# ElasticSearch入門

2013/08/29

@johtani

# 自己紹介

- 氏名：大谷　純
- Twitter：@johtani
- lucene-gosenのコミッター
- Solr入門著者の1人
- ブログ：http://blog.johtani.info
- 所属：株式会社シーマーク

# アジェンダ

- 検索エンジンって？
  - 転置インデックス、N-gram、形態素の話
- ElasticSearchとは？
  - インデックスの論理構成と物理構成
  - 機能概要
  - 参考資料など

# 検索エンジンって？

（ElasticSearchとは関係のない概要的な話）

# 全文検索って？

- 全文検索（Full text search）とは、コンピュータにおいて、複数の文書（ファイル）から特定の文字列を検索すること。「ファイル名検索」や「単一ファイル内の文字列検索」と異なり、「複数文書にまたがって、文書に含まれる全文を対象とした検索」という意味で使用される。
（Wikipediaより）

# 用語

- 文書（ドキュメント）
  検索エンジンに保存されたデータ。RDBでのレコードに相当

- クエリ
  検索条件、検索式

- スキーマ
  RDBのテーブルに相当

- フィールド
  RDBのカラムに相当

- ターム（Term）/トークン（Token）
  インデックスのキーになる単語（文字列）

- ドキュメント登録の流れ

1. カツオはサザエの弟
2. サザエはワカメの姉

ドキュメントの登録

- ドキュメント登録の流れ

| | |
|---|---|
| 1 | カツオはサザエの弟 |
| 2 | サザエはワカメの姉 |

ドキュメントの登録

| | | | | | | |
|---|---|---|---|---|---|---|
| 1 | カツオ | は | サザエ | の | 弟 | |
| 2 | サザエ | は | ワカメ | の | 姉 | |

単語に分割

- ドキュメント登録の流れ

1 カツオはサザエの弟

2 サザエはワカメの姉

ドキュメントの登録

1 カツオ | は | サザエ | の | 弟

2 サザエ | は | ワカメ | の | 姉

単語に分割

カツオ → 1

サザエ → 1 2

ワカメ → 2

の → 1 2

は → 1 2

弟 → 1

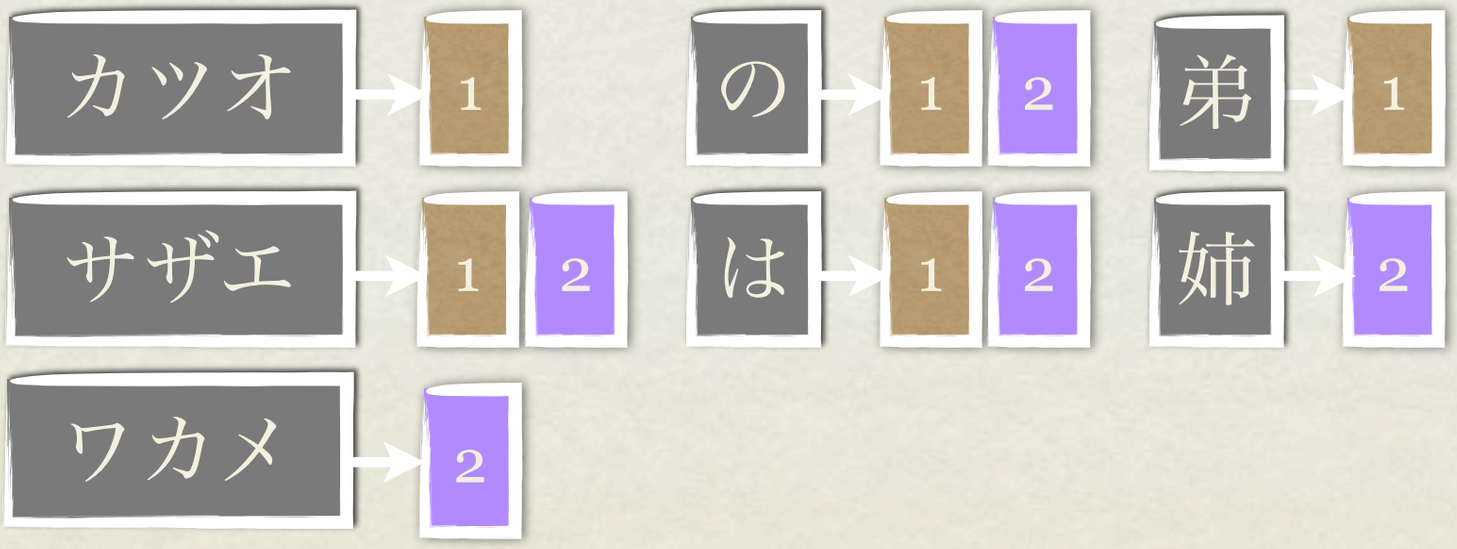
姉 → 2

単語からidの配列が
引けるように
＝転置インデックス

- 検索の流れ

カツオ　サザエ

検索文字入力



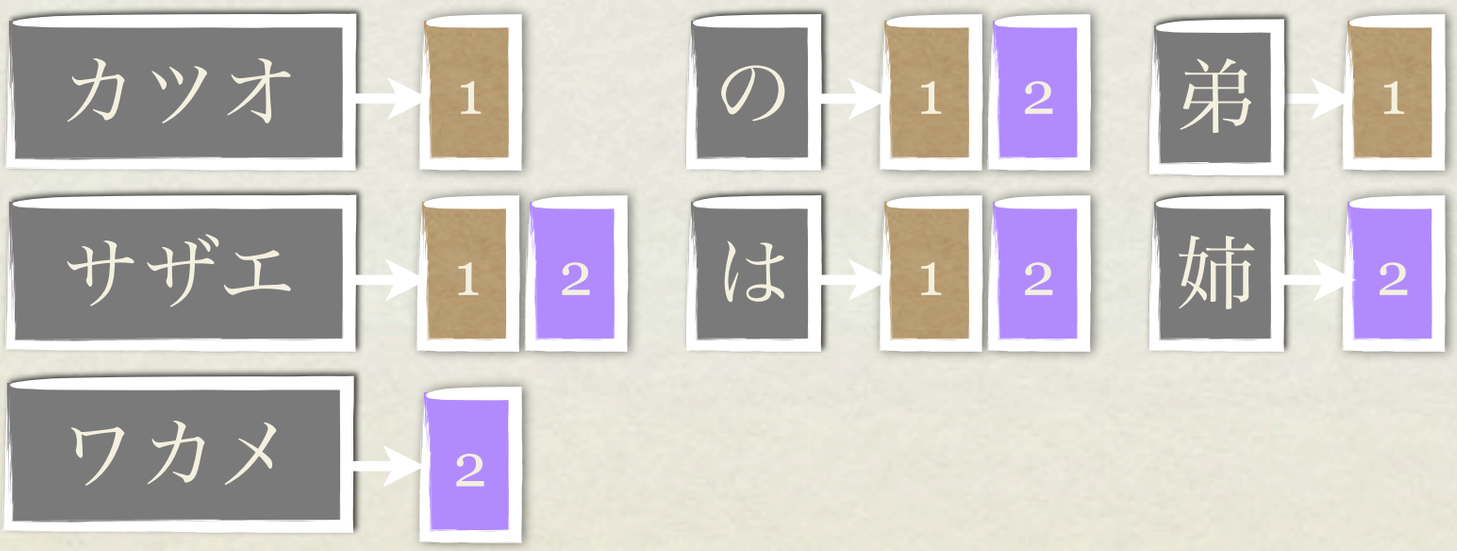
- 検索の流れ

カツオ　サザエ

検索文字入力

カツオ　AND　サザエ

検索文字のパース
＋検索クエリ化



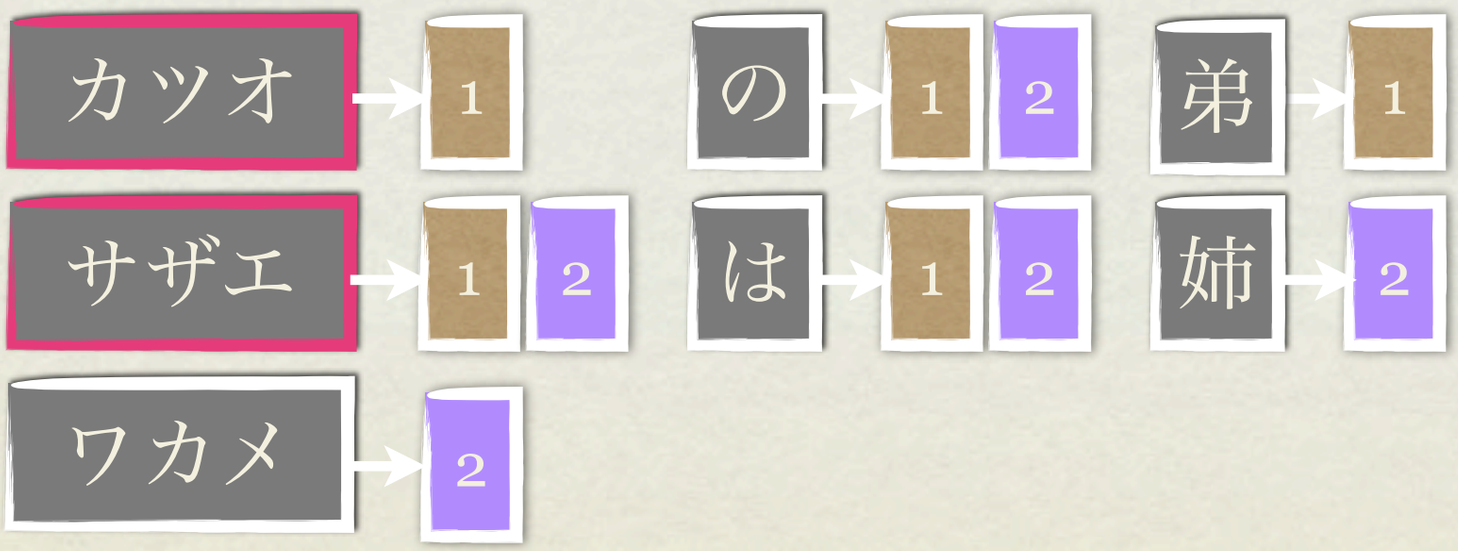
- 検索の流れ

カツオ　サザエ

検索文字入力

カツオ　AND　サザエ

検索文字のパース
＋検索クエリ化




転置インデックスを検索

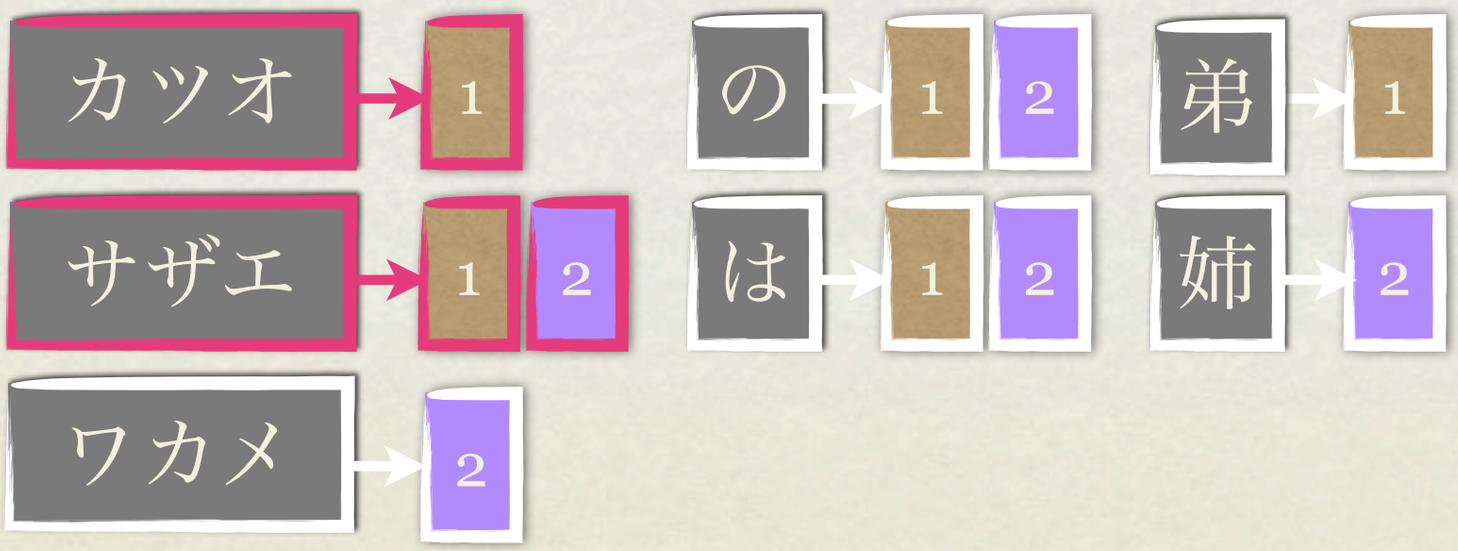
- 検索の流れ

カツオ　サザエ

検索文字入力

カツオ　AND　サザエ

検索文字のパース
＋検索クエリ化




転置インデックスを検索
→AND検索なのでidの
配列のANDをとる

- 検索の流れ

カツオ　サザエ

検索文字入力

カツオ　AND　サザエ

検索文字のパース
＋検索クエリ化

カツオ → 1

サザエ → 1 2

ワカメ → 2

の → 1 2

は → 1 2

弟 → 1

姉 → 2

転置インデックスを検索
→AND検索なのでidの
配列のANDをとる

# N-gramと形態素解析

- 転置インデックスのキー（単語）の作り方
  日本語は単語の切れ目がわからないので転置インデックスのキーは主につぎの2つの手法で作る

- N-gram
  - N（1以上の数字）文字ずつ文章を区切る
- 形態素解析
  - 辞書などを用いて意味のある単語で区切る

# N-gram

- Nが2の時の例：

カツオはサザエの弟

カツ　ツオ　オは　はサ　サザ　ザエ　エの　の弟

- メリット：
  - 検索漏れがない
- デメリット：
  - 精度が悪い→「カツ」でもヒットしちゃう
  - インデックスが肥大化
  - 語幹（単語の基本形とか）で検索不可

# 形態素解析

- 形態素解析の例：


- メリット：
  - 精度がよい：「カツ」で検索→「カツオ」ヒットしない
  - 語幹による検索が可能：「飲み」→「飲む」が検索可
- デメリット：
  - 新語（未知語）に弱い→辞書ベースの場合に辞書にない単語がわからない。辞書更新後はインデックス再作成

# ドキュメントとフィールド

- ドキュメントは複数のフィールドから構成される
- フィールド単位で転置インデックスを作成
- フィールド単位で検索が可能に

# ElasticSearchとは？

# 本日の要点

今日はこれだけは覚えて帰ってください。

# ElasticSearchは Amazon Web Servicesの サービスとは関係ありません

以上

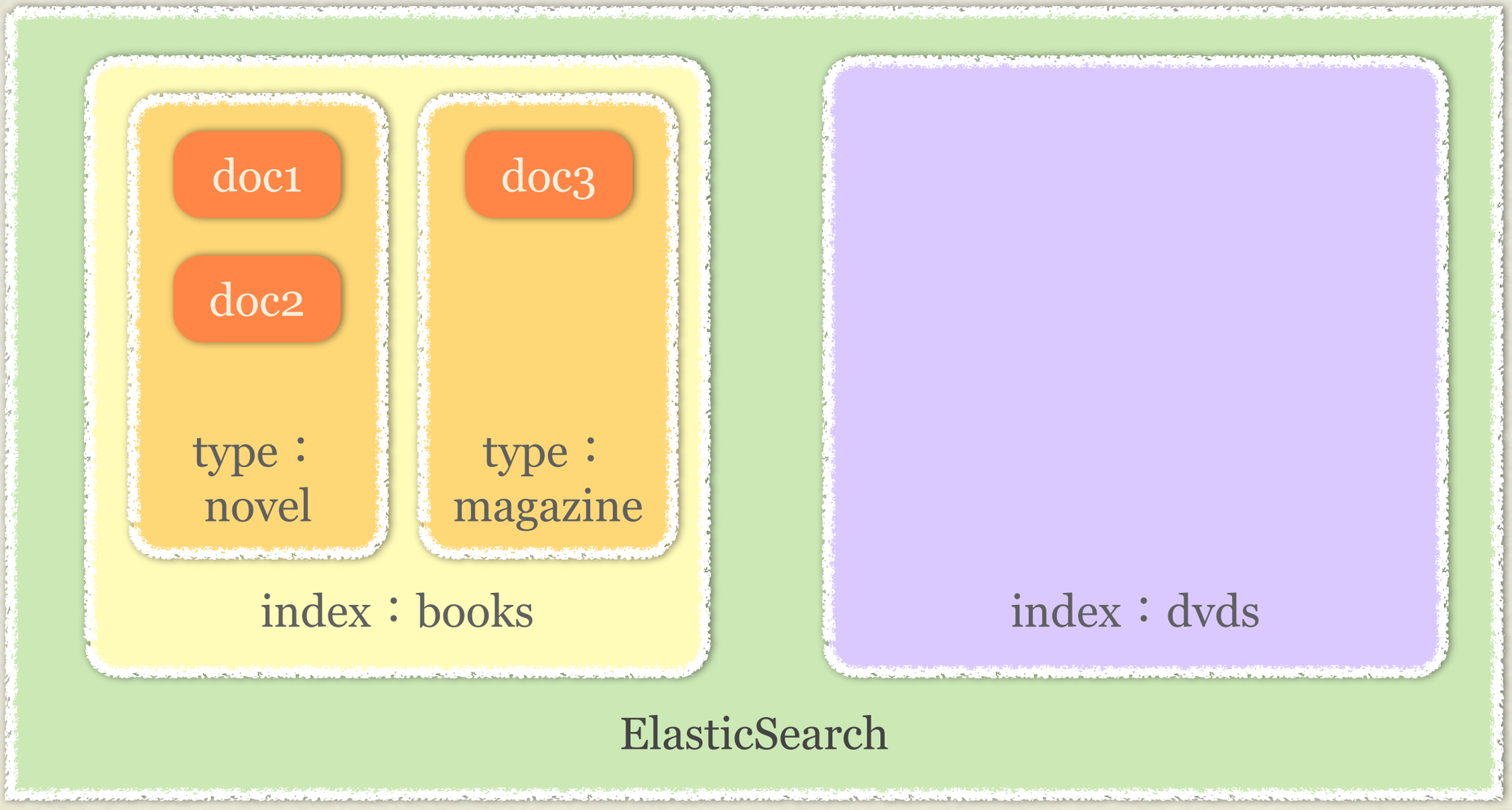
# ElasticSearchとは？

- 分散型RESTful検索＆解析サーバ
- OSS（Apache Software License Version 2.0）
- Apache Luceneをコアに採用（0.90.3はLucene 4.4.0に対応）
- マルチテナント対応
- スキーマフリー＆ドキュメント指向
- 楽観的排他制御
- などなど

# ElasticSearchの
# インデックス構成
# (論理と物理)

# インデックス構成（論理）

- 「インデックス/タイプ（マッピングタイプ）/ドキュメント」という構成
- ※Solrにはタイプという概念がない



# インデックス構成（論理）

- ドキュメント
  ：RDBの行に相当する概念。
  構造が固定でない＝スキーマフリー（スキーマを定義することも可能）

- タイプ（マッピングタイプ）
  ：RDBのテーブルに相当する概念。
  ドキュメントをまとめるコンテナ的なもの。
  タイプ毎にスキーマ（ESではマッピングという）を定義可能

- インデックス
  ：RDBのデータベースに相当する概念。
  物理的なファイルに保存される単位
  （タイプはあくまでも概念的なもので、
  物理的にはインデックスの単位で保存。※物理構成は後述）

# インデックス構成（論理）

- ElasticSearchでは論理構成がURLと同じ構成
- ドキュメントの登録
  - curl -XPUT http://localhost:9200/books/novel/doc1 -d '{...}'
- ドキュメントの検索（インデックス単位）
  - curl -XGET http://localhost:9200/books/_search?q=...
- ドキュメントの検索（タイプ単位）
  - curl -XGET http://localhost:9200/books/novel/_search?q=...
- ドキュメントのIDによる取得（ドキュメント単位）
  - curl -XGET http://localhost:9200/books/novel/doc1

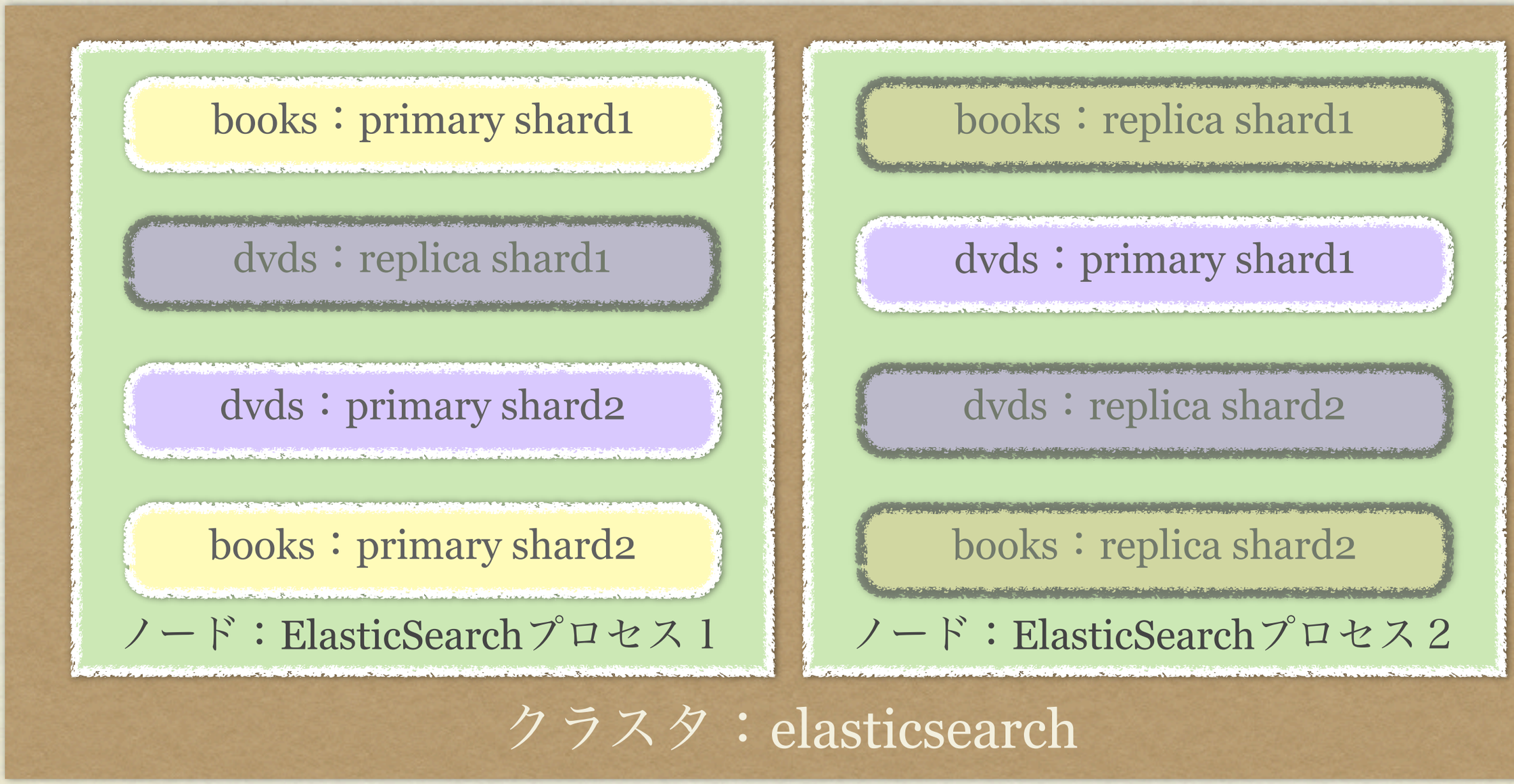
# インデックス構成（論理）

- 複数のインデックスを横断的に検索も可能（カンマ区切りでOK）
  - curl -XGET http://localhost:9200/books,dvds/_search?q=...
- 全インデックスを横断検索するには
  - curl -XGET http://localhost:9200/_search?q=...
- 全インデックスのタイプ指定での検索も可能
  - curl -XGET http://localhost:9200/_all/magazine/_search?q=...

# インデックス構成（物理）

# インデックス構成（物理）

- クラスタとノード、インデックスとシャード（shard）の関係



# インデックス構成（物理）

- クラスタはn個のノードから構成（cluster.nameが同一なら同一クラスタ）
- インデックスはn個のシャードから構成され、シャード単位でノードに配置されることで分散可能
- シャード単位でレプリカを保持可能
- デフォルトで、5シャード、1レプリカ（/1シャード）
- ※シャード数はインデックス作成時にのみ指定が可能。レプリカ数は変更可能
  - シャード分割機能がないため。Solrはshard splitting機能あり（2013/04）
- ※同一プロセス上に同一シャードのレプリカとプライマリが乗らないようになっている

# クラスタ管理

- クラスタへのノード追加などのノード探索はデフォルトZen Discovery
- マルチキャストorユニキャストによるノード探索
  - マルチキャスト：マルチキャストを利用したノード探索およびクラスタ構築
  - ユニキャスト：ホストリストを利用したクラスタの構築
- クラスタ起動後にマスターノードを選出（マルチマスタが可能）
- マスタノードがクラスタ全体の障害検知を実施
- http://www.elasticsearch.org/guide/reference/modules/discovery/zen/
- ※ZooKeeperを用いたノード探索のプラグインもある

# 機能概要

# 機能概要

- 設定関連
  - 型の自動決定、動的マッピング、テンプレートなど
- データ構造
  - 入れ子、親子構造、部分更新
- 便利機能
  - TTL、スロークエリログ、スローインデクシングログ、プラグイン
- 検索、登録機能
  - Geo、 Percorator、バルクインサート

# 設定関連

# 設定関連（基本）

- ほとんどすべての設定について、REST APIで設定が可能
  - 設定ファイルとして保存も可能
- 確認もREST APIで可能
  - クラスタの状態なども

```
curl -XGET 'http://localhost:9200/_cluster/health?pretty=true'
```

- インデックスの作成、削除も

```
curl -XPUT 'http://localhost:9200/twitter/'
```

```
curl -XDELETE 'http://localhost:9200/twitter/'
```

# 設定関連（型の決定機能）

- JSONオブジェクトから自動でフィールドの型を推測して登録可能（スキーマフリーと呼ばれる所以）

- 数値、日付、文字列などを推測してフィールドを決定
- 数値： numeric_detectionにtrue/falseを設定
- 日付：dynamic_date_formats：joda-timeライブラリの日付フォーマット文字列を指定（複数指定可）

# 型の自動決定の注意点

- 注意点
  - すでに型を決定したフィールドに異なる型のデータをいれるとエラーになったり、切り捨てられたり
    - エラー例：24（long型）を登録後、”hoge”（string型）を登録
      - ignore_malformedがtrueで設定されていればエラーにならない
    - エラーにならない例：24（long型）を登録後、12.3（float型）を登録
      - デフォルトでは小数点以下が切り捨てられて12で登録されてしまう

# 設定関連（動的マッピング）

- フィールド名などによるフィールド型の推定を設定可能
- Solrのダイナミックフィールドのより柔軟な機能
- mappings定義にdynamic_templatesの設定を配列で追加

```
{
    "person" : {
        "dynamic_templates" : [
            {
                "template_1" : {
                    "match" : "multi*",
                    "mapping" : {
                        "type" : "multi_field",
                        "fields" : {
                            "{name}" : {"type": "{dynamic_type}", "index" : "analyzed"},
                            "org" : {"type": "{dynamic_type}", "index" : "not_analyzed"}
                        }
                    }
                }
            },...
```

# 設定関連（テンプレート）

- インデックス単位の設定もテンプレート化可能
- インデックス名にtemplateが一致したものを適用。orderの大きいもので上書き
  - 例：基本設定はレプリカ0だが、ha_ではじまるインデックスはレプリカ5

```
curl -XPUT localhost:9200/_template/main_template -d '
{
    "template" : "*",
    "order" : "1",
    "settings" : {
        "index.number_of_replicas" : 0
    }
}
```

```
curl -XPUT localhost:9200/_template/ha_template -d '
{
    "template" : "ha_*",
    "order" : "10",
    "settings" : {
        "index.number_of_replicas" : 5
    }
}
```

# データ構造

# データ構造（ネスト）

- ネスト、親子、オブジェクトなどのデータも登録可能
  （Solrは基本フラットなデータのみ）
- 例：Tシャツの色とサイズの組み合わせ（ネスト）

データ

```
{
    "variation" : [
        {
            "color" : "blue",
            "size" : 4
        },
        {
            "name" : "green",
            "size" : 6
        }
    ]
}
```

クエリ

```
{
    "query" : {
        "nested" : {
            "path" : "variation",
            "query" : {
                "bool" : {
                    "must" : [
                        { "term" : {"variation.name" : "green"} },
                        { "term" : {"variation.size" : "6"} }
                    ]
                }
            }
        }
    }
}
```

# データ構造（親子）

- 親子関係に関する検索も可能（mappingで設定も必要）
- 子データ登録時に親IDを指定

```
$ curl -XPUT localhost:9200/shop/cloth/1 -d '{
    "name" : "something"
}'
```

親データ

```
$ curl -XPUT localhost:9200/shop/variation/1?parent=1 -d '{
    "color" : "red",
    "size" : "XL"
}'
```

子データ

```
$ curl -XPUT localhost:9200/shop/variation/2?parent=1 -d '{
    "color" : "green",
    "size" : "S"
}'
```

子データ

クエリ

```
{
  "has_child" : {
    "type" : "variation",
    "query" : {
      "bool" : {
        "must" : [
          { "term": {"size" : "XL"} },
          { "term": {"color" : "red"} }
        ]
      }
    }
  }
}
```

# データ構造（部分更新）

- _sourceフィールドにデフォルトで入力されたJSONのデータを保持（Solrにはない）
  - _sourceフィールドを使用する部分更新が可能

```
curl -XPOST 'localhost:9200/test/type1/1/_update' -d '{
    "script" : "ctx._source.counter += count",
    "params" : {
        "count" : 4
    }
}'
```

# 便利機能（TTL）

- TTL＝Time To Live
- データの生存期間を指定可能（例は1日）
- Expireのチェック処理は60s毎に行われる（変更可）

mapping

```
{
    "tweet" : {
        "_ttl" : { "enabled" : true, "default" : "1d" }
    }
}
```

# 便利機能（ログ）

- index slow log/search slow log（Solrにはない機能）
  - インデックス時、検索時に処理時間がしきい値を超えたらログ出力
  - ログレベルごとに値を設定可能

```
#index.search.slowlog.threshold.query.warn: 10s
#index.search.slowlog.threshold.query.info: 5s
#index.search.slowlog.threshold.query.debug: 2s
#index.search.slowlog.threshold.query.trace: 500ms

#index.search.slowlog.threshold.fetch.warn: 1s
#index.search.slowlog.threshold.fetch.info: 800ms
#index.search.slowlog.threshold.fetch.debug: 500ms
#index.search.slowlog.threshold.fetch.trace: 200ms
```

# 便利機能（プラグイン）

- ElasticSearchの拡張機能
  - River：データをElasticSearchに流し込むプラグイン（例：RabbitMQとか）
  - Analysis：アナライザのプラグイン（例：Kuromojiとか）
  - Site：クラスタ管理とか
- プラグインのインストールは付属のpluginコマンドにて可能
（通常はネットに接続している必要がある）
- プラグインの一覧（http://www.elasticsearch.org/guide/reference/modules/plugins/）

# 検索機能（Percorator）

- クエリをインデックス化して登録
  - 登録済みのクエリにヒットするデータかどうかを特定のリクエストにデータを送信することでわかる仕組み（利用可能なクエリは一部制限あり）

percoratorの登録

```
curl -XPUT localhost:9200/_percolator/test/kuku -d '{
    "query" : {
        "term" : {
            "field1" : "value1"
        }
    }
}
```

percoratorの確認

```
curl -XGET localhost:9200/test/type1/_percolate -d '{
    "doc" : {
        "field1" : "value1"
    }
}'
```

And the matches are part of the response:

```
{"ok":true, "matches":["kuku"]}
```

# 検索機能（Geo）

- 緯度経度データを利用した検索
  - 緯度経度、geohash、geo_shapeなど
- 中心点からの距離、矩形、geohashによる絞込などが可能
- 距離によるファセットも

# 登録機能（バルク処理）

- 大量の処理を行うためのAPI（_bulk）
  - http://www.elasticsearch.org/guide/reference/api/bulk/
- UDPも利用可能

バルク処理データ例

```
{ "index" : { "_index" : "test", "_type" : "type1", "_id" : "1" } }
{ "field1" : "value1" }
{ "delete" : { "_index" : "test", "_type" : "type1", "_id" : "2" } }
{ "create" : { "_index" : "test", "_type" : "type1", "_id" : "3" } }
{ "field1" : "value3" }
{ "update" : {"_id" : "1", "_type" : "type1", "_index" : "index1"} }
{ "doc" : {"field2" : "value2"} }
```

# その他（関連ツールとか）

- Kibana3＋ElasticSearch＋(logstash OR fluentd)
  - ElasticSearchをストレージにしてログデータの保存と可視化を行う

# 参考資料など

# 参考ページ

- 本家＆ガイド
  - http://www.elasticsearch.org
  - http://www.elasticsearch.org/guide/
- サポート＆トレーニング
  - http://elasticsearch.com
- Solr vs ElasticSearchの日本語翻訳版
  - https://github.com/minghai/ElasticSearch_VS_Solr
- Sematextのブログ
  - http://blog.sematext.com

# 参考書籍とか

- ElasticSearch Server（Packt Publishing、2013/02月発売）
  - http://www.packtpub.com/elasticsearch-server-for-fast-scalable-flexible-search-solution/book
  - http://elasticsearchserverbook.com
- Mastering ElasticSearch（Packt Publishing、2013/12発売予定）
  - http://www.packtpub.com/mastering-elasticsearch-querying-and-data-handling/book
- Elasticsearch in Action（Manning Publications、2014/春、MEAP購入可）
  - http://www.manning.com/hinman/

# おまけ

# LuceneとESのバージョン

| ElasticSearch | Lucene |
|---|---|
| 0.90.3 | 4.4 |
| 0.90.2 | 4.3.1 |
| 0.90.1 | 4.3 |
| 0.90.0 | 4.2.1 |
| 0.90.0rc2 | 4.2.1 |
| 0.90.0rc1 | 4.2 |
| 0.20.6 | 3.6.2 |
| 0.20.5 | 3.6.2 |
| 0.90.0 beta1 | 4.1 |