



ELASTICSEARCH

Base de Données Orientée Vecteur & Moteur de Recherche Distribué

3^e année en Génie Informatique GLID

Matière : Bases de données avancées

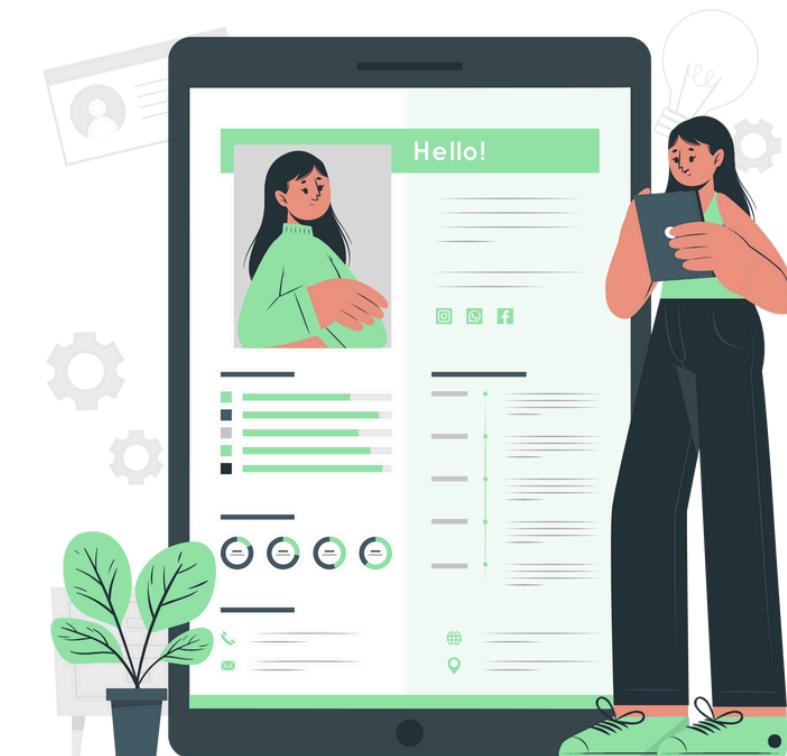
Présenté Par :

Ines Tmimi



Plan

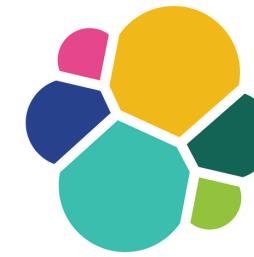
- 1. INTRODUCTION & CONTEXTE**
- 2. ARCHITECTURE & CONCEPTS TECHNIQUES**
- 3. CAS USAGE & AVANTAGES**
- 4. CONCLUSION**





Introduction

QU'EST-CE QU'ELASTICSEARCH ?



Elasticsearch est un **moteur de recherche et d'analyse distribué**, open-source, construit sur Apache Lucene.

BASE DE DONNÉES NoSQL

Stockage orienté documents (format JSON)
Schéma flexible et dynamique
Pas de relations complexes



RECHERCHE EN TEMPS RÉEL

Indexation quasi-instantanée (<1 seconde)
Recherche full-text ultra-rapide
Résultats pertinents et scorés



ANALYSE À GRANDE ÉCHELLE

Agrégations complexes en temps réel
Traitement de pétaoctets de données
Visualisations interactives

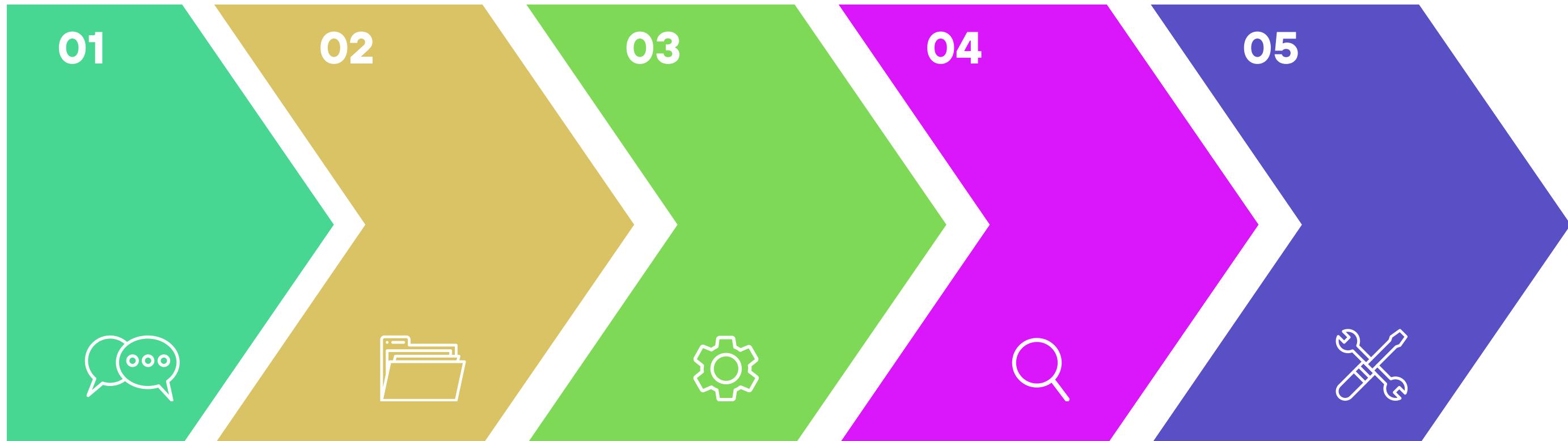


ARCHITECTURE DISTRIBUÉE

Scalabilité horizontale automatique
Haute disponibilité
RéPLICATION et failover automatiques



ÉVOLUTION



2010:
Création par Shay Banon

2012:
v1.0 - Adoption massive

2015:
Elastic Stack (ELK)

2020:
Machine Learning intégré

2021: Changement de licence (SSPL)

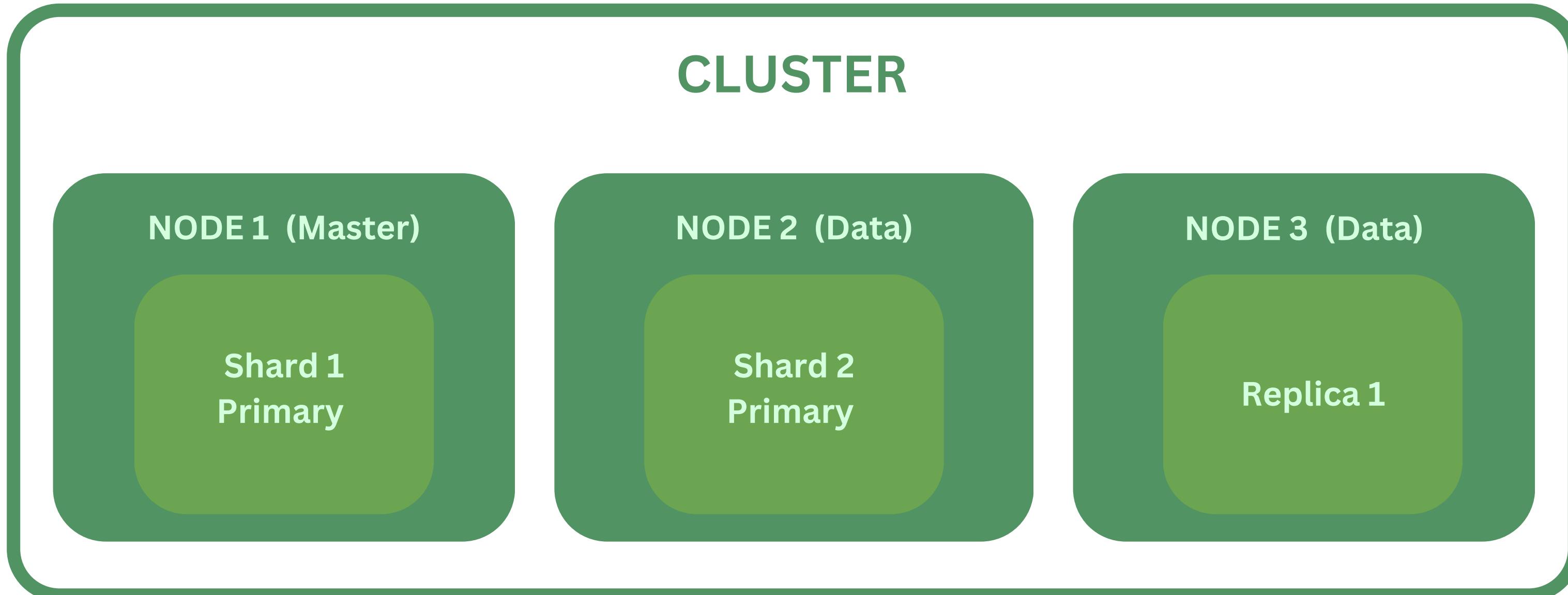
Pourquoi choisir Elasticsearch ?

Critère	Elasticsearch	PostgreSQL	MongoDB Atlas
Recherche full-text	Excellent	Moyen	Moyen
Recherche vectorielle	Natif	Extension	Récent
Scalabilité	Horizontale	Verticale	Horizontale
Écosystème	ELK Stack	SQL standard	Atlas
Coût	Open Source	Open Source	Freemium



Architecture & Concepts Clés

Architecture



RECHERCHE VECTORIELLE & INDEXATION INVERSÉE

PROBLÈME TRADITIONNEL :

Comment trouver "performance" dans 10 millions de documents ?

Approche naïve : Scanner tous les documents (trop lent !)

SOLUTION ELASTICSEARCH : INVERTED INDEX

Document	Termes (tokens)	Index inversé (liste des documents)
Doc1 : "Laptop haute performance"	laptop	[Doc1]
	haute	[Doc1, Doc3]
	performance	[Doc1, Doc2]
Doc2 : "Performance web"	performance	[Doc1, Doc2]
	web	[Doc2]
Doc3 : "PC haute gamme"	pc	[Doc3]
	haute	[Doc1, Doc3]
	gamme	[Doc3]

VECTOR SEARCH / k-NN

Évolution moderne : la recherche sémantique

Exemple de requête : "ordinateur puissant"

1. Étape 1 : Vectorisation

Texte : → "ordinateur puissant"

Transformé par un modèle ML en vecteur numérique :

→ [0.23, 0.87, 0.45, 0.12, 0.98, ...]

(768 dimensions)

2. Étape 2 : Comparaison vectorielle

Document Contenu Similarité sémantique

Doc1 "Laptop performance" 95%

Doc2 "Clavier sans fil" 12%

Doc3 "PC gaming puissant" 97%

– Le système compare le vecteur de la requête avec les vecteurs des documents indexés.

Algorithme HNSW (Hierarchical Navigable Small World) :

Elasticsearch utilise HNSW pour une recherche approximative rapide dans les espaces de haute dimension

Métriques de similarité supportées :

cosine : Mesure l'angle entre vecteurs (recommandé pour le texte)

dot_product : Produit scalaire (vecteurs normalisés)

l2_norm : Distance euclidienne

VECTOR SEARCH / k-NN

3. Étape 3 : k-NN (k-Nearest Neighbors)

L'algorithme k-NN sélectionne les k documents les plus proches dans l'espace vectoriel.

Avantages du Vector Search

- Comprend l'intention de l'utilisateur
- Recherche basée sur la similarité sémantique
- Fonctionne en multilingue naturellement
- S'applique à : textes, images, sons, vidéos

Comparaison : Recherche Classique vs Recherche Vectorielle

Recherche Classique	Recherche Vectorielle
Requête : "laptop performance"	Requête : "ordinateur puissant"
Correspondance exacte des mots	Compréhension du sens de la phrase
Résultats trouvés : 1 250	Résultats trouvés : 8 430
Pertinence moyenne : 65%	Pertinence moyenne : 94%

Elastic Stack (ELK) - L'Écosystème Comple





Elasticsearch

- Stocke et recherche les données rapidement (texte, filtres, agrégations)
- Supporte la recherche vectorielle et sémantique
- Distribué sur plusieurs nœuds pour haute performance

Logstash

- Collecte et transforme les données depuis différentes sources
- Nettoie et structure les données avant insertion dans Elasticsearch
- Compatible avec de nombreux plugins (fichiers, API, Kafka...)



Kibana

- Crée dashboards, graphiques et tableaux interactifs
- Permet recherche avancée et exploration des données en temps réel
- Supervision, monitoring et alerting grâce aux modules préconfigurés



Cas d'Usage & Avantages

Cas d'usage concrets

1. E-commerce – Amazon, eBay

Recherche de produits en temps réel

Suggestions automatiques et personnalisées



2. Observabilité – Netflix, Uber

Monitoring des logs et métriques

Détection d'anomalies en temps réel

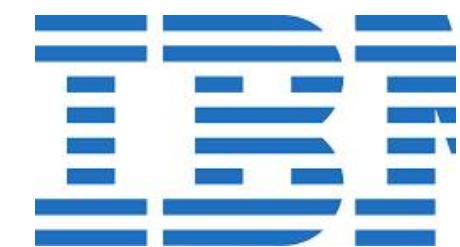


3. Sécurité – Cisco, IBM

SIEM (Security Information and Event Management)

Détection d'intrusions et menaces

Corrélation d'événements de sécurité



4. Applications – GitHub, Stack Overflow

Recherche rapide dans le code source

Full-text search et documentation



Démonstration – Exemple Pratique

Lien Github

Imaginons que nous gérons une plateforme e-commerce.
Nous avons 30 logs générés sur la dernière heure.
Un client signale un problème de paiement.
Comment trouver la cause en quelques secondes ?
C'est ce que je vais vous montrer maintenant.



Démonstration – Exemple Pratique

Lien Github

Premièrement, vérifions combien de logs nous avons.

```

        "type" : "application-logs"
    }
}
]

ondre.. ines@hpines:~$ # Terminal
curl -s "localhost:9200/app-logs-*/_count"
{"count":30,"_shards": {"total":1,"successful":1,"skipped":0,"failed":0}}ines@hpines:~$ |

```

"Je lance une recherche sur le niveau ERROR."

```

curl -X GET "localhost:9200/app-logs-*/_search?pretty" \
-H 'Content-Type: application/json' -d'
{
  "query": {
    "match": {"level": "ERROR"}
  },
  "size": 3
}'

```

```

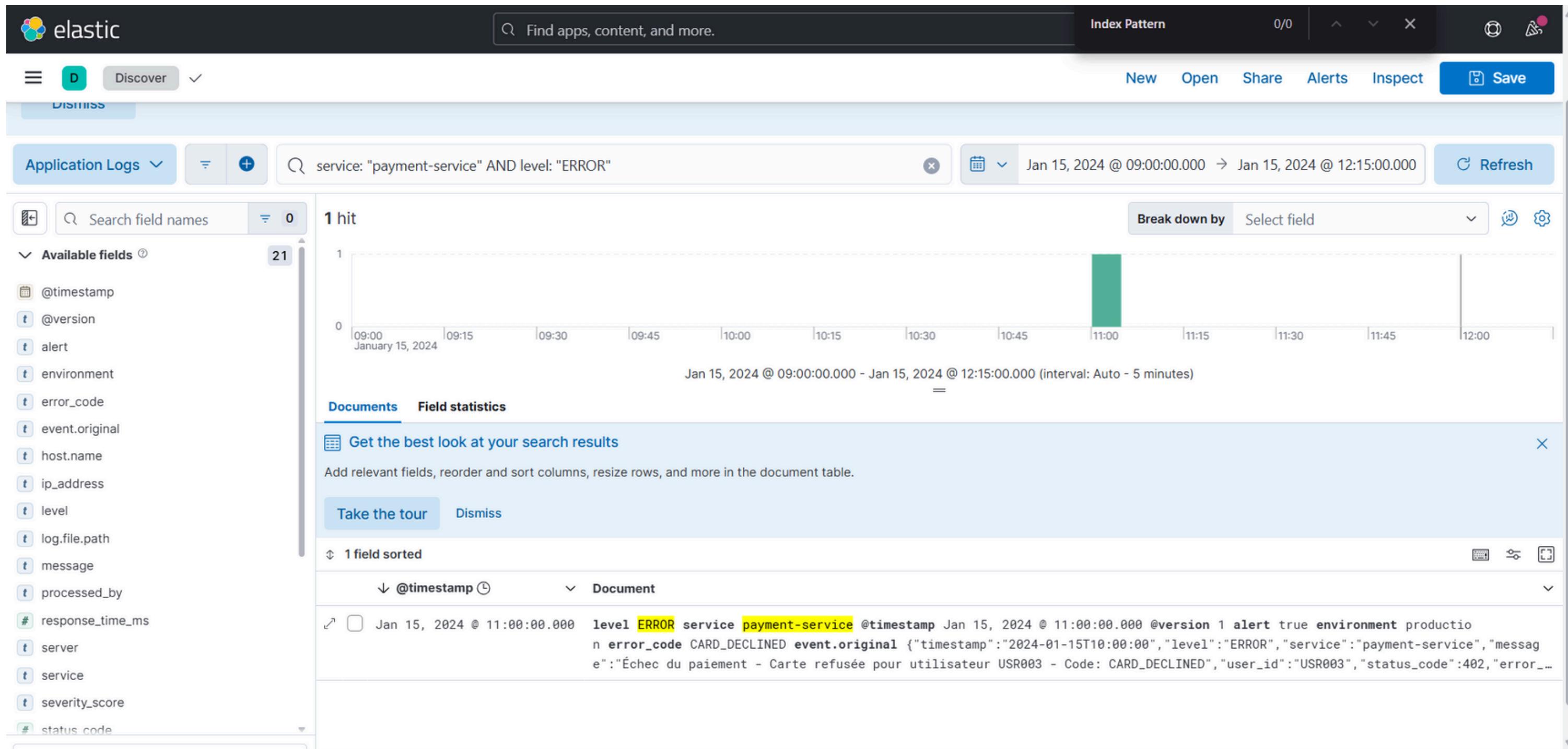
{
  "_index" : "app-logs-2024.01.15",
  "_id" : "LBNKFJsBHp8_eEQ-JIIf",
  "_score" : 1.293921,
  "_source" : {
    "timestamp" : "2024-01-15T10:00:00",
    "message" : "Échec du paiement - Carte refusée pour utilisateur USR003 - Code: CARD_DECLINED",
    "error_code" : "CARD_DECLINED",
    "event" : {
      "original" : "{\"timestamp\":\"2024-01-15T10:00:00\", \"level\":\"ERROR\", \"service\":\"payment-service\", \"message\":\"Échec du paiement - Carte refusée pour utilisateur USR003 - Code: CARD_DECLINED\", \"user_id\":\"USR003\", \"status_code\":402, \"error_code\":\"CARD_DECLINED\", \"server\":\"prod-03\"}"
    },
    "host" : {
      "name" : "4d45daa7ca09"
    },
    "log" : {
      "file" : {
        "path" : "/var/log/application/application.log"
      }
    },
    "alert" : "true",
    "status_code" : 402,
    "processed_by" : "logstash",
    "@timestamp" : "2024-01-15T10:00:00.000Z",
    "service" : "payment-service",
    "user_id" : "USR003",
    "server" : "prod-03",
    "level" : "ERROR"
  }
}

```

Démonstration – Exemple Pratique

Lien Github

Kibana - Interface Visuelle



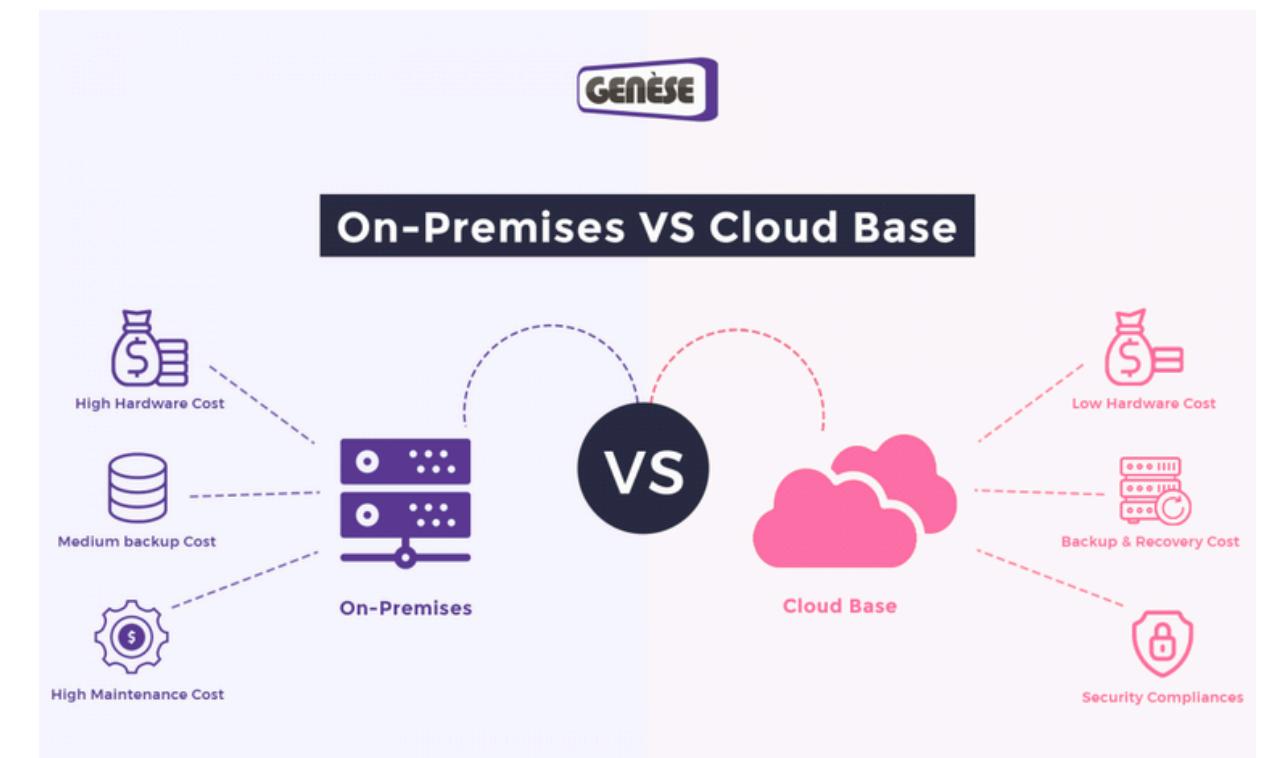
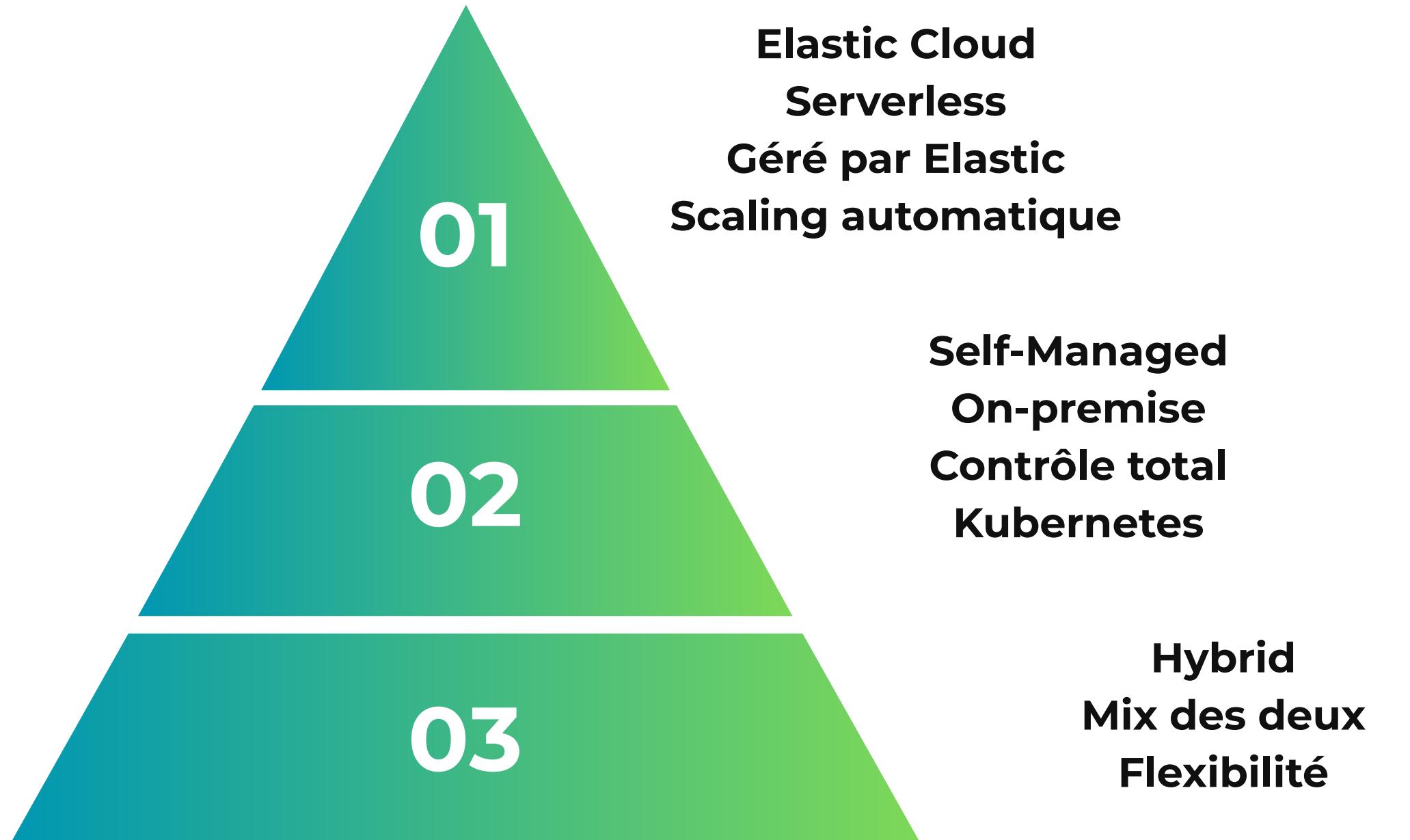
Limites et défis

Limite	Description
Consommation mémoire	Nécessite beaucoup de RAM (recommandé : 50% heap).
Complexité opérationnelle	Configuration et maintenance de clusters complexes.
Pas de transactions ACID	Non adapté aux opérations transactionnelles.
Courbe d'apprentissage	DSL de requête spécifique à maîtriser.
Coût en production	Licences commerciales pour certaines fonctionnalités (ML, sécurité avancée).
Cohérence éventuelle	Near real-time, pas de lecture immédiate après écriture.

Avantages & Performance

- 
- 01 Scalabilité horizontale
 - 02 Recherche en temps réel
 - 03 RESTful API simple
 - 04 Multi-tenant

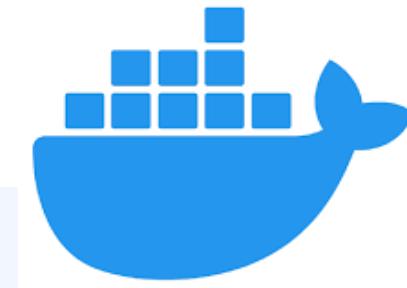
Déploiement & Options



Déploiement

Exemple Docker

```
docker run -d --name elasticsearch \
-p 9200:9200 -p 9300:9300 \
-e "discovery.type=single-node" \
docker.elastic.co/elasticsearch/elasticsearch:8.11.0
```



Écosystème & Intégrations





Conclusion

Conclusion



- Elasticsearch = Moteur de recherche distribué +
Base de données vectorielle
- Index inversé pour recherche full-text ultra-rapide
- k-NN avec HNSW pour recherche sémantique
- Elastic Stack (ELK) pour l'observabilité complète



Perspectives

- **Intégration croissante avec l'IA générative (RAG - Retrieval Augmented Generation)**
- **Recherche hybride lexicale + sémantique**
- **Elastic AI Assistant**



Documentation officielle :

- **Elasticsearch Guide : [Bibliographie](#) :**
- **Documentation officielle : [elastic.co/guide](#)**
- **GitHub repository : [github.com/elastic/elasticsearch](#)**
- **"Elasticsearch: The Definitive Guide" - O'Reilly**
- **Elastic Blog : [elastic.co/blog](#)**
- **Community Forums : [discuss.elastic.co](#)**
- **Vector Search : [Bibliographie](#) :**
- **Documentation officielle : [elastic.co/guide](#)**
- **Community Forums : [discuss.elastic.co](#)**

Ressources complémentaires :

- **"Elasticsearch: The Definitive Guide" - O'Reilly**
- **Elastic Blog : [elastic.co/blog](#)**
- **Community Forums : [discuss.elastic.co](#)**
- **Forums : [Bibliographie](#) :**
- **Documentation officielle : [elastic.co/guide](#)**
- **GitHub repository : [github.com/elastic/elasticsearch](#)**
- **"Elasticsearch: The Definitive Guide" - O'Reilly**
- **Elastic Blog : [elastic.co/blog](#)**
- **Community Forums : [discuss.elastic.co](#)**

Articles :

- **Elastic Blog - Vector Search : [Bibliographie](#) :**
- **Documentation officielle : [elastic.co/guide](#)**
- **GitHub repository : [github.com/elastic/elasticsearch](#)**
- **"Elasticsearch: The Definitive Guide" - O'Reilly**
- **Elastic Blog : [elastic.co/blog](#)**
- **Community Forums : [discuss.elastic.co](#)**



Merci Pour Votre Attention

Des questions ?
**Avez-vous déjà utilisé Elasticsearch dans vos
projets ?**