validation

https://stackoverflow.com/questions/30428639/check-database-schema-matches-sqlalchemy-models-on-application-startup

AJOUT LETTRES

NOM. Date. Demandes. [TSR, ESP, REP]

Exemple REP: { —inspection— name: “INSSN-BDX-2020-0007”, text: “. . . .”, theme: “. . . ”, date: date,

```
---exploitant---
site_name: "Ionisos Marseille",
interlocutor_name: "Ionisos",
--> suggestions vieux noms
interlocutor_city: "Marseille",
// nature: ["CNPE", "CPO"] / "Usine",
identifiers: [numéro d'INB, SIRET]
sectors: [INB, NPX, REP, LUDD, TSR, ESP, Industrie, Médical]
--> INB/NPX filtrés des champs possibles
--> suggestions: tsr / transport, ESP equip.sous.press., industrie/industriel etc ...
domains: [OA, LA, Vétérinaire, Médical, Recherche, Environnement]
// natures: [Gamma graphie, radiopro, ...]

---entite---
pilot_entity_name: Marseille / DRC / ...
resp_entity_name: Marseille / DRC (la plus intéressante!)

---demandes---
----> fuzzy & exact a priori.
category_a_demands_nb: 6,
category_a_demands_topics: ["incendie", "..."]
category_a_demands_subtopics: ["incendie", "..."]
    --> répliquer dans la liste augmente le score :)

category_b_demands_nb: 6,
category_b_demands_topics: ["incendie", "..."]
category_b_demands_subtopics: ["incendie", "..."]
    --> répliquer dans la liste augmente le score :)

observations_topics: ["incendie",...] (par phrase)
```

```
    --> à requêter « sans répétition »
synthese_topics: ["incendie", ...] (par phrase)
    --> à requêter « sans répétition »

topics: ["", ""] (avec répétition par phrase)

---matériel---
equipments: []
```

}

INDEX Demandes / Synthèse / Observations

{ text: "...", inspection: "...", letter_id:,

```
topics: []
subtopics: []
resp_entity_name: Marseille / DRC (la plus intéressante!)

date: date,
site_name: "Ionisos Marseille",
interlocutor_city: "Marseille",
```

}

Liens

https://www.elastic.co/guide/en/elasticsearch/reference/6.8/suggester-context.html https://www.elastic.co/guide/en/elasticsearch/reference/6.8/search-suggesters-completion.html https://www.elastic.co/guide/en/elasticsearch/reference/6.8/search-suggesters-phrase.html https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-simple-query-string-query.html https://jolicode.com/blog/construire-un-bon-analyzer-francais-pour-elasticsearch https://www.elastic.co/fr/blog/elastic-app-search-query-suggestion-api-now-available https://swiftype.com/documentation/app-search/guides/query-suggestions

https://fr.slideshare.net/linagora/prsentation-elasticsearch-linagora-open-source

+basic query string

FIRST MAPPING TEST

```
PUT /letters
{
  "settings": {
    "analysis": {
      "filter": {
        "french_elision": {
          "type": "elision",
          "articles_case": true,
          "articles": ["l", "m", "t", "qu", "n", "s", "j", "d", "c", "jusqu", "quoiqu", "lor
        },
        "french_stop": {
          "type":       "stop",
          "stopwords":  "_french_"
        },
        "french_synonym": {
          "type": "synonym",
          "expand": true,
          "synonyms": [
            "salade, laitue",
            "mayo, mayonnaise",
            "grille, toaste",
        "Orano, Areva"
          ]
        },
        "french_stemmer": {
          "type": "stemmer",
          "language": "light_french"
        }
      },
      "analyzer": {
        "french_heavy": {
          "tokenizer": "standard",
          "filter": [
            "french_elision",
            "lowercase",
            "french_stop",
            "french_synonym",
            "french_stemmer"
          ]
        },
        "french_light": {
          "tokenizer": "standard",
          "filter": [
            "french_elision",
            "lowercase"
          ]
```

```
      }
    }
  }
},
"mappings":{
    "properties": {
      "autocomplete": {
        "type": "search_as_you_type",
        "analyzer": "french_light"
      },
      "name": {
        "type": "search_as_you_type",
        "fields": {
          "keyword": {
            "type": "keyword"
          }
        }
      },
      "content": {
        "type": "text",
        "analyzer": "french_light",
        "fields": {
          "stemmed": {
            "type": "text",
            "analyzer": "french_heavy"
          }
        },
        "copy_to": "autocomplete"
      },
      "theme": {
        "type": "text",
        "norms": false,
        "analyzer": "french_light",
        "fields": {
          "keyword": {
            "type": "keyword"
          },
          "completion": {
            "type": "completion"
          }
        },
        "copy_to": "autocomplete"
      },
      "date": {
        "type": "date",
        "format": "dd/MM/yyyy||yyyy-MM-dd"
```

```
},
"site_name": {
  "type": "text",
  "analyzer": "french_light",
  "fields": {
    "suggest": {
      "type": "completion"
    },
    "keyword": {
      "type": "keyword"
    }
  }
},
"interlocutor_name": {
  "type": "text",
  "analyzer": "french_light",
  "fields": {
    "suggest": {
      "type": "completion"
    },
    "keyword": {
      "type": "keyword"
    }
  }
},
"interlocutor_city": {
  "type": "text",
  "analyzer": "french_light",
  "fields": {
    "suggest": {
      "type": "completion"
    },
    "keyword": {
      "type": "keyword"
    }
  }
},
"identifiers": {
  "type": "long"
},
"sectors": {
  "type": "keyword",
  "copy_to": "autocomplete"
},
"domains": {
  "type": "keyword",
```

```
          "copy_to": "autocomplete"
        },
        "pilot_entity": {
          "type": "keyword"
        },
        "resp_entity": {
          "type": "keyword"
        },
        "demands_a": {
          "type": "integer"
        },
        "demands_b": {
          "type": "integer"
        },
        "observations_topics": {
          "type": "keyword"
        },
        "synthesis_topics": {
          "type": "keyword"
        },
        "demands_a_topics": {
          "type": "keyword"
        },
        "demands_b_topics": {
          "type": "keyword"
        },
        "topics": {
          "type": "keyword",
          "copy_to": "autocomplete"
        },
        "equipments": {
          "type": "keyword",
          "copy_to": "autocomplete",
          "fields": {
            "completion": {
              "type": "completion"
            }
          }
        }
      }
    }
}
```

Queries

Autocomplete « did you mean »

```
POST /letters/_search
{
  "suggest": {
    "dym": {
      "text": "arrêt de racteur",
      "phrase": {
        "field": "autocomplete",
        "size": 3,
        "gram_size": 3,
        "highlight": {
          "pre_tag": "<em>",
          "post_tag": "</em>"
        }
      }
    }
  }
}
```

Autocomplete « as you type »

1. Prendre le dernier mot de la requête
2. Faire une recherche de complétion sur les champs

```
POST /letters/_search
{
  "_source": "",
  "suggest": {
"text": "lil",
"city": {
  "completion": {
    "field": "interlocutor_city.suggest",
    "skip_duplicates": true,
    "size": 2
  }
},
"equipment": {
  "completion": {
    "field": "equipments.completion",
    "skip_duplicates": true,
    "size": 2
  }
}
```

```
      }
    }
```

Real Query: fulltext part

```
POST /letters/_search
{
  "_source": ["name", "theme", "content"],
  "query": {
    "multi_match": {
      "query": "arrêt de réacteur",
      "type": "bool_prefix",
      "fields": ["content", "content.stemmed"]
    }
  },
  "highlight": {
    "pre_tags": ["<em>"],
    "post_tags": ["</em>"],
    "fields": {
      "content": {}
    }
  },
  "aggs": {
    "equipments": {
      "significant_terms": {
        "field": "equipments.keyword" / equipments.completion??
      }
    }, ...
  }
}
```

Remarque: il faut ajouter les filtres qui vont par dessus!

Idées ajouter « simple_query_string »

Cela permet d'utiliser « + » / « | » / « " " » ce qui est VRAIMENT utile ! On met cela à la place de multi_match et hop c'est parti ;).

Ajouter BERT

https://xplordat.com/2019/10/28/semantics-at-scale-bert-elasticsearch/

Construire des variantes

- Où la query cherche aussi dans les champs de méta-données (via autocomplete en fait)

Tester le modèle version [JE SAIS CE QUE JE VEUX]

Pour N,K fixés 1. Sélectionner un document aléatoirement 2. Extraire les informations suivantes - date - interlocuteur - ville - thème - équipements - entité responsable / entité pilote - secteurs / domaines 3. Sélectionner N valeurs à chercher, et faire au plus K erreurs. 4. Produire l'histogramme des scores et placer notre document dans cet histogramme.

Tester le modèle version [Je cherche un thème]

1. Sélectionner un document aléatoirement
2. Extraire les informations suivantes
   - date
   - interlocuteur
   - ville
   - thème
   - équipements
   - entité responsable / entité pilote
   - secteurs / domaines
3. Sélectionner N valeurs à chercher, et faire au plus K erreurs.
4. Produire l'histogramme des scores et placer notre document dans cet histogramme.
5. Effectuer cette même recherche en plein texte
6. Effectuer cette même recherche dans le champ "autocomplete"

Exemples de lettres


```
{
    "name": "INSSN-BDX-2020-0004",
    "content": "Le réacteur 2 du CNPE du Blayais a été arrêté le 9 mai 2020 pour maintenance
    "theme":  "Arrêt de réacteur",
    "date": "2020-07-23",
    "interlocutor": ["EDF", "CNPE du Blayas"],
    "identifiers": [789988765447766, 102, 104],
    "sectors": ["Environnement", "ESP", "INB", "REP"],
    "demands": 4,
    "topics": ["générateur vapeur", "FOH", "environnement"],
        "equipments": ["2 RCP 003 MO", "PA 100343", "RPVOT", "MSR", "ECE", "GV", "hydrazine"
}
```


```
PUT /letters/_doc/2
{
    "name": "INSNP-LIL-2020-0464",
    "content": "L'ASN a conduit, le 25 novembre 2020,Source du renvoi introuvable. une inspe
    "theme":  "Radioprotection",
    "date": "2020-11-25",
```

```
    "interlocutor": ["Hôpital Saint Philibert", "Hôpitaux de l'Institut Catholique de Lille"
    "identifiers": [457876543456789],
    "sectors": ["Médical", "NPX"],
    "demands": 5,
    "topics": ["radioprotection", "intervetions radioguidées", "zonage", "FOH", "générateurs
  "equipments": ["générateur X", "ASL", "rayonnement X", "SIEMENS", "ARCADIS", "AVANTIC", "V
}
```


Index des demandes

PUT /demands { “settings”: { “analysis”: { “filter”: { “french_elision”: { “type”: “elision”, “articles_case”: true, “articles”: [“l”, “m”, “t”, “qu”, “n”, “s”, “j”, “d”, “c”, “jusqu”, “quoiqu”, “lorsqu”, “puisqu”] }, “french_stop”: { “type”: “stop”, “stopwords”: “*french*” }, “french_synonym”: { “type”: “synonym”, “expand”: true, “synonyms”: [“salade, laitue”, “mayo, mayonnaise”, “grille, toaste”] }, “french_stemmer”: { “type”: “stemmer”, “language”: “light_french” } }, “analyzer”: { “french_heavy”: { “tokenizer”: “standard”, “filter”: [“french_elision”, “lowercase”, “french_stop”, “french_synonym”, “french_stemmer”] }, “french_light”: { “tokenizer”: “standard”, “filter”: [“french_elision”, “lowercase”] } } } }, “mappings”:{ “properties”: { “autocomplete”: { “type”: “search_as_you_type”, “analyzer”: “french_light” }, “name”: { “type”: “keyword” }, “content”: { “type”: “text”, “analyzer”: “french_light”, “fields”: { “stemmed”: { “type”: “text”, “analyzer”: “french_heavy” } }, “copy_to”: “autocomplete” }, “theme”: { “type”: “text”, “norms”: false, “analyzer”: “french_light”, “fields”: { “keyword”: { “type”: “keyword” } }, “copy_to”: “autocomplete” }, “date”: { “type”: “date”, “format”: “dd/MM/yyyy||yyyy-MM-dd” }, “site_name”: { “type”: “text”, “analyzer”: “french_light”, “fields”: { “suggest”: { “type”: “completion” }, “keyword”: { “type”: “keyword” } } }, “interlocutor_name”: { “type”: “text”, “analyzer”: “french_light”, “fields”: { “suggest”: { “type”: “completion” }, “keyword”: { “type”: “keyword” } } }, “interlocutor_city”: { “type”: “text”, “analyzer”: “french_light”, “fields”: { “suggest”: { “type”: “completion” }, “keyword”: { “type”: “keyword” } } }, “sectors”: { “type”: “keyword”, “copy_to”: “autocomplete” }, “domains”: { “type”: “keyword”, “copy_to”: “autocomplete” }, “pilot_entity”: { “type”: “keyword” }, “resp_entity”: { “type”: “keyword” }, “demand_type”: { “type”: “keyword” }, “topics”: { “type”: “keyword”, “copy_to”: “autocomplete” } } } } }