RediSearch

Provides secondary indexing,full-text-search,query language for redis
These features enable multi-field queries,aggregation,exact phrase matching and numeric filtering for text queries.

Redis allows storing data into json and hash format as well.
Hash → key value pair
Json → entire json object

Redis searches the data based on the ranking that we provided.
Redis uses stemming → let's say you provide word management , redis search does the stemming operation on manage and it will try to search all the data that matches a particular word.
Manage word → manager,manage,management

Aggregation operations also can be done using this.
When to use redis search → as we know redis work on principal of in memory data
Wherever we need quick search we can implement redis search but for large data don't use this because huge amounts of data stored in memory doesn't make sense.

How redis search works → work on principal of secondary indexing.
You can store data on hash and json and create indexes
For all json we can create an index and with that index we can define for what particular key you need to create the index. Let's say I want to create indexing on all the keys starting with a particular prefix.
We can define a prefix and based on that it will create an index.
Also define a field and other config.
Currently there is no way to modify the index , you have to drop the index and recreate the new index.
Once you do the index , try to search on the index that you have created.

Install redis-stack - first install docker and then run this command

```
docker run -d --name redis-stack -p 6379:6379 -p 8001:8001
redis/redis-stack:latest
```

Go to browser - localhost:8001

[***Redis cloud , redisinsight**]

go to redisinsight →

Different redis search command →
→JSON.SET “KEY” “PATH” “JSON OBJECT” –STORE DATA
⇒ JSON.SET “post:1” “$” “ {\”postId\”:\”1\ ”,\”title\” ….}”

→”FT.CREATE” “INDEX-NAME” “ON” “JSON” “PREFIX” “1” “PREFIX-NAME” “SCHEMA” “$.PROPERTY” “AS” “PROPERTY” “TEXT” “SORTABLE” “$.PROPERTY” “AS” “PROPERTY” “TEXT” “$.PROPERTY” “AS” “PROPERTY” “NUMERIC” “NOINDEX”
⇒ ”FT.CREATE” “post-idx” “ON” “JSON” “PREFIX” “1” “post” “SCHEMA” “$.content” “AS” “content” “TEXT” “SORTABLE” “$.title” “AS” “title” “TEXT” “$.views” “AS” “views” “NUMERIC” “NOINDEX”

→ FT_LIST – check how many index are available
⇒ post_idx

→FT_INFO post_idx – give info about index

→FT.DROPINDEX post-idx – delete the index
→