Data Engineering

Elasticsearch #2

Rollover

Rollover

- Neuer, zusätzlicher Index des gleichen Typs wird erstellt
- Sinnvoll bei *time series Daten*
- Wieso?

Rollover – Grund

- Time Series Daten wie Logs & Metriken mehren sich kontinuierlich
- Idr sind Time Series Daten immutable
- Idr sinkt die Zugriffsfrequenz der Daten mit ihrem Alter
- Hoher Ressourcenbedarf und Bedarf an regelmäßiger Löschung
- Zur Erinnerung: Shards werden in Elasticsearch zum Zeitpunkt der Erstellung des Indexes festgelegt => Für unendliches Wachstum ungeeignet

Rollover – Vorteile

- **Performance-Optimierung:** Rollierende Indizes ermöglichen hohe Ingest-Raten in einem aktiven Index auf leistungsstarken Hot-Nodes.
- **Effiziente Suche:** Ältere Indizes werden auf Warm-Nodes verschoben, um die Suchleistung zu optimieren.

Rollover – Vorteile

- **Kosteneffiziente Speicherung:** Ältere, selten genutzte Daten lassen sich auf Cold-Nodes ablegen, um Kosten zu senken.
- **Automatisierte Datenlöschung:** Daten können nach Ablauf der Aufbewahrungsfrist durch das Löschen ganzer Indizes entfernt werden.

Rollover – Anforderung

- Um die Vorteile eines Rollovers nutzen zu können werden folgende Funktionen benötigt:
  - Wiederverwendbare Templates für Mappings und Index-Einstellungen
  - Automatische Rollierung auf Basis definierter Kriterien (z.B. Alter oder Größe)
  - Konfiguration alter Indizes muss geändert werden können
  - Einheitlicher Endpunkt für Abfragen über mehrere Indizes eines Typen
  - Namenskonvention

Wildcards

- Elasticsearch besitzt wildcards „*“ für die Referenzierung von Indizes
- **Index-Muster „logs--“:** Dieses Muster erfasst alle Indizes, die mit „logs-“ beginnen, gefolgt von einer beliebigen Zeichenkette, dann einem „-“, und abschließend erneut von einer beliebigen Zeichenkette. Beispiel: Es matcht Indizes wie „logs-2024-01“, „logs-2020-03“
- Diese Patterns können auch in Abfragen verwendet werden

Index Templates

Index Templates – Einheitliche Index Struktur

- **Konsistenz**: Index Templates ermöglichen es, eine einheitliche Struktur (Mappings, Einstellungen) für alle Indizes eines bestimmten Typs festzulegen.
- **Automatisierung**: Neue Indizes erhalten automatisch das gewünschte Mapping, ohne dass manuelle Konfiguration nötig ist.

Index Templates – Einheitliche Index Struktur

- **Index Templates:**
  - **Direkte Anwendung:** Werden direkt auf eine oder mehrere Indizes angewendet.
  - **Inhalte:** Können Einstellungen, Mappings und Aliase direkt spezifizieren.
  - **Komposition:** Können eine Sammlung von Component Templates enthalten, um Konfigurationen wiederzuverwenden.
- **Component Templates:**
  - **Wiederverwendbare Bausteine:** Dienen zur Konfiguration von Mappings, Einstellungen und Aliasen.
  - **Indirekte Anwendung:** Werden nicht direkt auf Indizes angewendet.
  - **Verwendung:** Werden in Index Templates eingebunden, um modulare und konsistente Konfigurationen zu ermöglichen.

Index Templates

```
PUT _index_template/template_1
{
  "index_patterns": ["te*", "bar*"],
  "template": {
    "settings": {
      "number_of_shards": 1
    },
    "mappings": {
      "_source": {
        "enabled": true
      },
      "properties": {
        "host_name": {
          "type": "keyword"
        },
        "created_at": {
          "type": "date"
        }
      }
    },
    "aliases": {
      "mydata": { }
    }
  },
  "priority": 500,
  "composed_of": ["component_template1", "runtime_component_template"],
  "version": 3,
  "_meta": {
    "description": "my custom"
  }
}
```

Alias

Was ist ein Alias?

- Ein Alias ist ein alternativer Name für eine Gruppe von Indizes
- Ermöglicht die Verwendung eines Alias anstelle des tatsächlichen oder Indexnamens in den meisten Elasticsearch-APIs

Alias

```
POST _aliases
{
  "actions": [
    {
      "add": {
        "index": "logs-*",
        "alias": "logs"
      }
    }
  ]
}
```

Ingest Pipelines

Was sind Ingest Pipelines?

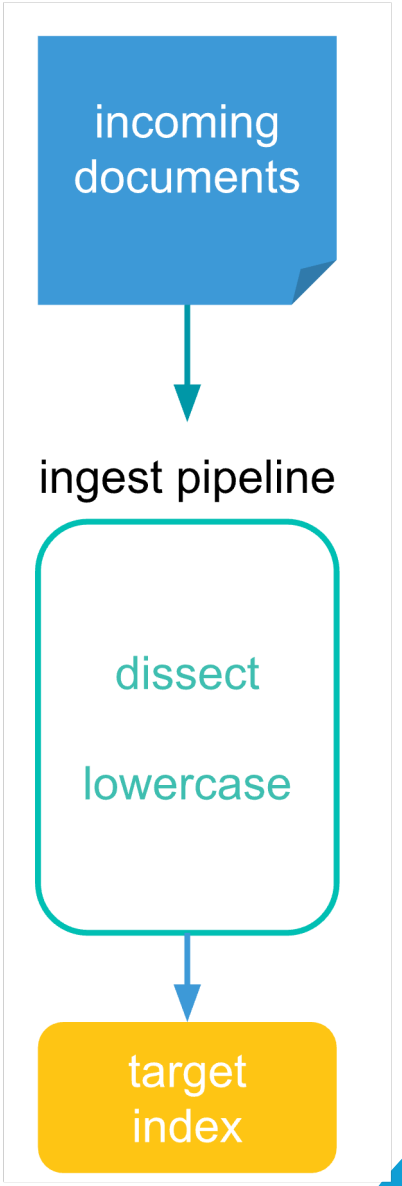
- Ermöglichen die Durchführung gängiger Transformationen an Daten vor dem Indexieren.
- **Anwendungsbeispiele:**
  - Entfernen von Feldern.
  - Extrahieren von Werten aus Text.
  - Anreichern von Daten mit zusätzlichen Informationen.

Was sind Ingest Pipelines?

- Ingest Pipelines werden auf Elastic Nodes selbst ausgeführt
- Dafür werden Nodes mit der Rolle „Ingest“ verwendet
- Der ELK Stack bietet das Tool „Logstash“, welches eine eigenständige Processing Pipeline darstellt
- Logstash bietet die gleiche Funktionalität und teilweise mehr als Ingest Pipelines

Was sind Ingest Pipelines?

- **Prozessoren:** Eine Pipeline besteht aus einer Reihe konfigurierbarer Aufgaben, die als Prozessoren bezeichnet werden
- **Sequenzielle Ausführung:** Jeder Prozessor führt nacheinander spezifische Änderungen an den eingehenden Dokumenten durch
- Nach Abschluss aller Prozessoren werden die transformierten Dokumente von Elasticsearch dem Index hinzugefügt.



Beispiel Ingest Pipeline

```
PUT _ingest/pipeline/my-pipeline
{
  "description": "My optional pipeline description",
  "processors": [
    {
      "set": {
        "description": "My optional processor description",
        "field": "my-long-field",
        "value": 10
      }
    },
    {
      "set": {
        "description": "Set 'my-boolean-field' to true",
        "field": "my-boolean-field",
        "value": true
      }
    },
    {
      "lowercase": {
        "field": "my-keyword-field"
      }
    }
  ]
}
```

Ingest Pipeline testen

```
POST _ingest/pipeline/my-pipeline/_simulate
{
  "docs": [
    {
      "_source": {
        "my-keyword-field": "FOO"
      }
    },
    {
      "_source": {
        "my-keyword-field": "BAR"
      }
    }
  ]
}
```

```
PUT my-data-stream/_bulk?pipeline=my-pipeline
{ "create":{ } }
{ "@timestamp": "2099-03-07T11:04:06.000Z", "my-keyword-field": "foo" }
{ "create":{ } }
{ "@timestamp": "2099-03-07T11:04:07.000Z", "my-keyword-field": "bar" }
```

Nutzung der Pipelines

- Pipelines können explizit in einem PUT Request gesetzt werden
- Oder als Standard Pipeline für einen oder mehr Indezes => Alle Dokumente die in einen Index geschrieben werden (Option: index.default_pipeline)

Pipelines: Prozessoren

- Eine Ingest Pipeline besteht aus einer Sequenz von Prozessoren
- Jeder Prozessor führt eine spezifische Aufgabe aus, wie z.B. Filtern, Transformieren oder Anreichern von Daten
- Jeder nachfolgende Prozessor baut auf dem Output des vorherigen auf
- Über 40 konfigurierbare Prozessoren: Elasticsearch bietet eine breite Palette an Prozessoren für unterschiedliche Aufgaben.

Grok Prozessor

- Der Grok-Prozessor extrahiert strukturierte Felder aus einem einzelnen Textfeld innerhalb eines Dokuments
- Basieren auf REGEX
- Anwendung: Log Files
  - Informationen liegen in semistrukturiertem Text vor
  - Informationen wie: IP Adresse des Clients, Zeitstempel, HTTP Statuscode, Endpunkt und viele mehr sind allerdings nur in strukturierter Form sinnvoll zu verarbeiten
  - Lösung GROK Patterns parsen Text und extrahieren Informationen

Grok Prozessor

```
POST _ingest/pipeline/_simulate
{
  "pipeline": {
    "description" : "...",
    "processors": [
      {
        "grok": {
          "field": "message",
          "patterns": ["%{IP:client} %{WORD:method} %{URIPATHPARAM:request} %{NUMBER:bytes:int}
           %{NUMBER:duration:double}"]
        }
      }
    ]
  },
  "docs":[
    {
      "_source": {
        "message": "55.3.244.1 GET /index.html 15824 0.043"
      }
    }
  ]
}
```

Grok Prozessor

- GROK-Prozessoren bestehen aus einem oder mehreren Patterns, die zu einer Regular Expression kompiliert werden
- Patterns setzten sich wiederum aus anderen Patterns zusammen; vergleichbar mit einer Funktion in der klassischen Programmierung
- Hinter einem Pattern verbirgt sich eine Regular Expression
- Elastic bietet eine Reihe vordefinierter Patterns für gängige Daten wie IP Adressen
- In den Patterns wird außerdem festgelegt in welchem Feld die extrahierten Daten gespeichert werden sollen

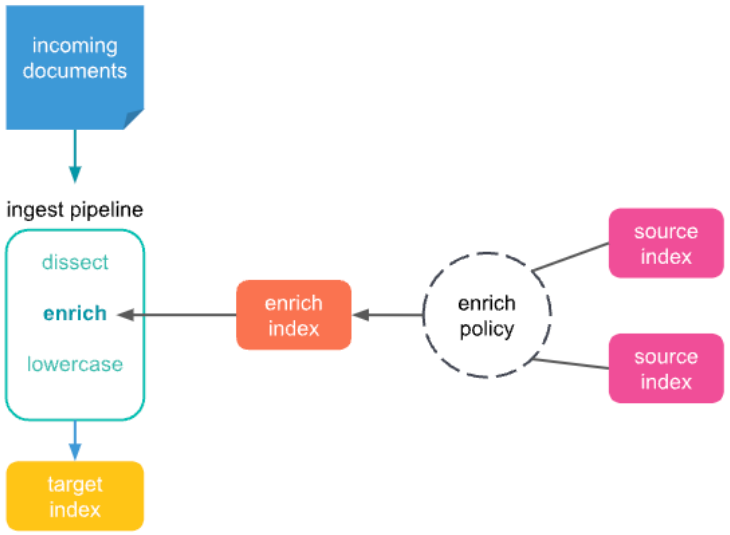
Custom Patterns

```
{
  "description" : "...",
  "processors": [
    {
      "grok": {
        "field": "message",
        "patterns": ["my %{FAVORITE_DOG:dog} is colored %{RGB:color}"],
        "pattern_definitions" : {
          "FAVORITE_DOG" : "beagle",
          "RGB" : "RED|GREEN|BLUE"
        }
      }
    }
  ]
}
```

Pattern
REGEX
Field

Enrich Processor

- Der Enrich-Prozessor fügt während des Ingest-Prozesses Daten aus bestehenden Indizes zu eingehenden Dokumenten hinzu
- **Anwendungsbeispiele:**
  - Basierend auf bekannten IP-Adressen können entsprechende Dienste oder Anbieter identifiziert werden.
  - Ergänzung von Bestellungen mit detaillierten Produktinformationen anhand von Produkt-IDs.
  - Anreicherung von Daten mit zusätzlichen Kontaktdetails basierend auf einer E-Mail-Adresse.
  - Ergänzung von Benutzerkoordinaten mit den zugehörigen Postleitzahlen.



Reindex

Reindex

- Kopiert die Daten eines Source Index in einen Destination Index
- **Sehr wichtige Funktion!**
- **Anwendungsfälle:**
  - Veränderung der Primaries und Replicas des Index (z.b. Shriking)
  - Zusammenführen mehrerer Indizes
  - Reindexierung auf Basis von Bedingungen (e.g. alle Logs mit Statuscode 500)
  - Reindex mit neuer Ingest Pipeline
  - Kopieren von Daten zwischen Clustern