International Journal of Management, IT & Engineering
Vol.14 Issue 10, October 2024,
ISSN: 2249-0558 Impact Factor: 7.119
Journal Homepage: http://www.ijmra.us, Email: editorijmie@gmail.com
Double-Blind Peer Reviewed Refereed Open Access International Journal - Included in the International Serial Directories Indexed & Listed at: Ulrich's Periodicals Directory ©, U.S.A., Open J-Gate as well as in Cabell's Directories of Publishing Opportunities, U.S.A

How elastic search is boosting performance of property searches on Real estate Platforms

Jatla.Prasanna Senior Backend Engineer at One Real

This paper gives comprehensive details on how ElasticSearch could power property searching in Real estate platforms. Searching is at the heart of any real estate listing platform. Usually these platforms get feeds from external agencies with millions of property listings. This feed contains property details like type of property, address, price and amenities to name a few.

When a user is searching for a property, they have the ability to apply multiple filters. Users can search for something like looking for a 3-4 bedroom house with a pool and home theater around 5 miles from zip code 95034. The platform has to show all the listings in the area which matches the filter queries. The search includes a combination of Geospatial search, full text search and term search. This is the best use case where we can leverage the full potential of elastic search.
We could search millions of documents within milliseconds and sort the results by custom relevance sorting. This api should be able to send recommendations with low latency.

Elastic search is a search engine built on top of Apache lucene. A document in ElasticSearch is analogous to a row in RDS. Property listings from agencies are typically given as json strings including property details like when is it listed, who listed it, pictures of the property, agent details, amenities, etc,. And different agencies will have different schema formats to send these details. While reading data from different agencies, we need to create a common canonical schema regardless of its source for internal usage. Real estate companies need to run a batch job nightly to read the feed. In most of the cases, data is first stored to RDS and then sent to elasticsearch for indexing. There could be many ways to index the feed, but one of the efficient ways is to create a new index with a timestamp whenever we are accepting a full feed from external agencies. We could maintain data about active indexes in a RDS or create an alias in elastic search and make sure the alias always points to the active index. When the data being ingested to the new index, all the search queries are routed to the old index and once the ingestion is completed we need to make aliases now point to the new index. For any individual document update, we can use update api. One good practice is to update the document in both current active index and new index which is being processed if any.
Elastic Search allows to do Term search as well Full text search.Analyzer is the component in ElasticSearch that is responsible for parsing the document and tokenizing. Text analysis in elastic search unables to full text search rather finding just exact matches. It is possible through Tokenization, which breaks the text into individual tokens. And in the subsequent steps this tokens are standardized, this allows you to match tokens that are not exactly the same as the search terms, but similar enough to still be relevant.To ensure search terms match these words as intended, you can apply the same tokenization and normalization rules to the query string.Term search looks for exact matches of a keyword or value. It is primarily used for structured or keyword data where exact matches are required. In case of our Property searches we can utilize the power of full text search, term search and fuzzy search. Fuzzy search is another concept where elastic search not just matches words or synonyms but finds matched terms by inserting, updating or deleting characters from analyzed terms. It uses Damerau-Levenshtein distance formula to calculate distance between saved term and query term.

High Availability and Low latency:

When a user is looking for properties, they expect to see results as quickly as possible and the platform should be highly available. ElasticSearch is a distributed search engine, we can scale horizontally by adding new nodes in the cluster. While creating the index we set up shards and replication factors. Data is replicated to more than one node, in case if any of the nodes goes down, we still get results from one of the replication nodes. There are two steps involved while Searching data. For CRUD operations documents are routed to corresponding shard depending on the document id and no.of shards. Whereas in Search query it involves two phases, first Search and then Fetch. Search query is sent to all the shards and each shard locally finds the document ids that match the query and sorts them according to relevance. In this phase all the documents in all shards are traversed and calculates score for each matching document. That is potentially time consuming. All these document ids are then sent to the coordinator query where it sorts all the results and fetches the actual document content for those ids. We need to make sure there is no data skew on any particular shard, as it becomes a hotspot while fetching data. We can configure the routing value to make sure data is evenly distributed among all shards. There are millions of active listings and tens of millions of active users searching for properties daily. To return results quickly, it is best practice to implement techniques like pagination. Data ingestion to ElasticSearch can be done as bulk inserts/updates. With this approach of bulk indexing, through nightly batches as we discussed earlier would give better performance for ingestion but it would result in stale data. If any new listing is created, it won't be available on the platform until the nightly job is done.

Explaining GeoSpatial capabilities of ElasticSearch:

For Geo spatial queries, ElasticSearch provides GeoPoint and GeoShape. GeoShape can represent a rectangle, polygon, etc,. There are two sets of geo spatial data in Property Search, one is the property address itself and another is agent service area. In some cases, users would be interested in finding the agents on the platform who are licensed realtors in that service area. In this case Agent's service area has to be saved as Geo_shape, which indicates the area in which this agent is working. Users usually query for something all "get agents near me within a 5 miles radius". Another case is where the user is searching for listings within 5 mile radius from given latitude, longitude. For this to work, usually property addresses are saved as Geopoint and we can query elastic search using GeoPoint queries to get results within the boundary.
ElasticSearch uses KD and BKD trees to store geospatial information.

PUT /properties/_mapping { "properties": { "location": { "type": "geo_point" } } }

Then you can search for documents within a specified distance from a given location using the **geo_distance** query:

GET /properties/_search { "query": { "geo_distance": { "distance": "5km", "location": { "lat": 37.7749, "lon": -122.4194 }} } }

Elasticsearch provides the convenience of completing a search query without typing in all of the characters.ElasticSearch can power auto-recommendation systems by leveraging its advanced

search and machine learning capabilities. These recommendations can be tailored based on user behavior, preferences, or content similarity, making it a powerful tool for delivering personalized results.Autocomplete is a widely used feature in search applications, providing users with relevant suggestions as they type their query.Achieving Elasticsearch autocomplete functionality is facilitated by the search_as_you_type field datatype. No one is interested in typing complete addresses, so use elastic search auto suggestions to your advantage in the portal while the user is trying to type an address for property search.

References

[1] Optimizing Elasticsearch for Low Latency, Real-Time Recommendations.

[2] https://www.elastic.co/docs/

[3] Elasticsearch's Distributed Search: Query and Fetch Phases by Musab Dogan
[4]BKD trees, used in Elasticsearch by Gaurav sarma

Author Profile

Prasanna is working as Senior Backend Engineer at OneReal.
Email : Prasanna.webtech93@gmaail.com