Rapport d'Analyse

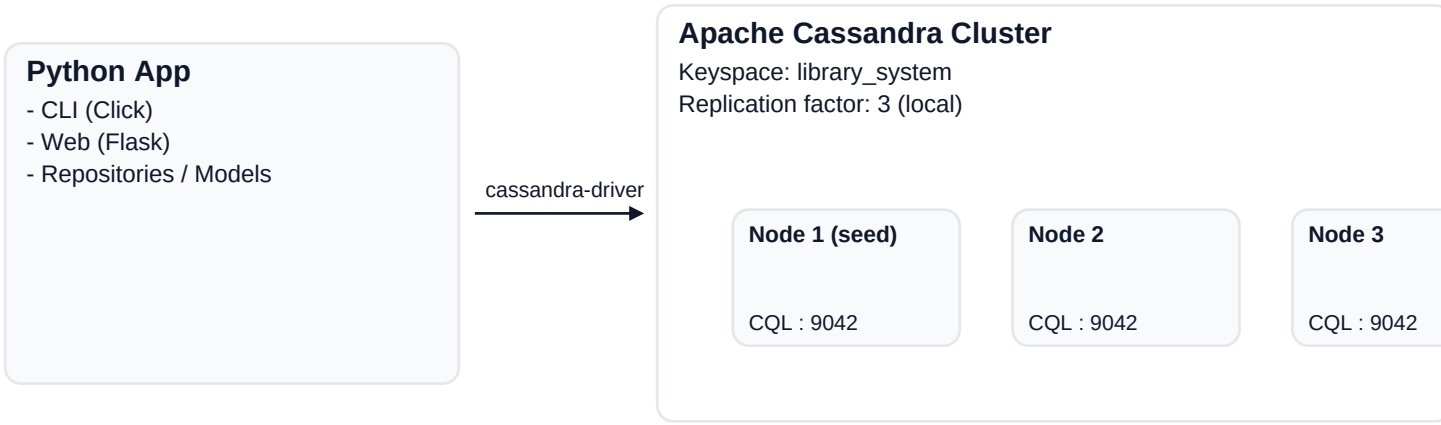
Repository Git :
https://github.com/noewisselmann/library_system_Noe_WISSELMANN.git

Systeme de Gestion de Bibliotheque Numerique (Python + Apache Cassandra)

Objectif	Documenter la modelisation Cassandra, justifier les cles (partition/clustering), expliquer les co
Perimetre	Livres, utilisateurs, emprunts, emprunts actifs, historiques et tables de navigation (denormalis
Source de verite	schema.cql: trouve

Architecture

Architecture - Application Python + Cluster Cassandra



Note: si le fichier schema/schema.cql est absent, le rapport indiquera 'Non detecte' au lieu d'inventer des cles.

1. Modelisation Cassandra (oriente requetes)

Cassandra impose une modelisation par patterns de requetes: une table par besoin de lecture/tri, avec duplication volontaire des donnees pour eviter joins et scans.

Tables detectees dans le schema

Table	Partition key(s)	Clustering key(s)
books_by_id	Non détecté (schema.cql manquant ?)	-
borrows_by_user	Non détecté (schema.cql manquant ?)	-
users_by_id	Non détecté (schema.cql manquant ?)	-

Raisons principales de la denormalisation

- Lecture rapide par cle de partition (pas de scans).
- Ordonner les resultats via clustering keys (historique par date, etc.).
- Eviter ALLOW FILTERING et les index secondaires systematiques.

2. Justification des Partition Keys et Clustering Keys

Cette section est generee a partir de schema.cql. Pour chaque table, on rappelle la cle primaire et on explique l'impact sur distribution et tri dans Cassandra.

Table: books_by_id

Primary key (brut): Non detecte

Partition key(s)	Non detecte
Clustering key(s)	-
But principal	A completer: decris ici le query pattern principal de la table (lecture/tri).

Table: borrows_by_user

Primary key (brut): Non detecte

Partition key(s)	Non detecte
Clustering key(s)	-
But principal	A completer: decris ici le query pattern principal de la table (lecture/tri).

Table: users_by_id

Primary key (brut): Non detecte

Partition key(s)	Non detecte
Clustering key(s)	-
But principal	A completer: decris ici le query pattern principal de la table (lecture/tri).

A completer: remplace les lignes 'But principal' par tes explications exactes (ex: 'recherche par ISBN', 'historique d un utilisateur trie par date', etc.). Le PDF est genere proprement, mais il ne peut pas deviner tes intentions si elles ne figurent pas explicitement dans le schema ou la doc.

3. Coherence vs Disponibilite (CAP) et niveaux de consistency

Cassandra privilegie la disponibilite et la tolerance aux partitions. La coherence est ajustable par requete via les niveaux (ONE, QUORUM, ALL).

Points a presenter pendant la demo

- Lecture/criture en LOCAL (cluster 3 noeuds) : on peut illustrer la haute dispo en arretaient un noeud.
- QUORUM: compromis classique (majorite) pour limiter les lectures incoherentes.
- ONE: latence minimale mais risque plus eleve de lire une donnee non encore repliquee.
- ALL: coherence maximale mais indisponibilite si un noeud est down.

Recommandation (a adapter)

Pour une bibliotheque: ecritures en QUORUM et lectures en QUORUM/ONE selon criticite (ex: disponibilite d exemplaires peut demander QUORUM).

4. Comparaison avec une approche SQL

SQL normalise les donnees et repose sur joins et transactions ACID. Cassandra evite joins/subqueries et accepte la duplication pour obtenir des lectures predecibles a grande echelle.

Sujet	SQL (relationnel)	Cassandra (NoSQL distribue)
Modelisation	Normalisation, joins	Une table par requete, denormalisation
Scalabilite	Scale-up, sharding complexe	Scale-out horizontal natif
Transactions	ACID (souvent)	Pas ACID global, coherence ajustable
Requetes	Flexible (WHERE, joins)	Doit fournir partition key pour perf optimale

5. Tests de performance (optionnel)

Cette section est un gabarit: tu peux ajouter des mesures (latence moyenne, throughput) en lecture/criture via un script de benchmark.

Idees de mesures simples

- Insert de N livres / N utilisateurs (prepared statements).
- Lecture par ISBN (books_by_id).
- Historique des emprunts d un utilisateur (borrows_by_user).
- Arret d un noeud et verification que l app reste utilisable (selon consistency).

Astuce: evite SELECT COUNT(*) sans partition key en demo, Cassandra affichera un warning (et ce n est pas un pattern de prod).