**Keywords**

Elasticsearch has a rich set of keywords that are used in various parts of its configuration, queries, and indexing process. These keywords are categorized based on the operations they are associated with, such as mapping, querying, aggregations, and cluster management. Here’s a comprehensive list of the most important Elasticsearch keywords you should know:

**1. Mapping & Indexing Keywords**

These keywords are used for defining the structure of your data and how it's indexed in Elasticsearch.

* **index**: Defines the index name where data is stored.
* **type**: Defines the type of document in older versions (deprecated in recent versions).
* **mappings**: Defines how fields should be mapped (their data type, etc.).
* **properties**: Defines fields within a mapping (field names and their types).
* **fields**: Defines multi-fields, where a single field is indexed in multiple ways (e.g., as both a text and keyword field).
* **dynamic**: Controls whether fields that are not explicitly mapped should be added automatically.
* **enabled**: Controls whether the mapping allows storing or indexing a field.
* **copy\_to**: Allows copying the content of multiple fields into one.
* **meta**: Allows you to store metadata as part of the field.
* **store**: Whether the original field value should be stored in the index.
* **analyzer**: Defines which analyzer (tokenizer, filters, etc.) should be used on a text field.
* **norms**: Stores normalization factors for scoring (text fields).
* **eager\_global\_ordinals**: Forces global ordinals to be loaded at search time for better performance.
* **boost**: Boost the relevance score of a field or query.

**Field Data Types:**

* **text**: Used for full-text search.
* **keyword**: Used for structured fields like tags, IDs, etc. (exact matches).
* **integer**, **long**, **short**, **byte**: Numeric field types.
* **float**, **double**, **half\_float**, **scaled\_float**: Floating point field types.
* **date**: Stores date values.
* **boolean**: Stores true or false.
* **ip**: Stores IP addresses.
* **object**: Defines a JSON object.
* **nested**: Defines arrays of objects where each object is indexed independently.
* **geo\_point**: Defines geographical points.
* **geo\_shape**: Defines complex geographical shapes.
* **completion**: Used for autocomplete functionality.
* **token\_count**: Stores the number of tokens for a text field.

**2. Querying & Filtering Keywords**

These keywords are used to perform searches and filter data in Elasticsearch.

* **query**: The root keyword to define the search query.
* **match**: Used for matching text fields (full-text search).
* **term**: Used for exact match queries (e.g., keywords, numbers).
* **range**: Used to filter data within a range (e.g., dates, numbers).
* **bool**: Combines multiple queries using boolean logic (must, should, filter).
* **must**: Queries that **must** match.
* **must\_not**: Queries that **must not** match.
* **should**: Queries that are optional but will boost relevance if matched.
* **filter**: A filter that doesn’t affect scoring, often for exact matches.
* **match\_all**: Returns all documents.
* **multi\_match**: Matches a query across multiple fields.
* **exists**: Checks if a field exists in a document.
* **missing**: Checks if a field is missing (deprecated).
* **prefix**: Searches for documents where fields start with a specific string.
* **wildcard**: Searches for a field matching a wildcard expression.
* **fuzzy**: Performs fuzzy matching based on edit distance (Levenshtein distance).
* **terms**: Finds documents where a field matches any of the provided terms.
* **nested**: Queries nested fields.
* **boosting**: Allows decreasing the relevance of certain documents.
* **constant\_score**: Wraps a query to return a constant score.
* **dis\_max**: Returns documents that match one or more queries but uses the highest score.
* **function\_score**: Adjusts the relevance scores based on custom functions.
* **match\_phrase**: Searches for phrases (exact order).
* **script\_score**: Allows using a script to calculate relevance scores.
* **rescore**: Adjusts the ranking of search results based on secondary queries.
* **highlight**: Returns highlighted snippets for matching fields.
* **sort**: Defines sorting criteria for results.

**3. Aggregations Keywords**

These are used for performing data aggregations (similar to SQL GROUP BY and SUM operations).

* **aggs**: The root keyword for defining aggregations.
* **terms**: Aggregates data based on unique terms.
* **range**: Aggregates data within specific ranges (dates, numbers).
* **histogram**: Aggregates data into buckets based on intervals.
* **date\_histogram**: Aggregates data into buckets based on date intervals.
* **cardinality**: Counts the number of distinct values.
* **avg**: Calculates the average of a numeric field.
* **sum**: Calculates the sum of a numeric field.
* **min**: Returns the minimum value for a field.
* **max**: Returns the maximum value for a field.
* **stats**: Returns basic statistics (min, max, avg, sum, and count).
* **extended\_stats**: Returns extended statistics (variance, std deviation, etc.).
* **top\_hits**: Returns the top documents for each bucket.
* **percentiles**: Returns percentiles for a numeric field.
* **bucket\_script**: Allows executing a script on the buckets (e.g., for calculations).
* **nested**: Aggregates nested fields.
* **global**: Aggregates data across all documents, ignoring any query.

**4. Index Settings Keywords**

These control how the index behaves and how it's stored.

* **settings**: Defines the configuration for an index.
* **number\_of\_shards**: Specifies the number of primary shards.
* **number\_of\_replicas**: Specifies the number of replica shards.
* **refresh\_interval**: Defines how often the index is refreshed (visible to search).
* **max\_result\_window**: Controls the maximum number of results that can be paginated through.
* **analysis**: Defines analyzers, tokenizers, and filters used during indexing and querying.
* **filter**: Defines filters to be used in custom analyzers.
* **tokenizer**: Defines tokenizers used in custom analyzers.
* **similarity**: Controls how scoring is calculated (e.g., BM25, classic, etc.).

**5. Cluster Management Keywords**

These manage the Elasticsearch cluster and its nodes.

* **cluster**: Refers to the cluster-level operations.
* **nodes**: Refers to Elasticsearch nodes in the cluster.
* **shards**: Individual parts of an index that are distributed across nodes.
* **allocation**: Controls shard allocation on nodes.
* **health**: Provides the health status of the cluster (green, yellow, red).
* **status**: Provides the operational status of the cluster or an index.
* **snapshot**: Used for backup and restore operations.
* **reindex**: Reindex data from one index to another.
* **aliases**: Define alternate names for an index (virtual names).
* **ilm**: Index Lifecycle Management, defines policies for managing index lifecycle stages (rollover, deletion, etc.).

**6. Scripts and Painless Keywords**

Elasticsearch allows using scripts in queries for custom logic.

* **script**: Defines inline or stored scripts to be executed in queries.
* **lang**: Specifies the scripting language (e.g., painless for Elasticsearch’s default language).
* **source**: The source code of the script.
* **params**: Parameters passed into a script.
* **painless**: The default scripting language used by Elasticsearch for writing inline scripts.

**7. Administration Keywords**

Useful for managing the cluster and indices.

* **\_cluster**: Used for cluster-level operations like health and stats.
* **\_cat**: Returns human-readable information (e.g., \_cat/indices for a list of indices).
* **\_nodes**: Provides information about nodes in the cluster.
* **\_flush**: Manually triggers a flush of the index (forces data to be written to disk).
* **\_refresh**: Manually refreshes an index, making recent changes searchable.
* **\_optimize**: Optimizes the index for better performance (deprecated).
* **\_aliases**: Manages index aliases.
* **\_forcemerge**: Forces segment merging for an index.

**8. Monitoring & Debugging Keywords**

These help you monitor and debug your Elasticsearch cluster.

* **\_stats**: Returns statistics for indices or the cluster.
* **\_segments**: Returns information about index segments.
* **\_recovery**: Shows the current recovery status for indices or shards.
* **\_pending\_tasks**: Shows tasks pending in the cluster.
* **\_tasks**: Shows current running tasks.
* **\_explain**: Explains the scoring of a document for a specific query.