GUIDA PER GLI SVILUPPATORI

Creazione di app Web di prim'ordine tramite Node.js e Couchbase

Eric Bishard, Developer Advocate

Couchbase

Indice

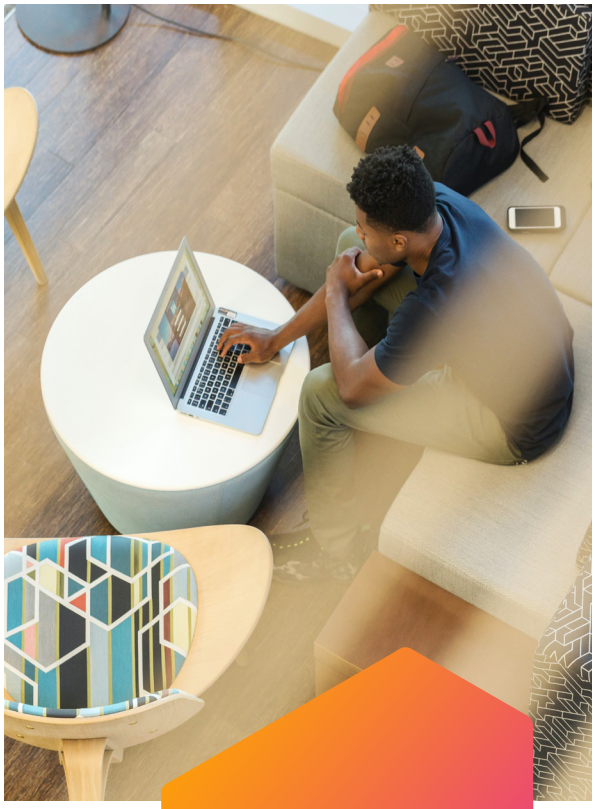
Introduzione

In qualità di sviluppatore Node.js, l'ultima cosa che dovrebbe causare preoccupazione è il database di applicazioni. Dopotutto, ci sono questioni più importanti che richiedono il suo tempo e la sua attenzione.

Per fortuna, i database NoSQL quali Couchbase si adattano perfettamente alle applicazioni JavaScript grazie al loro uso nativo di documenti JSON. Con JavaScript dal front-end al back-end, lo sviluppo è integrato e in questo modo può occuparsi di sfide più interessanti.

Se sta cercando di creare un'app Web di prim'ordine utilizzando Node.js, Couchbase è la soluzione perfetta per il suo database di back-end. L'ecosistema Couchbase offre un insieme di strumenti di altissimo livello e l'ambiente di sviluppo integrato perfetto per gli sviluppatori Node.js, quindi l'unico limite alle possibilità di creazione è la sua immaginazione.

In questa guida per gli sviluppatori, osserveremo da vicino come sviluppare app Web con prestazioni elevate utilizzando Node.js e Couchbase. Iniziamo.



Perché Couchbase?

Couchbase è un database distribuito di documenti JSON con tutte le capacità di un archivio chiave-valore, di un motore di ricerca e di un RDBMS, che comprende SQL, transazioni ACID distribuite e tanto altro.

Da un lato, Couchbase è creato per microservizi e per il computing basato sui consumi serverless nel cloud; dall'altro, è progettato per l'edge computing su dispositivi mobili/IoT connessi occasionalmente e in locale. Riassumendo: Couchbase è perfetto per i deployment multi-cloud.

Poiché Couchbase gestisce documenti JSON, uno schema hard-coded nel database non è più necessario, anche se Couchbase include una struttura logica che rende l'organizzazione di quei documenti intuitiva come quella di uno schema in vecchio stile. La definizione degli oggetti dell'applicazione, disponibile all'interno di JSON, è lo schema modificabile controllabile dallo sviluppatore.

Dovrà scrivere il JSON nel database una sola volta, quindi potrà applicarvi numerose capacità di elaborazione dei dati, tra cui:

- Piena programmabilità SQL, comprese le transazioni ACID
- Velocità di elaborazione di cache in memoria
- Capacità dell'archivio chiave-valore
- Ricerca di testo completo (recupero di informazioni)
- Analisi dei dati (query ad hoc)
- Programmazione basata su eventi (reattiva) e acquisizione dei dati di modifica

Di conseguenza, Couchbase funge da sistema affidabile di registrazione e al contempo gestisce operazioni chiave-valore con una latenza di microsecondi, query SQL e ricerche di testo in millisecondi e query di analisi della durata di decine di secondi senza che una query ostacoli un'altra.

I database tradizionali non sono assolutamente in grado di offrire queste capacità in modo predefinito.

Utilizzando Couchbase, riduce la proliferazione di dati e database, migliora la sicurezza, riduce l'amministrazione e i costi complessivi. Soprattutto, però, Couchbase le consente di creare rapidamente applicazioni e di distribuirle su vasta scala.

Non conosce Couchbase? Inizi con i nostri documenti per gli sviluppatori. In alternativa, provi gratuitamente Couchbase Cloud: si iscriva oggi.

Utilizzo di Node.js con Couchbase

Sono disponibili tre modi principali per utilizzare Node.js con Couchbase Server:

- SDK di Node.js
- Object Data Modeler (ODM) Ottoman di Node
- Stack RAGE con Couchbase

Vediamoli tutti più dettagliatamente.

SDK di Node.js

L'SDK di Node.js per Couchbase è la scelta predefinita per gli sviluppatori JavaScript che creano applicazioni con Node.js.

La versione SDK 3.x è una riscrittura completa dell'API 2.x che fornisce un'area più semplice e aggiunge un supporto delle funzionalità future di Couchbase Server, quali Collections and Scopes (Raccolte e ambiti).

L'SDK 3.x inoltre introduce le promesse per ridurre la complessità di JavaScript asincrono nelle applicazioni client, oltre che ampliare le API di gestione e offrire migliori opzioni di debugging e di registrazione per lo sviluppatore.

Per ulteriori informazioni su come utilizzare l'SDK di Node.js, consulti Install and Start Using the Node.js SDK with Couchbase Server (Installazione e utilizzo dell'SDK di Node.js con Couchbase Server) o visiti il Portale per gli sviluppatori di Node.js.

ODM Ottoman di Node

Ottoman è un Object Data Modeler (ODM) per l'SDK di Node.js per Couchbase che offre lo schema e la convalida JSON per NoSQL. È progettato per eliminare la maggior parte del codice boilerplate necessario per creare applicazioni Node.js con Couchbase, in modo da poter creare sistemi facili da progettare, aggiornare e scalare.

Con Ottoman, dichiara lo schema nel codice. Anche se Couchbase non ha l'applicazione dello schema per i suoi documenti, la maggior parte delle applicazioni richiedono qualche livello di schema anche in NoSQL.

Sebbene Ottoman crei un'astrazione tramite l'SDK Couchbase, i vantaggi superano gli svantaggi. Uno sviluppatore genera molta logica durante la creazione e l'aggiornamento di documenti, la scrittura nelle fasi precedenti e successive del ciclo di vita e l'utilizzo delle strutture dei dati e della convalida.

Per ulteriori informazioni su come utilizzare l'ODM Ottoman con Couchbase, consulti la nostra Developer Quick Start Guide for Ottoman (Guida all'avvio rapido di Ottoman per sviluppatori) o faccia un tour generale di Ottoman in questo post del blog.

Stack RAGE con Couchbase

Oltre a Couchbase Server, è anche possibile utilizzare lo stack RAGE — React, Apollo Client, GraphQL ed Express — per lo sviluppo di applicazioni JavaScript con stack completo.

È possibile utilizzare React, Apollo e GraphQL sul lato client ed Express ed Express GraphQL sul lato server. L'applicazione Express utilizza anche l'SDK di Node.js per Couchbase, il che le consente di connettersi al database Couchbase, eseguire query e restituire i dati all'API.

Per una dimostrazione dettagliata dello sviluppo con stack completo con RAGE e Couchbase, guardi questa presentazione di Couchbase Connect on-demand (con il relativo repository di codice di accompagnamento).

Progetti esemplificativi utilizzando Couchbase e Node.js

Ora che ha acquisito familiarità con le modalità di utilizzo di Node.js e Couchbase insieme, è giunto il momento di immergerci in alcuni casi di utilizzo e progetti.

Il caso di utilizzo più conosciuto di Couchbase e Node.js riguarda i profili degli utenti e gli archivi di sessioni.

Archivi di profili di utenti e di sessioni

Uno dei più comuni casi di utilizzo per un database NoSQL riguarda la creazione di un archivio di profili di utenti e di sessioni. Perché proprio NoSQL? Perché i profili spesso devono essere flessibili e accettare frequenti modifiche dei dati. Mentre le modifiche sono possibili con un RDBMS tradizionale, le modifiche regolari con un database relazionale richiedono una maggiore mole di lavoro e impongono un peso maggiore sulle prestazioni.

Quindi, come creare un profilo utente utilizzando Node.js e Couchbase Server? Ecco una panoramica generale. Per una descrizione dettagliata, consulti questo video: Develop a User Profile and Session Store with Node.js (Sviluppo di un archivio di profili di utenti e di sessioni con Node.js).

Quando si tratta di gestire i dati degli utenti, ha bisogno di un modo per creare un archivio di profili di utenti e un archivio di sessioni, quindi di associarli ad altri documenti JSON. Di seguito alcuni principi importanti da tenere a mente durante la creazione di un archivio di profili di utenti:

- Archiviare i dati degli account, quali nomi utenti e password, in un documento di profilo.
- Trasmettere dati sensibili degli utenti con ogni richiesta di azione dell'utente.
- Utilizzare una sessione che scade dopo un periodo di tempo stabilito.
- Archiviare i documenti della sessione con una scadenza.

Video post del blog: Develop a User Profile Store and Session Store with Node.js - Video

Post del blog precedente: Create a User Profile Store with Node.js and a NoSQL Database

FASE 1: CREAZIONE DELL'API CON NODE ED EXPRESS

Innanzitutto, crei una directory di progetto per l'app Node.js e installi eventuali dipendenze.

Questa azione crea una directory funzionante per il progetto e inizializza un nuovo progetto Node.js. Le dipendenze includono l'SDK di Node.js per Couchbase ed Express Framework e altre librerie di utilità quale body-parser per l'accettazione dei dati JSON tramite richieste POST, uuid per la generazione di chiavi univoche e bcryptjs per l'hash delle password in modo da dissuadere utenti malintenzionati.

Quindi, esegua il bootstrap dell'applicazione con un file server.js. È anche necessario creare un indice in Couchbase Server, poiché utilizzerà il linguaggio di query N1QL per uno degli endpoint. Gli indici primari non sono consigliati per il codice a livello produttivo.

FASE 2: SALVATAGGIO DI UN NUOVO UTENTE NELL'ARCHIVIO DI PROFILI

Un profilo utente contiene tutte le informazioni che descrivono un utente, quali indirizzo, numero di telefono, profili dei social media e tanto altro. Non è mai una buona idea archiviare le credenziali dell'account nello stesso documento contenente le informazioni di base sul profilo di un utente. Avrà bisogno di almeno due documenti per ogni utente, quindi vediamo come strutturare questi documenti.

Il documento del profilo utente deve avere una chiave da utilizzare come riferimento nei documenti correlati. Nella maggior parte dei casi, questa chiave è un UUID generato automaticamente.

Il documento del profilo avrà un valore JSON che comprende due proprietà email e una proprietà type. La proprietà type rappresenta un indicatore importante che descrive il documento in maniera analoga a come una tabella organizza i record in un database relazionale. Si tratta di una convenzione standard in un database di documenti.

Il documento dell'account associato al profilo utente avrà una chiave che corrisponde all'indirizzo e-mail di un determinato utente. Il documento dell'account deve avere un type e un pid che fa riferimento alla chiave del documento di profilo, oltre a un indirizzo e-mail e una password con hash.

Ora ha un modello stabilito per ogni documento e una strategia per correlare quei documenti senza aver creato alcun vincolo per il database.

FASE 3: UN ENDPOINT PER LA CREAZIONE DI ACCOUNT

La fase successiva vedrà la creazione di un endpoint per la creazione dell'account nel file server.js. Si assicuri di includere tutti gli elementi seguenti.

Innanzitutto, verifichi che nella richiesta siano presenti sia una email sia una password.

Quindi, il suo endpoint dovrebbe creare un oggetto account e un oggetto profile sulla base dei dati inviati nella richiesta. Il pid che sta salvando nell'oggetto account deve corrispondere a una chiave univoca e deve essere impostato come chiave del documento per l'oggetto profile.

Il documento account deve utilizzare l'e-mail come chiave. In futuro, se saranno necessari altri dettagli dell'account (ad esempio, e-mail alternativa, accesso ai social ecc.) potrà associare altri documenti allo stesso profilo.

Piuttosto che salvare la password nell'oggetto account come testo semplice, dovrà trasformarlo in hash tramite Bcrypt. Per motivi di sicurezza, la password deve essere separata dall'oggetto profile. (Per maggiori informazioni sull'hash delle password, veda questo tutorial).

Ora che i dati sono pronti, è arrivato il momento di inserirli in Couchbase. L'obiettivo di questo salvataggio deve essere tutto o niente. I documenti account e profile devono essere creati correttamente; in caso contrario, è necessario eseguire il rollback dell'intero salvataggio. A seconda del risultato del salvataggio, restituirà al client alcune informazioni.

Per inserire i dati, avrebbe potuto utilizzare le query N1QL, ma è più semplice utilizzare le operazioni CRUD senza conseguenze sulle prestazioni.

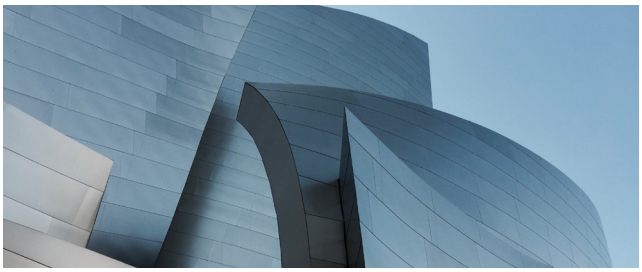
FASE 4: UTILIZZO DI TOKEN DI SESSIONE PER DATI SENSIBILI

Dopo aver creato il profilo e l'account, l'utente dovrebbe accedere e stabilire una sessione che verrà archiviata nel database con riferimento al relativo profilo utente. Per questo documento di sessione deve essere impostata una scadenza finale e la sua rimozione dal database.

Il modello dei dati della sessione, così come altri documenti, devono avere un type diverso. Proprio come il documento account, il modello deve avere una proprietà pid che fa riferimento al profilo utente di destinazione.
Il codice che rende possibile questa operazione deve trovarsi nell'endpoint login.

Questo endpoint deve convalidare i dati in entrata, quindi eseguire una ricerca dell'account per indirizzo e-mail. Se per l'e-mail vengono restituiti dati, può confrontare la password in entrata con la password con hash restituita nella ricerca dell'account. Se questa operazione riesce, l'endpoint crea una nuova sessione per l'utente.

A differenza della precedente operazione insert, deve impostare una scadenza del documento a un'ora (3600 secondi). Se la scadenza non viene aggiornata, è necessario impostare il documento in modo che venga rimosso. Questo va bene, perché costringe l'utente a effettuare nuovamente l'accesso e a creare una nuova sessione. In tutte le richieste future, verrà quindi trasmesso il token di questa sessione anziché la password.



FASE 5: GESTIONE DI UNA SESSIONE UTENTE CON I TOKEN

Alla fine, per ottenere le informazioni sul profilo di un utente e associare al profilo nuove informazioni, dovrà confermare l'autorità tramite la sessione.

È possibile confermare la validità della sessione utilizzando il middleware. Una funzione middleware può essere aggiunta a qualsiasi endpoint Express. Questa convalida deve essere una semplice funzione che ha accesso alla richiesta HTTP dell'endpoint in uso.

Il codice dovrà verificare la presenza di un'intestazione di autorizzazione nella richiesta. Se è presente un token di connessione con ID di sessione (sid) è possibile eseguire una ricerca. Il documento della sessione deve contenere l'ID del profilo. Se la ricerca della sessione ha esito positivo, salvi l'ID del profilo (pid) nella richiesta.

Quindi, dovrà aggiornare la scadenza della sessione, passare al middleware e tornare all'endpoint. Se la sessione non esiste, non verrà trasmesso alcun profile id e la richiesta non andrà a buon fine.

Successivamente, potrà utilizzare il middleware per ottenere informazioni sul profilo nell'endpoint dell'account. Nota: la convalida deve avvenire prima del resto della richiesta.

RIFLESSIONI CONCLUSIVE

Queste sono le fasi generali della creazione di un archivio di profili di utenti e di sessioni utilizzando Node.js e Couchbase.

Come già indicato, per una descrizione dettagliata, consulti questo video: Develop a User Profile and Session Store with Node.js (Sviluppo di un archivio di profili di utenti e di sessioni con Node.js). Per materiale di lettura più avanzato, consulti questo articolo: User Profile Store: Advanced Data Modeling Part 1 (Archivio di profili utente: modellazione avanzata dei dati - parte 1). Il codice citato negli articoli di cui sopra si trova nel repository couchbaselabs/couchbase-nodejs-blog-api in GitHub.

Altri progetti esemplificativi

Gli archivi di profili di utenti e di sessioni sono il caso di utilizzo più comune, ma potrà utilizzare Couchbase e Node.js per molte altre applicazioni e molti altri progetti, tra cui:

- Funzionalità di ricerca di testo completo
(sia autonoma che come parte di un progetto più grande)
- Applicazioni chatbot
- Applicazioni di luoghi di interesse e di viaggio
- Applicazioni di bitcoin e altre criptovalute
- Analisi di set di dati di grandi dimensioni

Conclusione

Non dovrà avere alcuna incertezza su quale database utilizzare per la sua prossima app Node.js. Come database di documenti NoSQL, Couchbase è la soluzione perfetta. In questa guida per gli sviluppatori, ha imparato:

- Perché Couchbase è la scelta più indicata per la creazione con Node.js
- Come utilizzare Node.js insieme a Couchbase, compresi SDK, ODM Ottoman e stack RAGE
- Quali progetti comuni utilizzino sia Node.js sia Couchbase per affrontare le odierne sfide di sviluppo

Per maggiori informazioni su come utilizzare Node.js con Couchbase, consulti il nostro portale per gli sviluppatori Node.js. Ora, quando vorrà creare una nuova app Web, le sue possibilità saranno illimitate. L'unica domanda è: che cosa creerà?

Couchbase

About Couchbase

At Couchbase, we believe data is at the heart of the enterprise. We empower developers and architects to build, deploy, and run their mission-critical applications. Couchbase delivers a high-performance, flexible and scalable modern database that runs across the data center and any cloud. Many of the world's largest enterprises rely on Couchbase to power the core applications their businesses depend on.

For more information, visit www.couchbase.com.