Indexing MapR-DB Data in Elasticsearch

You can create external indexes for your MapR-DB data by indexing columns, column families, or entire tables in Elasticsearch. When client applications update data in a source table, MapRDB replicates the update to the Elasticsearch type that is associated with it.

Updates to indexes happen in near real-time because individual updates to your MapR-DB source tables are replicated to Elasticsearch. There is no batching of updates, which would lead to recurring times where data is available in MapR-DB but not searchable in your indexes. Therefore, there is minimal latency between the availability of data in MapR-DB and the searchability of that data by end users.

The MapR distribution does not include Elasticsearch, which you can get from https://www.elastic.co/. MapR-DB works with Elasticsearch version 1.4.

Uses of Indexing MapR-DB Data in Elasticsearch

Indexing MapR-DB data in Elasticsearch makes possible these use cases:

Full-text searches

Client applications can run queries against text documents to search for specific words.

For example, a web-based travel agency could allow customers to search for destinations based on keywords. The travel agency indexes text documents that describe hotels, their location, their accommodations, and nearby attractions. So, a customer could search on the terms “Costa Rica”, “spa”, and “monkeys” to try finding a hotel in Costa Rica that offers a spa and is near businesses that give jungle tours in which the customer can see monkeys.

To enable full-text search in your existing MapR-DB applications, you do not need to add complex code for indexing data. Instead, you add code for searching with Elasticsearch and let the integration of MapR-DB with Elasticsearch handle indexing.

Geospatial searches

Enable geospatial searches by indexing location information (latitude and longitude). For example, indexed Elasticsearch documents might represent restaurants. If those documents contained location information, end users could use query filters to search for all restaurants within a certain radius of their current location.

Elasticsearch entities

MapR-DB data is indexed in Elasticsearch indexes, which consist of types. Types in turn consist of JSON documents.

An *index* is the highest logical entity in Elasticsearch. It is equivalent to a database in traditional relational databases and to a group of related tables in MapR-DB. An Elasticsearch cluster can host multiple indexes.

An index contains one or more *types*. A type is equivalent to a table in traditional databases or in MapR-DB.

A type stores multiple JSON *documents*. Each row of a MapR-DB table that you are indexing corresponds to a JSON document in Elasticsearch. Consider the following example. Suppose that you are indexing the columns that are listed in the table below.

Column Family	Indexed Columns
personal	last_name
	first_name
purchase	brand
	item
	price
review	text
	rating

For each row of the table, there would be a JSON document in Elasticsearch with this structure:

```
{
      personal : {
           first_name : value,
```

```
            last_name : value
             },
       purchase : {
            brand : value,
            item : value,
            price : value
             },
       review : {
            text : value,
            rating : value
             }
}
```

This diagram summarizes how entities in Elasticsearch correspond to entities in MapR-DB: