COMPENSATION DETAILS SEARCH

In order to pass three schemas into single index , a common schema should be created. Based on given questions three schema works well with below schema

Schema for Compensation Data:

```
     Compensation  {
            "role": text,
            "salary": double,
            "currency": text,
            "city": text,
            "timestamp":
        }
```

MAPPING :

```
"role": text,keyword,
"salary": double,keyword,
"currency": text,keyword,
"city": text,keyword,
"timestamp": text,keyword
```

Justification for Chosen Mapping:

Role:"Text" is chosen for flexibility in case roles have variations. "Keyword" is included for exact matching and aggregation.

Salary: "Double" is chosen to handle numerical values. "Keyword" is included for exact matching, if needed.

Currency: "Text" is suitable to handle currency codes or names. "Keyword" is included for exact matching.

City: "Text" is chosen to handle city names. "Keyword" is included for exact matching.

Timestamp: "string" is chosen for timestamp data as date is stored in string format and "Keyword" is included for exact matching.

CSV Data Overview:

CSV1: Includes salary and both known and unknown currency fields.

CSV2 and CSV3: Contain base salary, joining bonus, stocks, and signing bonus as separate fields, with default and dynamically placed currency.

Data Parsing Strategy:

Given the diverse formats in the three datasets, a standardized approach is needed.

For Instance, salary is provided differently in three datasets

Different ways to parse salary:

- manually editing csv
- ask machine to solve using algorithm
- using NLP to identify salary

I preferred second way (algorithmic parsing), Python scripting is utilized to preprocess and standardize the data before indexing it into Elasticsearch.

After Parsing I have inserted Documents using **Python API**

Kibana Exploration:

To kickstart the process, the Kibana Console is utilized for initial interactions. Queries are rn to retrieve and analyse the indexed data, facilitating a better understanding of its structure and content.

Elasticsearch API Integration with Spring Boot: A Spring Boot application is developed to call the Elasticsearch API. This enables the execution of queries involving filtering and sorting to extract specific information as needed.
Tried querying sparse fields

==
"The essence of the task lies in transforming the provided CSVs into a uniform format"
==

Improvement:
Need to perform conversion rate for specific currency when average salary needs to be computed