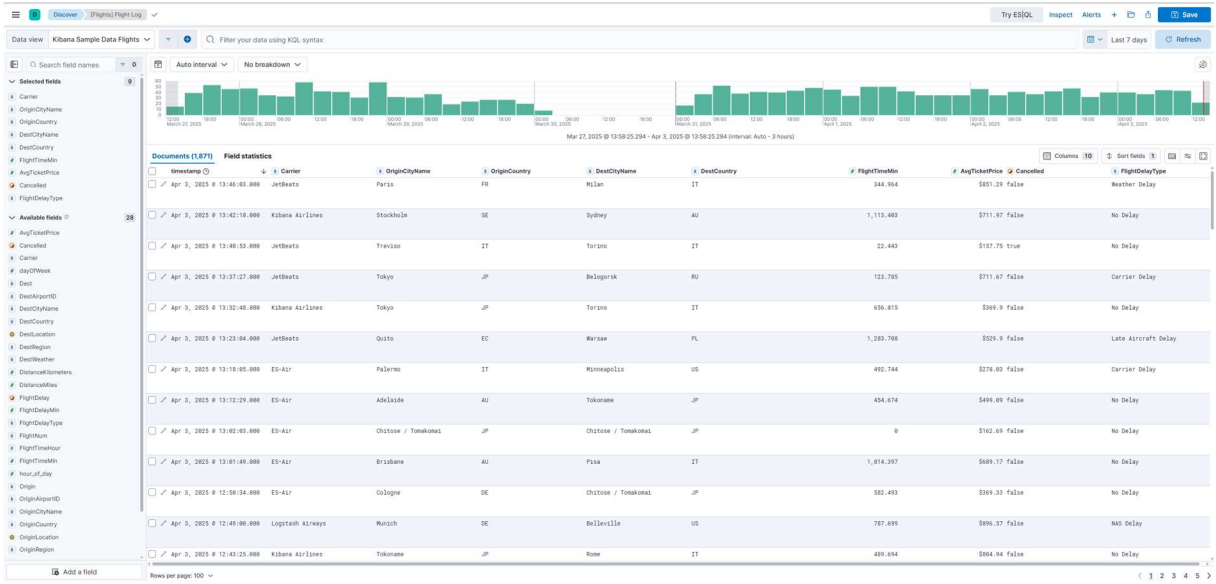
Demonstration of Using Elasticsearch

To demonstrate working with Elasticsearch and Kibana, we will use the built-in Sample flight data dataset, which simulates real flight data. The dataset contains fields such as the flight date, country of origin and destination, departure and arrival times, delay and other details. Using this example, we will show how these data can be analysed using the Elasticsearch and Kibana tools – including filtering, visualization and the creation of dashboards.

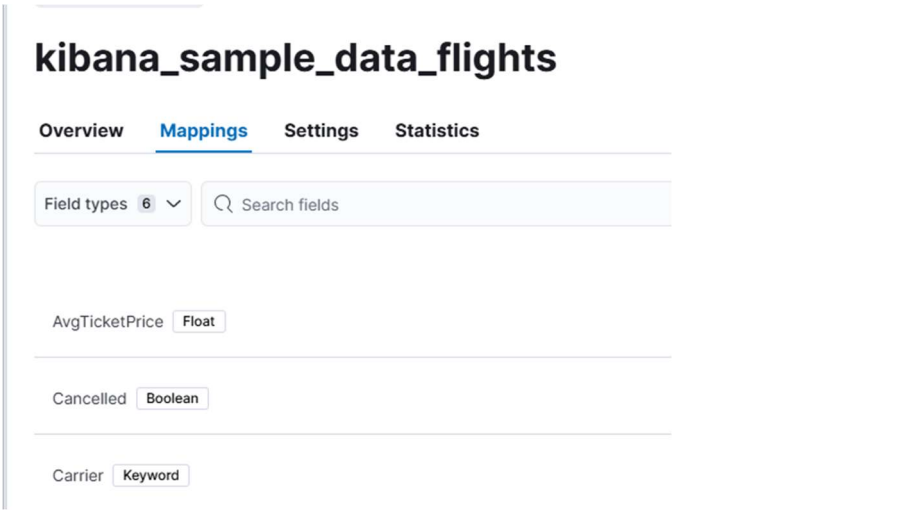
In the Discover section of Kibana, we can interactively browse all documents stored in Elasticsearch. The documents themselves (flight records) are displayed on the right-hand side, and on the left-hand side we see a list of the available fields. Each field can be used for filtering or for creating visualizations.



This figure shows the use of filtering – in the upper part, we enter conditions for selecting records. For example, we can display all flights departing from Abu Dhabi by entering the filter OriginCityName: "Abu Dhabi". This method of filtering is very practical when analysing larger amounts of data.

ndices and field types

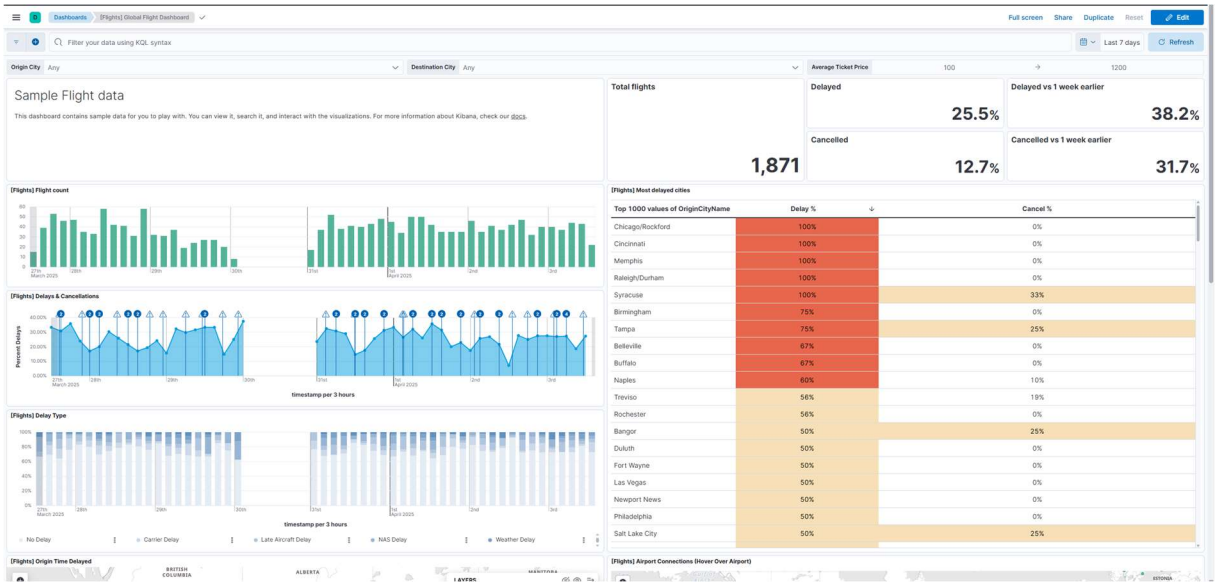
Elasticsearch stores data in indices, which represent the basic structure for storing and searching data. Each index contains documents in JSON format, which are similar to rows in a database table. Each document has fields of various types, for example text, keyword, integer, date, geo_point or boolean.



For efficient and fast searching, it is important to design the so-called mapping correctly, which defines the data types and the way the individual fields are processed. For example, the OriginAirportID field, which contains airport codes, should be of type keyword in order to achieve exact matching. Conversely, a field with a description or a city name is suitable as type text for the purposes of full-text search.

For text fields, a so-called analyser is used, which breaks the text down into individual tokens. For example, the sentence "Flight delayed by weather" is broken down into the words: flight, delayed, by, weather. This enables efficient full-text search. However, analysis is not suitable for exact values (e.g. airport codes), as it could lead to incorrect results.

In the Dashboards section, we can combine various visualizations in one place. Creating charts does not require programming – it is sufficient to select the visualization type (e.g. bar chart, pie chart, map, data table) and specify the data source and aggregations (e.g. number of flights by country, average delay by carrier). These visualizations can be filtered interactively – for example, clicking on a specific country on the map automatically updates all the other visualizations.

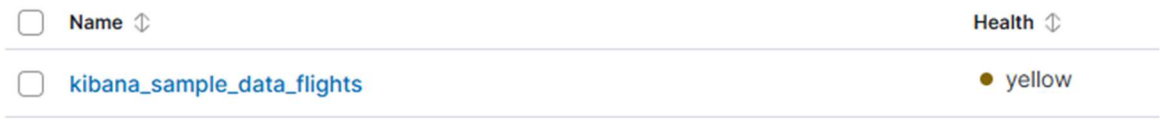


Elasticsearch makes it possible to work with aliases – alternative names for indices. An alias can point to one or more indices and helps with switching to a new index without the need to change queries in applications. This is useful, for example, when rotating indices by date (e.g. monthly indices with the alias flights_current).

Templates, in turn, define pre-prepared index structures. If we create indices automatically (e.g. by date), we can use a template to specify the mapping, shard settings, replicas and aliases. This ensures uniformity and optimization without the need for manual intervention for each new index.

In the Index Management section, we can monitor all the indices created. We can see their names, the number of documents, the size on disk, the status of the shards and the so-called health status:

- Green means that all primary and replica shards are available,
- Yellow means that a replica is missing (e.g. when we have only one node),
- Red indicates a problem – some primary shards are unavailable.



In addition to the status, we can also monitor:

- the size of each index,
- the number of documents,
- the number of shards (partitions) and replicas (copies for availability).

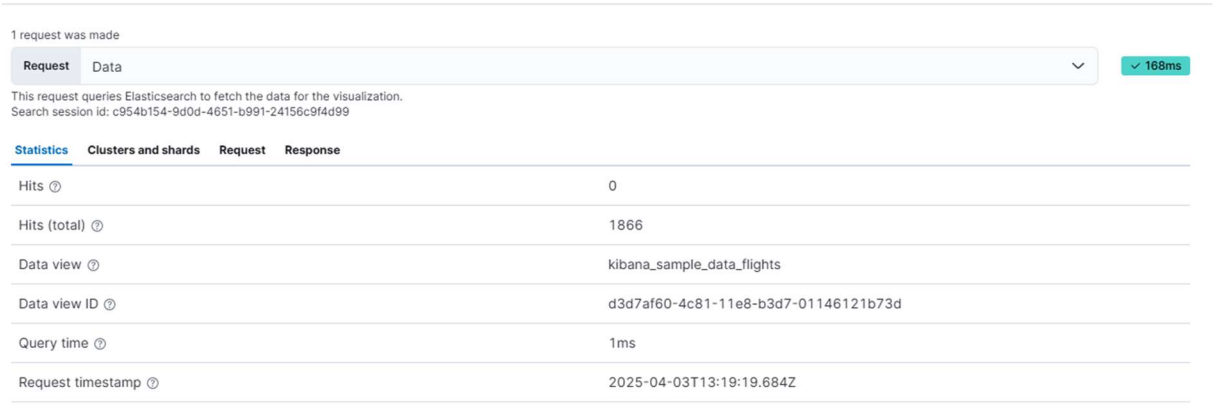
This information is important for performance planning, load estimation and scaling.

When created, each index is divided into primary shards, which contain the data themselves. For these, we can configure replicas – backup copies of the shards that serve to increase availability and at the same time enable parallel query processing. For example, an index with 3 primary shards and 1 replica will have a total of 6 shards (3 originals and 3 copies).

The correct configuration depends on:

- the expected volume of data (very small indices may have only 1 shard),
- the number of nodes in the cluster (more replicas make sense only with multiple servers),
- the need for speed vs. redundancy.

Using the Profile API, Elasticsearch makes it possible to find out exactly how a query was executed – which phases took the longest and which operations were performed in the background (e.g. filtering, scoring, aggregations). In Kibana, we can display this via the Inspect panel (available for visualizations or in the Discover section).



Here we obtain:

- the query execution time (in milliseconds),
- details of the individual processing phases,
- the filters used and their cost,
- the output JSON queries, which we can further modify and test.

This tool is excellent for:

- debugging complex queries with multiple conditions,
- finding so-called bottlenecks – for example, expensive aggregations,
- validating whether we are using indices and field types correctly.