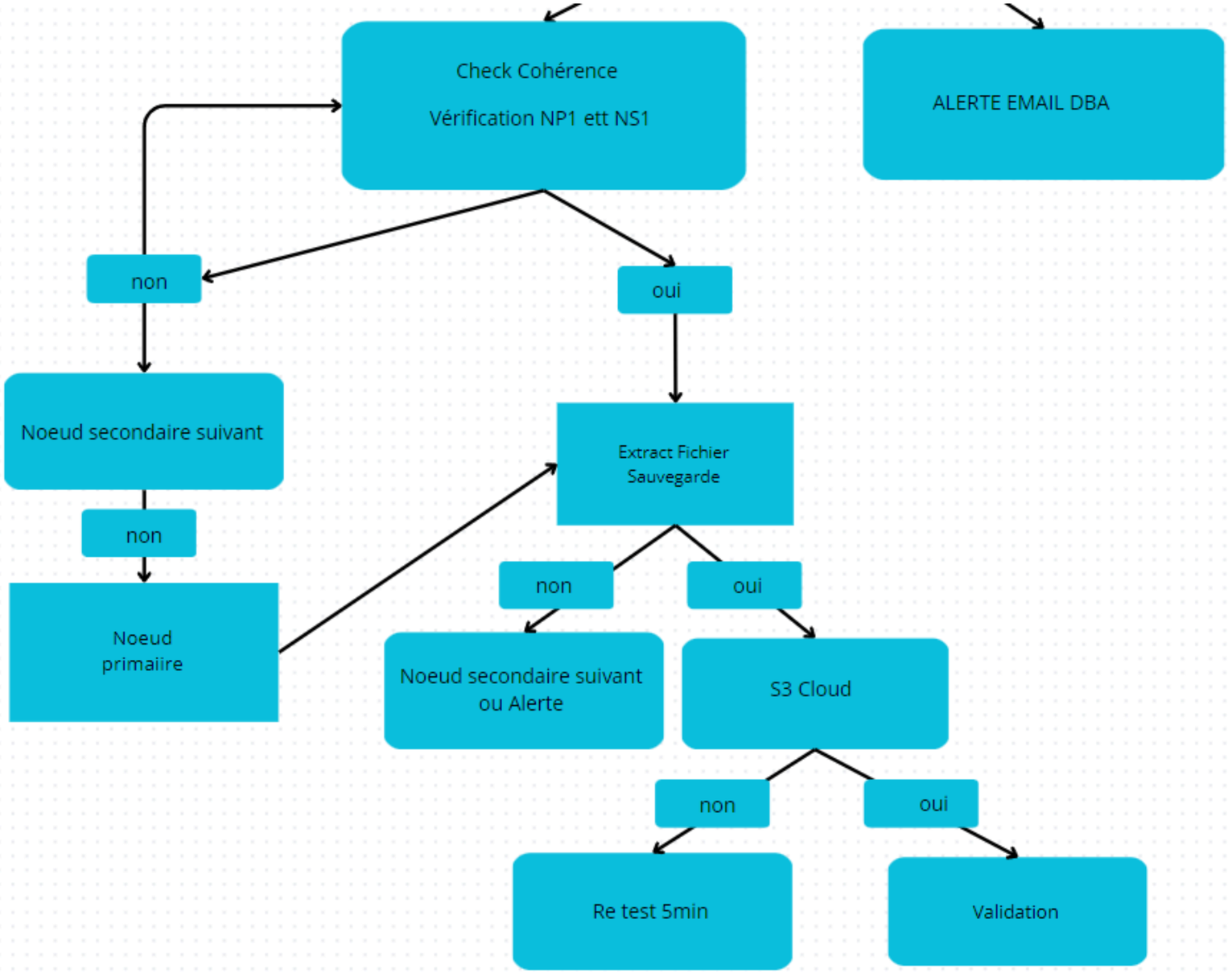
Demande client

- Traitement critique de facturation chaque nuit
- Restauration fiable et rapide
- Les données altérés doivent être copiés pour analyses
- Copie cohérentes des données de manière quotidienne
- Cluster de pré-production qui doit pouvoir être restauré grâce a la sauvegarde quotidienne

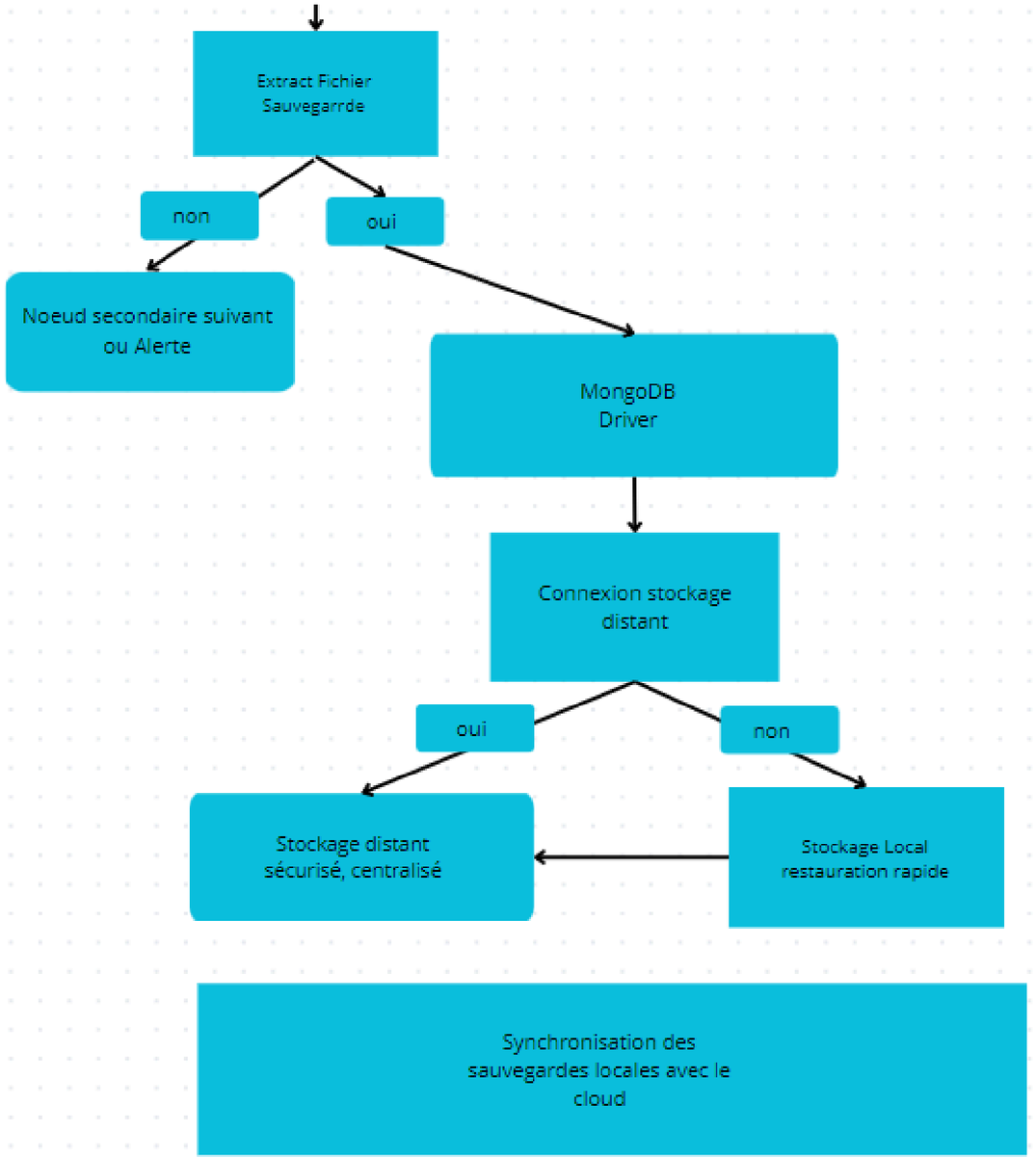
Gestion de la sauvegarde



Gestion de la sauvegarde



Sauvegarde Locale vs Externalisée



Démarrage consistant et automatique

01 VERIFICATION DE L'INTEGRITER ENTRE LES NOEUDS

02 SYNCHRONISATION DES NOEUDS

03 VERIFICATION QUE ⅔ NOEUDS SOIENT OPERATIONNELS

04 ECRITURE CONFIRMÉE PAR LA MAJORITÉ

05 LECTURE CONFIRMÉE PAR LA MAJORITÉ

06 SYNCHRONISATION POUR ENVOI