**Progetto di E-commerce di Libri con Database NoSQL**

**1. Introduzione e Descrizione del Problema**

**Presentazione del Progetto**

Nel contesto dell'evoluzione dell'e-commerce e della gestione dei dati, l'adozione di database NoSQL offre molteplici vantaggi, soprattutto per applicazioni con necessità di scalabilità, flessibilità e alta disponibilità. Abbiamo deciso di proporre un progetto di e-commerce specializzato nella vendita di libri, utilizzando una base di dati poliglotta composta da Riak e MongoDB.

**Obiettivi Principali**

Il progetto "Libreria-NOSQL" si propone di realizzare un sistema di gestione per un sito e-commerce di libri utilizzando database NoSQL. Le tecnologie principali coinvolte sono MongoDB per la gestione degli utenti, dei libri e degli ordini, e Riak per implementare una funzionalità specifica di raccomandazione dei libri agli utenti in base ai loro acquisti. L'obiettivo è garantire un sistema scalabile, resiliente e ad alta disponibilità, capace di gestire grandi volumi di dati e di offrire un'esperienza utente solida.  
**Analisi del Problema**

Identificazione delle esigenze di scalabilità e flessibilità nella gestione dei dati di un e-commerce, compresa la gestione di un ampio inventario

**Requisiti Funzionali e Non Funzionali**

**Requisiti Funzionali**

* Gestione dell'inventario dei libri (sono stati ovviamente dati per scontato le funzionalità di inserimento ed eliminazione di un libro in quanto ragioniamo in un ottica scalabile e per il n ostro progetto non è influente il caricamento/eliminazione o modifica di un singolo libro , che se pur estremamente semplice da realizzare non è il focus del progetto)
* Gestione dello storico degli acquisti
* Sistema di registrazione e autenticazione utenti mediante le sessioni
* Sistema di raccomandazione di libri basato sugli acquisti precedenti

**Requisiti Non Funzionali**

* Scalabilità
* Alta disponibilità
* Flessibilità nella gestione dei dati

**2. Analisi dei Requisiti**

**3. Scelta della Tecnologia di Database**

**Stack Tecnologico Scelto**

* **MongoDB**: per la gestione dell'inventario dei libri dei profili utenti e dei loro ordini
* **Riak**: per la gestione del sistema di raccomandazione di libri basata sugli acquisti degli utenti

**Motivazione delle Scelte Tecnologiche**

* **MongoDB**: scelto per la sua flessibilità nella gestione di dati strutturati e non strutturati e le sue potenti capacità di query.
* **Riak**: scelto per la sua capacità di gestione di dati distribuiti e la sua alta disponibilità, ideale per il sistema di raccomandazione e per la gestione delle sessioni utente.

**Dettagli di Utilizzo**

**Riak**

* Organizzato utilizzando i bucket per le regioni, con sotto-bucket per le librerie.
* Implementa un Inverted Index che mappa le parole chiave ai libri, consentendo una ricerca efficiente basata sulle parole chiave.

**MongoDB**

* Gestisce lo storico degli acquisti degli utenti.
* Utilizzato per proporre libri in base agli acquisti precedenti degli utenti.

**4. Progettazione del Database**

**Schema ER (Entity-Relationship)**

Descrizione delle entità principali (Libri, Utenti, Ordini, Raccomandazioni) e relazioni tra le entità.

**Immagine che contiene testo, schermata, cerchio, Prodotti generali

Descrizione generata automaticamente  
Schema logico delle funzionalità del progetto**

**Immagine che contiene schermata, Blu cobalto, testo

Descrizione generata automaticamente  
Schema logico dei replicaset per ogni network**

**Descrizione delle Tabelle e delle Relazioni**

Struttura delle collezioni e dei bucket, chiavi primarie e secondarie.

**5. Implementazione del Database**

**Costruzione dei nodi e dei cluster**

Una volta stabilita la necessità di avere cluster di nodi separati per ogni regione geografica e quindi per ogni lingua, abbiamo deciso di implementare (sia con MongoDB che con Riak) tre cluster composti da tre nodi ciascuno: i cluster Italia, Germania e Francia.  
Come suggeriscono i nomi, infatti, l’idea è quella di mantenere un partizionamento a livello geografico dei dati, così da rendere più efficienti i singoli nodi, ridurre i tempi necessari al soddisfacimento delle query e, all’occorrenza, poter fare statistiche localizzate sui vari utenti. Nonostante non abbiamo integrato questa funzionaliltà nel nostro progetto, uno sviluppo interessante tra quelli affrontati durante le lezioni potrebbe essere proprio l’implementazione di alcuni semplici script che forniscano statistiche localizzate sui generi di libri - o i singoli titoli - più letti nei vari Paesi, e organizzarne così in modo più efficiente la distribuzione. Inoltre, con una funzione Map-Reduce si potrebbero ottenere statistiche globali.  
L’implementazione descritta è agevolata proprio dal fatto che, di base, in ogni area geografica si venderanno libri nella lingua di appartenenza.

(la spiegazione appena fatta si può anche spostare, non deve stare qui)

Abbiamo dunque installato Docker sulle nostre macchine e seguito le guide ufficiali nella documentazione di Riak e MongoDB per scaricare le immagini *basho/riak-kv:latest* e *mongo:latest* e creare poi nove container organizzati in tre cluster.  
Nonostante l’interfaccia grafica di Docker, dopo alcune prove abbiamo stabilito che sia più conveniente inizializzare i vari nodi tramite una CLI (come il cmd di Windows), così da poter specificare espressamente alcuni parametri specifici come la porta del nodo, la rete e il ReplicaSet di appartenenza.  
Come prima cosa si crea una rete per ogni cluster (reti che noi abbiamo chiamato riakItalia, riakFrancia, riakGermania, clusterItalia, clusterFrancia, clusterGermania).

I nodi Riak sono stati chiamati:  
-Italia1, Italia2, Italia3  
-Francia1, Francia2, Francia3  
-Germania1, Germania2, Germania3

I nodi Mongo invece si chiamano:  
-libreria1, libreria2, libreria3 (Italia)  
-libreria4, libreria5, libreria6 (Francia)  
-libreria7, libreria8, libreria9 (Germania)

Per MongoDB, inoltre, abbiamo specificato il Replica Set di ogni cluster – rispettivamente ReplicaSetItalia, ReplicaSetFrancia, ReplicaSetGermania.

Per una vista completa su tutti i comandi eseguiti, consultare il file ‘Comandi-Mongo-Riak.txt’.

(qui si può inserire la parte sul quorum di Riak e Mongo, MA prima dobbiamo modificarli fisicamente -> per i replica set chiedere a chatgpt Write concern e Read Concern)

**Script e query utilizzati**

* Query per l'inserimento di dati
* Query per l'aggiornamento e la cancellazione di dati
* Query per la ricerca e il recupero di dati

**6. Prove di Utilizzo del Database**

**Descrizione dei Test Effettuati**

* Test di carico e performance
* Test di integrità dei dati

**Evidenza dell'Utilizzo del Database**

* Esempi di inserimento dati nell'inventario
* Esempi di recupero dati sugli acquisti
* Analisi delle performance durante i test

**7. Conclusioni**

**Risultati Ottenuti**

* Successi nel raggiungimento degli obiettivi
* Valutazione delle performance del sistema

**Criticità Riscontrate**

* Problemi tecnici affrontati e soluzioni adottate

**Possibili Miglioramenti Futuri**

* Estensioni funzionali e miglioramenti architetturali

**8. Appendice**

**Codice SQL**

* Script di creazione delle collezioni e dei bucket
* Esempi di query utilizzate

**Diagrammi e Grafici**

* Diagrammi ER e schema logico
* Grafici di performance

**Altro Materiale di Supporto**

* Documentazione aggiuntiva e riferimenti

**Modello di Business**

**Selezione Globale**

Offrire una vasta gamma di libri per soddisfare i gusti e le preferenze dei clienti.

**Personalizzazione**

Utilizzare l'analisi degli acquisti passati per suggerire libri pertinenti e personalizzati agli utenti, migliorando l'esperienza di shopping.

**Spedizioni Internazionali**

Collaborare con fornitori e corrieri per garantire spedizioni efficienti e tempestive in tutto il mondo.

**Programmi di Fidelizzazione**

Implementare programmi di fidelizzazione per incoraggiare gli acquisti ripetuti e premiare i clienti fedeli.

**Profilo dell'Acquirente**

Il nostro acquirente ideale è un appassionato di libri, desideroso di esplorare una vasta gamma di generi e autori provenienti da tutto il mondo. Si tratta di un individuo che apprezza la convenienza dello shopping online e cerca una piattaforma che offra una vasta selezione di libri, insieme a suggerimenti personalizzati basati sui suoi interessi e acquisti passati.

**Flusso del Processo di Acquisto**

**Navigazione e Ricerca**

Gli acquirenti navigano attraverso il catalogo online o utilizzano la barra di ricerca per trovare libri di loro interesse.

**Selezione del Prodotto**

Dopo aver trovato un libro desiderato, gli acquirenti possono visualizzare dettagli aggiuntivi come il prezzo, la disponibilità e le recensioni.

**Aggiunta al Carrello**

Gli acquirenti aggiungono il libro al loro carrello virtuale e possono continuare a navigare o procedere al pagamento.

**Pagamento**

Gli acquirenti procedono al pagamento utilizzando metodi di pagamento sicuri e affidabili.

**Conferma e Spedizione**

Dopo il completamento del pagamento, ricevono una conferma dell'ordine e il libro viene preparato per la spedizione.

**Consegna**

Una volta spedito, il libro viene consegnato all'indirizzo specificato dall'acquirente, con tempi di consegna stimati.

**Consistenza dei Dati e Replica Set**

**Replica Set per i Cluster Italia, Francia e Germania**

Un replica set in MongoDB è un gruppo di istanze di mongod che mantengono gli stessi dati. I replica set forniscono ridondanza e aumentano la disponibilità dei dati. Questo significa che se un nodo fallisce, un altro nodo può prendere il suo posto senza interrompere il servizio.

**Componenti di un Replica Set**

* **Primary**: Il nodo che riceve tutte le operazioni di scrittura. Ci può essere solo un nodo primario in un replica set.
* **Secondary**: I nodi che replicano i dati dal nodo primario. Possono servire operazioni di lettura (se configurati per farlo) e diventare il nodo primario se l'attuale primario fallisce.
* **Arbiter**: Un nodo che partecipa alle elezioni per determinare il nuovo nodo primario ma non mantiene una copia dei dati. È utile per mantenere un numero dispari di voti nel replica set.

**Vantaggi del Replica Set**

* **Disponibilità**: Se il nodo primario fallisce, i secondari possono eleggere un nuovo primario.
* **Scalabilità delle letture**: Le letture possono essere distribuite tra i nodi secondari.
* **Ridondanza**: I dati sono replicati su più nodi, riducendo il rischio di perdita di dati.

**Utilizzo di Replica Set in MongoDB**

Abbiamo deciso di utilizzare cluster con replica set in quanto i dati utilizzati non risultano eccessivamente grandi (sono solo stringhe di testo). Per quanto riguarda gli utenti, abbiamo pensato di non utilizzare uno sharding in quanto le richieste e gli ordini sul portale web non sarebbero così eccessive da necessitare uno shard. Il traffico non sarebbe così elevato dato l'argomento (libri) e quindi anche per gli utenti è più utile mantenere un replica set per la consistenza e per garantire in caso di guasti che la ridondanza di dati non crei disagi agli utenti.