Requêter dans Elasticsearch

Elasticsearch propose plusieurs **langages de requête** pour interagir avec les données.

Certains s'utilisent via **API** (programmation), d'autres directement dans **Kibana** (interface graphique).

---

1. Query DSL

Langage principal, en JSON.
Permet les **requêtes complexes** : recherche plein texte, filtres, agrégations, tris, etc.

```
POST /users/_search
{
  "query": {
    "term": { "status": "active" }
  }
}
```

Usage : tout type de recherche avancée (le plus complet et flexible).

---

2. ES|QL (Elasticsearch Query Language)

Langage introduit en **version 8.11**.
Fonctionne en **pipeline (|)**, comme dans un terminal, pour filtrer, transformer et analyser les données.

```
POST /_query
{
  "query": "FROM products | SORT price DESC | LIMIT 5"
}
```

Usage : requêtes interactives, visualisations et analyses dans Kibana.

ES|QL tend à remplacer **KQL** et **Lucene** pour les requêtes dans Kibana.

ES|QL est l'**interface d'analyse** privilégiée dans Kibana (Discover et Lens) pour tout ce qui dépasse le simple filtrage.

---

3. EQL (Event Query Language)

Conçu pour les **données temporelles** (champ `@timestamp` obligatoire).
Permet de rechercher des **séquences d'événements** dans le temps.

```
POST /logs/_eql/search
{
  "query": "process where event.action == 'login' and @timestamp >
'2025-10-01T00:00:00Z'"
}
```

Usage : détection de menaces et analyses de sécurité (threat hunting).

⚠ Maintenu, mais réservé aux **cas de sécurité** (Elastic Security). Ne pas l'utiliser pour les données classiques.

---

4. Elasticsearch SQL

Permet d'utiliser la **syntaxe SQL** classique sur les données Elasticsearch.
Compatible avec les outils BI via **JDBC/ODBC**.

```
POST /_sql
{
  "query": "SELECT name, created_at FROM users ORDER BY created_at DESC LIMIT 10"
}
```

Usage : reporting, tableaux de bord, intégration BI.

Langage qui sert de passerelle pour les habitués du SQL et lier à des outils de reporting BI.

---

5. KQL (Kibana Query Language)

Langage simple pour **filtrer** et **rechercher** dans Kibana.
Pas de syntaxe JSON. Exemple :

```
price > 100
```

Usage : filtres rapides dans les visualisations Kibana.

> Surtout pour le **filtrage simple** dans Kibana, sinon passer sur ES|QL si le filtrage devient plus complexe.

---

6. Lucene syntax

Syntaxe textuelle d'origine, basée sur **Apache Lucene**.
Toujours disponible pour des **recherches simples**.

```
category:book
```

Usage : recherche basique ou tests rapides.

> ⚠ Toujours fonctionnel, mais **remplacé par KQL** dans Kibana. Réservé aux tests ou compatibilité historique.

Résumé :

Langage	Type	Complexité	Usage principal	Endpoint
Query DSL	JSON	Élevée	Requêtes avancées	`_search`
ES	QL	Pipelined	Moyenne	Analyse et exploration
EQL	Séquentiel	Moyenne	Sécurité / événements	`_eql`
SQL	SQL standard	Moyenne	BI et reporting	`_sql`

Langage	Type	Complexité	Usage principal	Endpoint
KQL	Textuel	Simple	Filtres Kibana	N/A
Lucene	Textuel	Simple	Recherches rapides	`_search`