Couchbase NoSQL Developer Workshop

ラボハンドブック

ラボ **2:** 全文検索操作

手順

このラボの目的は、Couchbase の Node.js SDK を使用して、フルテキスト検索 (FTS) 操作を実行し、FTS の結果に基づいて複数のドキュメントを返すことによって、製品の検索を有効にするロジックを作成することです。 FTS 操作の詳細については、SDK のドキュメントを参照してください。

ステップ 1: API にロジックを追加する

API リポジトリ ディレクトリで *repository.js* ファイルを開きます (付録の API のプロジェクトのストラクチャーを参照してください)。 *searchProducts( )*メソッドを検索します。

ドキュメント：

- SDK を使用した全文検索

NOTE ***

K/V get 操作によって得られる result オブジェクトには、ドキュメントの内容とメタデータが含まれています。 このラボでは、ドキュメントの内容のみを返す必要があります。 そのため、*result.value* を返します。

searchProducts() 入力：

- product: 文字列 - 製品で使用する検索用語
- fuzziness: 整数
- callback

searchProducts() 出力:

- エラー オブジェクト (該当する場合)
- products: 配列 - 検索で見つかった製品ドキュメント

*searchProducts()*メソッドの実装については、以下のコード スニペットを参照してください。
注意: NOP コード行 (例. callback(null, “NOP”)) をコメントアウトするか、新しいコードに置き換えます。

```
0:  async searchProducts(product, fuzziness) {
1:    try {
2:      /**
3:       * Lab 2:  Search operation (FTS)
4:       *  1.  FTS:
5:       *        term query w/ fuzziness
6:       *        use "basic-search" as index name for searchQuery
7:       *  2.  K/V getMulti() using FTS results
8:       *
9:       */
10:
11:      let result = await this.cluster.searchQuery(
12:        "basic-search",
13:        couchbase.SearchQuery.term(product).fuzziness(fuzziness),
14:        {
15:          limit: 100,
16:        }
17:      );
```

```
18:
19:       let docIds = result.rows.map((hit) => hit.id);
20:       //uncomment to see doc count
21:       // outputMessage(
22:       //   docIds.length,
23:       //   "repository.js:searchProducts() - total docs:"
24:       // );
25:       let results = await Promise.all(
26:         docIds.map((id) => {
27:           return this.collection.get(id);
28:         })
29:       );
30:       return { products: results.map((res) => res.value), error: null };
31:     } catch (err) {
32:       //Optional - add business logic to handle error types
33:       outputMessage(err, "repository.js:searcProducts() - error:");
34:       return { products: null, error: err };
35:     }
36:   }
```

コードに関する注意事項:

- aysnc/await 構文を使用します。
- 行 11 から 17: 次のパラメーターを使用して検索クエリを実行しています。 検索操作は、3.x SDK のクラスター・レベルで行われます。
    - 使用する FTS インデックス (これは Couchbase サーバーが持つインデックスです)
    - 使用する検索クエリの種類
    - 結果の制限
- 19 行: 検索結果に基づいて製品ドキュメント ID の一覧を取得します。
- 25-29 行: Promise API を使用して一括取得操作を実行し、検索から返されたすべての製品ドキュメントを取得する
- 30 行目: 検索で見つかった製品ドキュメントの返品
- outputMessage(): コンソールに情報を簡単に出力するためのヘルパーメソッドは、/library ディレクトリにあります (付録で詳しく説明されている API のプロジェクト構造を参照してください)
- try/catch & err オブジェクトの処理は汎用的な方法で意図的に行われます。 ラボ参加者は、エラーを処理するさまざまな方法をテストするために、それに応じてロジックを自由に追加できます。
- outputMessage() コード ブロックのコメントを解除してテストする場合は、コンソール ログにその出力が表示されることを忘れないでください。 *表示するには、`docker logs api`*を実行します。

完了したら、*repository.js* ファイルが保存されていることを確認します。 API *Docker* コンテナは API の作業ディレクトリにマップされるため、API コードに対して行われたすべての更新はコンテナに反映される必要があります。
以下の手順に従って、*searchProducts()* ロジックを検証します。

1. http://localhost:8080 に移動
2. ログインしていない場合、ログインします。
3. ホームページにいない場合(*検索ボックス*が表示されない)
    a. 左上隅にある*[Couchbase|NoEQUAL]*画像をクリックしてホーム画面に移動します。

4. *検索ボックス*に製品名（の一部）を入力し、*虫眼鏡*をクリックして検索を実行します。
    a. 検索語の例: shirt
5. 検索語に一致する製品が存在する場合、製品が画面に表示されます(下の画像を参照)。