The RediSearch 2.x module is a source-available project that lets you build powerful searches for open-source Redis databases. When combined with Redis Enterprise Software (RS), you can use the same RediSearch protocols and commands to get a geo-replicated query, and full-text search over efficient in-memory indexes.

You can index more than one field per document, and these fields can represent text, numeric, or geospatial data types. With RediSearch indexes you can do language-aware fuzzy matching, fast auto-complete, exact phrase matching, numeric filtering, and geo-radius queries.

As the documents in your database change, the index automatically processes these changes to keep your searches updated. You can also limit the index to only contain the index structure so that the results return a unique docID.

For full-text searches, you can customize the field queries and ranking of the search results. When querying, you can use multiple predicates that query both text, numeric, and geospatial fields in one query. You can also sort by a specific field and limit the results with an offset to easily produce customized results pages.

RediSearch supports over 15 natural languages for stemming and includes auto-complete engines with specific commands that can provide real-time interactive search suggestions.

Storing documents

The RediSearch engine indexes "documents", which are a list of field-value pairs. The index knows how to index each field, but that’s not enough. We need to actually store the data for retrieval. In a simple use case, you can just push documents to the database and RediSearch creates a complete index and document database.

For storing documents, the RediSearch engine relies on Redis hashes, where each document is represented by a single key, and each property and its value by a hash key and element. For retrieval, you simply perform an HGETALL query on each retrieved document, returning its entire data. If the user needs to retrieve a specific document by its ID, a simple HGETALL can be performed by the user.

RediSearch in Active-Active databases

As a result of the new RediSearch architecture and methodology, RediSearch 2.x supports Active-Active databases. You can now serve your index information from geo-distributed database instances.

Resharding and RediSearch

By moving the index out of the keyspace and structuring the data as hashes, RediSearch 2.x makes it possible to reshard the database. When half of the data moves to the new shard, the index related to that data is created synchronously and RediSearch removes the keys from the index when it detects that the keys were deleted. Because the index on the new shard is created synchronously though, it’s expected that the resharding process will take longer than resharding of a database without RediSearch.

Limitations

- You cannot use RediSearch with the OSS Cluster API.

More info

- Getting Started with RediSearch 2.0