Amazon CloudSearch
Amazon Elasticsearch Service

2016/03/23

AWS Black Belt Tech Webinar 2016

アマゾン ウェブ サービス ジャパン株式会社
ソリューションアーキテクト
篠原英治

```
{
    "Name" : ”Eiji Shinohara",
    "Twitter" : "@shinodogg",
    "Blog" : "http://shinodogg.com",
    "Profile" : {
        "Role" : "Solutions Architect",
        "Market": [
            "Startup",
            "Ad-Tech"
        ],
        "Subject Matter Expert" : [
            "Amazon CloudSearch",
            "Amazon Elasticsearch Service"
        ]
    }
}
```

Agenda

- 全文検索(Full-Text Search)
- 検索エンジンの基礎-Apache Lucene
- AWSの検索サービスのご紹介
  - Amazon CloudSearch
  - Amazon Elasticsearch Service
- CloudSearchとAmazon ESの比較

Agenda

- **全文検索(Full-Text Search)**
- 検索エンジンの基礎-Apache Lucene
- AWSの検索サービスのご紹介
  - Amazon CloudSearch
  - Amazon Elasticsearch Service
- CloudSearchとAmazon ESの比較

全文検索(Full-Text Search)

- 全文検索
  - https://ja.wikipedia.org/wiki/全文検索
  - “**全文検索**（ぜんぶんけんさく、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。”

検索と索引(インデックス)

- 順次検索
  - Unixのgrepのようなイメージ
- B-tree
  - RDBMSなどで使われるインデックス
- 転置インデックス
  - 検索エンジンで使われるインデックス
  - Amazon CloudSearch / Amazon Elasticsearch Serviceともに転置インデックスを作成

順次検索

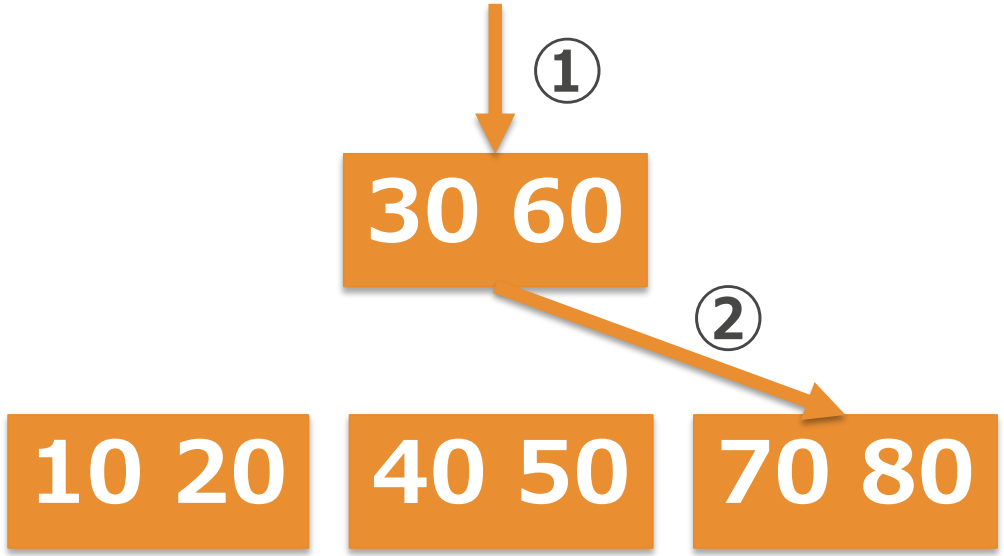
- UNIXのgrepのように上から全てみていく
  - 文書の量が膨大になると素早く結果を返すことが難しい

```
$ grep -n "Lucene" README.txt

1:# Apache Lucene README file
5:Lucene is a Java full-text search engine.  Lucene is not a complete
9: * The Lucene web site is at: http://lucene.apache.org/
10: * Please join the Lucene-User mailing list by sending a message to:
18:  The compiled core Lucene library.
23:To build Lucene or its documentation for a source distribution, see BUILD.txt
```

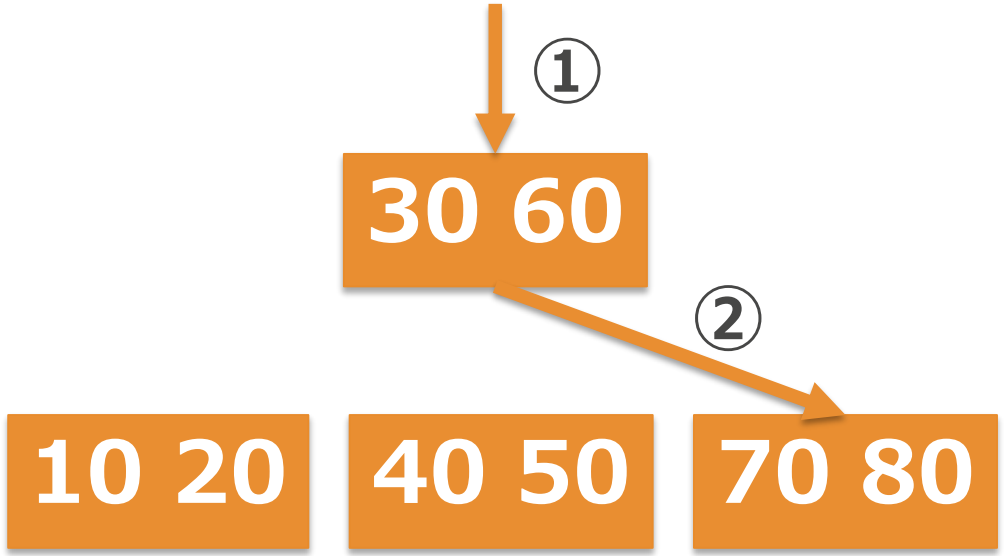
B-tree

- RDBMS等でもよく使われるアルゴリズム
  - データが格納されているブロックのポインタを索引で保持
    - クエリに対して早くデータを引き当てることができる
  - 例) 70のデータが欲しい

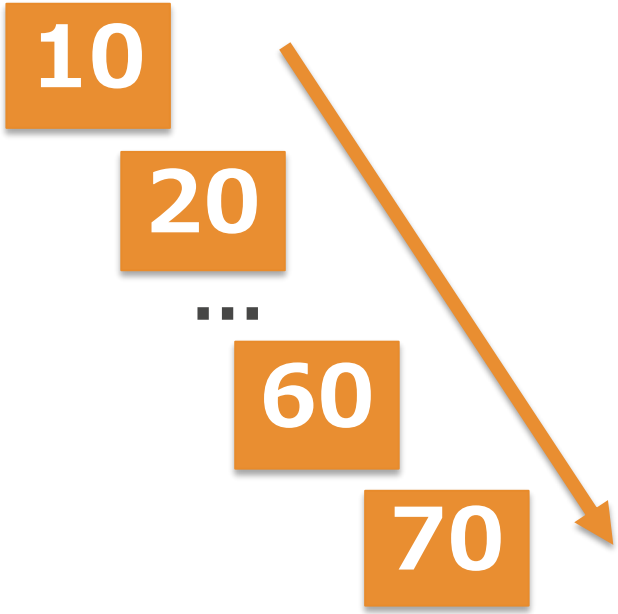


B-tree

- RDBMS等でもよく使われるアルゴリズム
  - データが格納されているブロックのポインタを索引で保持
    - クエリに対して早くデータを引き当てることができる
  - 例) 70のデータが欲しい

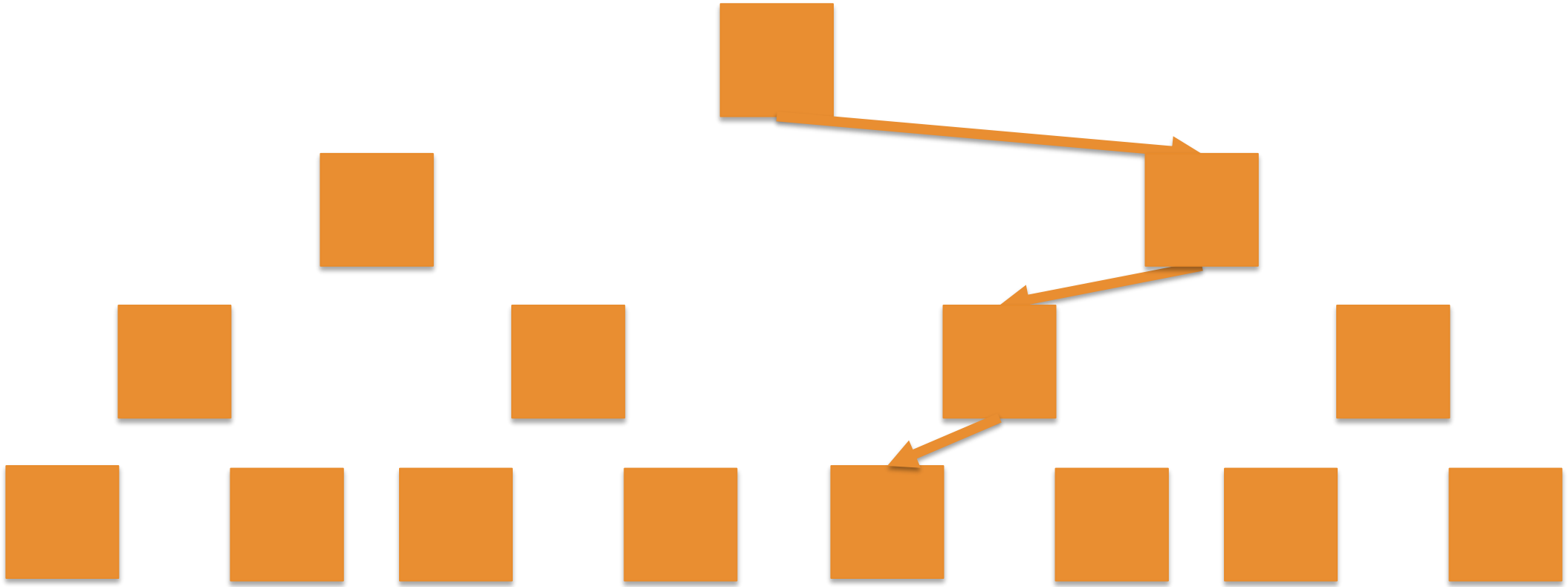


もし索引が無ければ



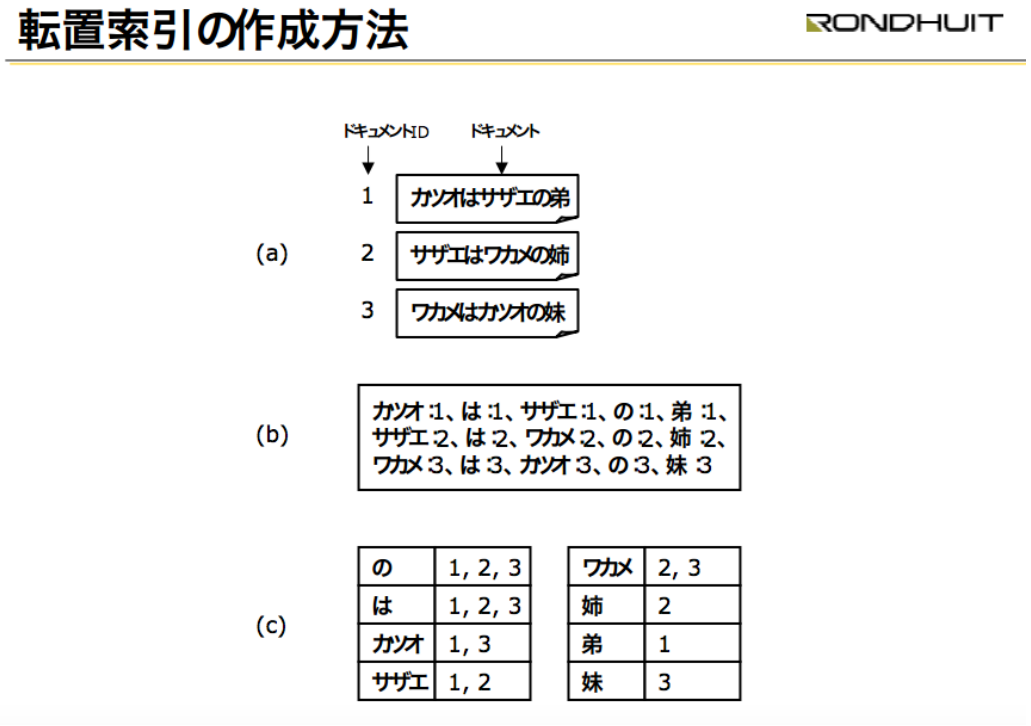
B-tree

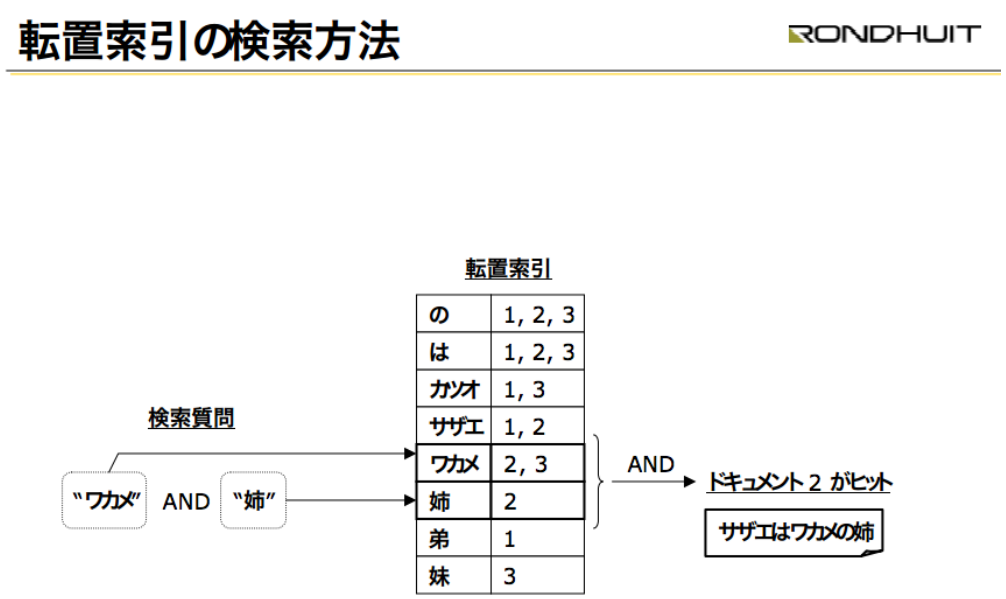
- RDBMS等でもよく使われるアルゴリズム
  - インデックスが巨大になったら？？



転置インデックス

- キーワードがどの文書に存在するかを索引付け
  - http://rondhuit.com/lucene-for-bea-060710.pdf





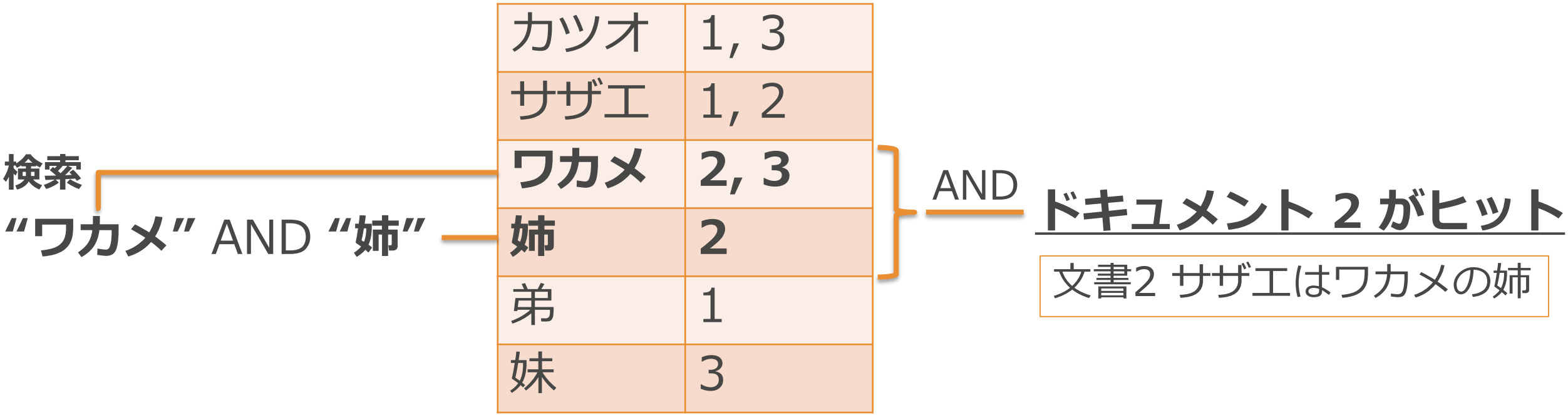
転置インデックス

1. 3つの文書(ドキュメント)があるとする
   - 文書1 - カツオはサザエの弟
   - 文書2 - サザエはワカメの姉
   - 文書3 - ワカメはカツオの妹
2. 文書を単語に分割(『は』と『の』は重要でない品詞とみなして割愛)
   - カツオ :1, サザエ :1, 弟 :1
   - サザエ :2, ワカメ :2, 姉 :2
   - カツオ :3, サザエ :3, 弟 :3
3. 出来上がったインデックス

カツオ ⇒ 1, 3	**姉 ⇒ 2**
サザエ ⇒ 1, 2	**弟 ⇒ 1**
ワカメ ⇒ 2, 3	**妹 ⇒ 3**

転置インデックス

- 全文検索エンジンの動作



転置インデックス

- インデックスが巨大になっても？？
  - 例) “ビール”で検索した時にヒットする文書は23

カツオ	1, 3
サザエ	1, 2
ワカメ	2, 3
姉	2
弟	1
妹	3

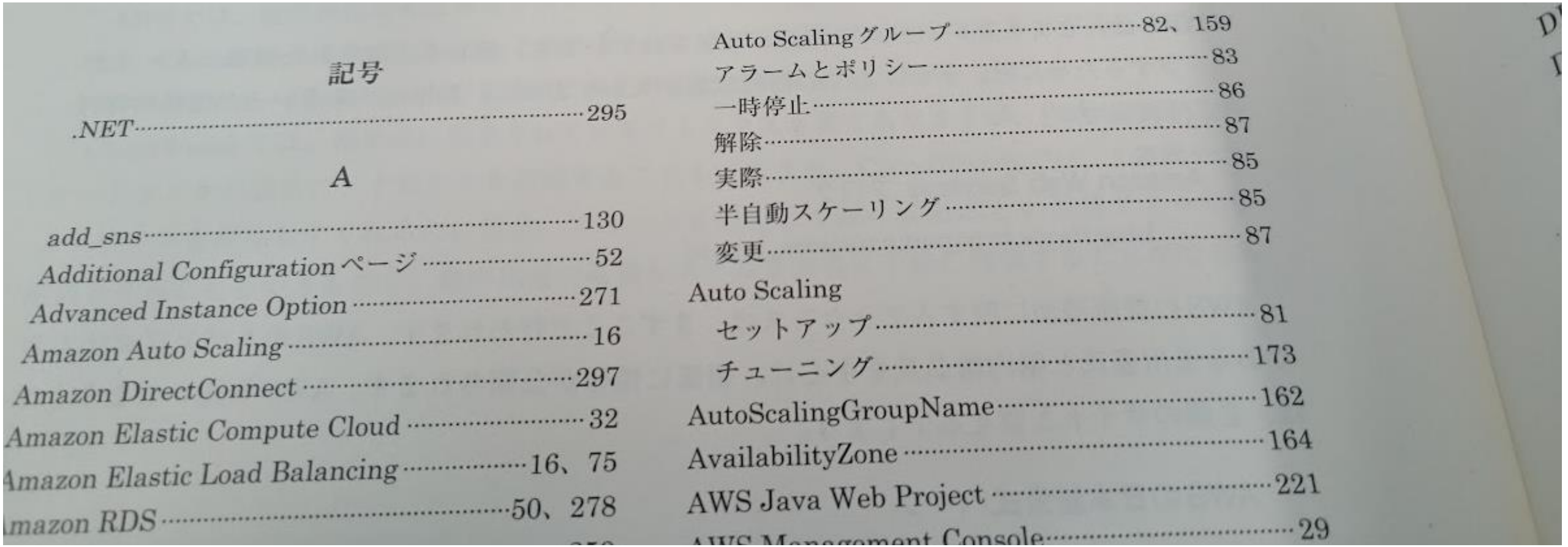
イルカ	9, 21
カエル	4, 8
イカ	7, 11
タコ	10
父	4, 8
母	9, 12

馬	9, 12
牛	9, 12
豚	12
東京	3, 14
福島	2, 14
人参	23

政治	3, 15
金	20
銀	4, 20
ビール	**23**
枝豆	23
ハサミ	5, 23

転置インデックス

- イメージ的には本の索引



記号

A

転置インデックス

- インデクシング
  - キーワードがどの文書に存在するか振り分ける処理
  - 検索精度という観点では以下のような処理も必要
    - 表記ゆれ
      - 例) “サーバー”、”サーバ”
    - 類義語
      - 例) “ベニス”、”ベネチア”
    - 正規化
      - 例) “54”、“五十四”、“五四”
        - » Add Japanese Kanji number normalization to Kuromoji
        - » https://issues.apache.org/jira/browse/LUCENE-3922

Agenda

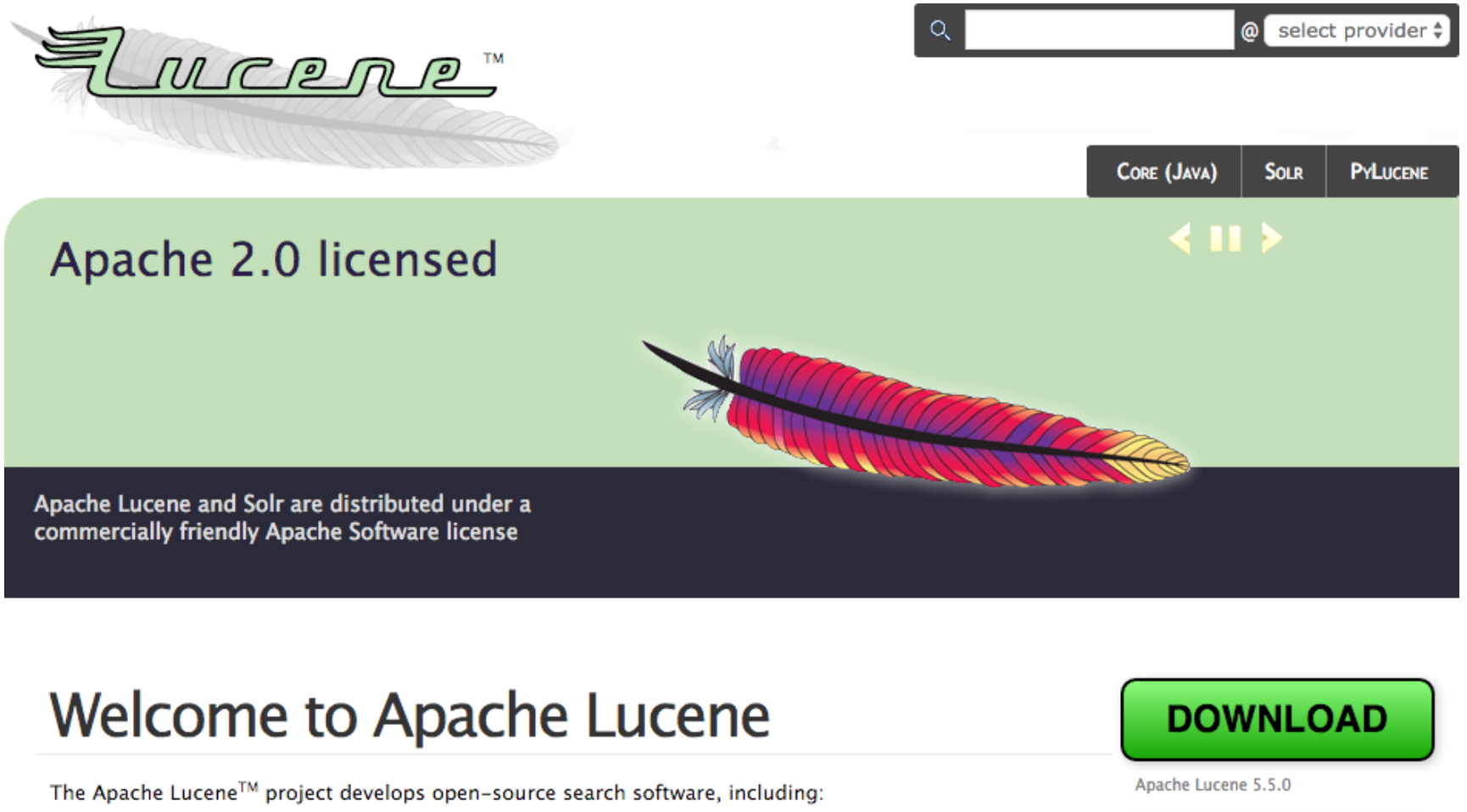
- 全文検索(Full-Text Search)
- **検索エンジンの基礎-Apache Lucene**
- AWSの検索サービスのご紹介
  - Amazon CloudSearch
  - Amazon Elasticsearch Service
- CloudSearchとAmazon ESの比較

検索エンジンの基礎 - Apache Lucene

- Javaで書かれた検索エンジンライブラリ
  - 今回のWebinarでご紹介する以下のプロダクト/サービスは全てon top of Luceneと言える
    - Apache Solr
    - Elasticsearch
    - Amazon CloudSearch
    - Amazon Elasticsearch Service
  - グローバルで豊富な実績
    - http://wiki.apache.org/lucene-java/PoweredBy

検索エンジンの基礎 - Apache Lucene

- 手軽にEC2やお手元のMac/Winで動かすことができる



検索エンジンの基礎 - Apache Lucene

- ダウンロードしてきたアーカイブを解凍して、jarファイルをJavaのCLASSPATHに追加
  1. **Lucene JAR**(core/lucene-core-{version}.jar)
  2. **queryparser JAR**(queryparser/lucene-queryparser-{version}.jar)
  3. **common analysis JAR**(analysis/common/lucene-analyzers-common-{version}.jar)
  4. **Lucene demo JAR**(demo/lucene-demo-{version}.jar)

```
export CLASSPATH=<<LucenePath>>/core/lucene-core-5.5.0.jar:<<LucenePath>>queryparser/lucene-queryparser-5.5.0.jar:<<LucenePath>>analysis/common/lucene-analyzers-common-5.5.0.jar:<<LucenePath>>demo/lucene-demo-5.5.0.jar
```

検索エンジンの基礎 - Apache Lucene

- Demo用のIndexerを使ってインデクシング

```
$ java org.apache.lucene.demo.IndexFiles -docs ./docs/
Indexing to directory 'index'...

adding ./docs/analyzers-common/allclasses-frame.html
adding ./docs/analyzers-common/allclasses-noframe.html
～略～
adding ./docs/test-framework/serialized-form.html
adding ./docs/test-framework/stylesheet.css

9895 total milliseconds
```

検索エンジンの基礎 - Apache Lucene

- Demo用のSearcherを使ってインデクシング

```
$ java org.apache.lucene.demo.SearchFiles
Enter query: kuromoji
Searching for: kuromoji 27 total matching documents

1. ./docs/analyzers-kuromoji/org/apache/lucene/analysis/ja/tokenattributes/package-summary.html
2. ./docs/analyzers-kuromoji/overview-summary.html
3. ./docs/analyzers-kuromoji/org/apache/lucene/analysis/ja/util/package-summary.html
```

検索エンジンの基礎 - Apache Lucene

- Demo用のIndexer&Searcher
  - このままでは日本語の解析等は出来ない

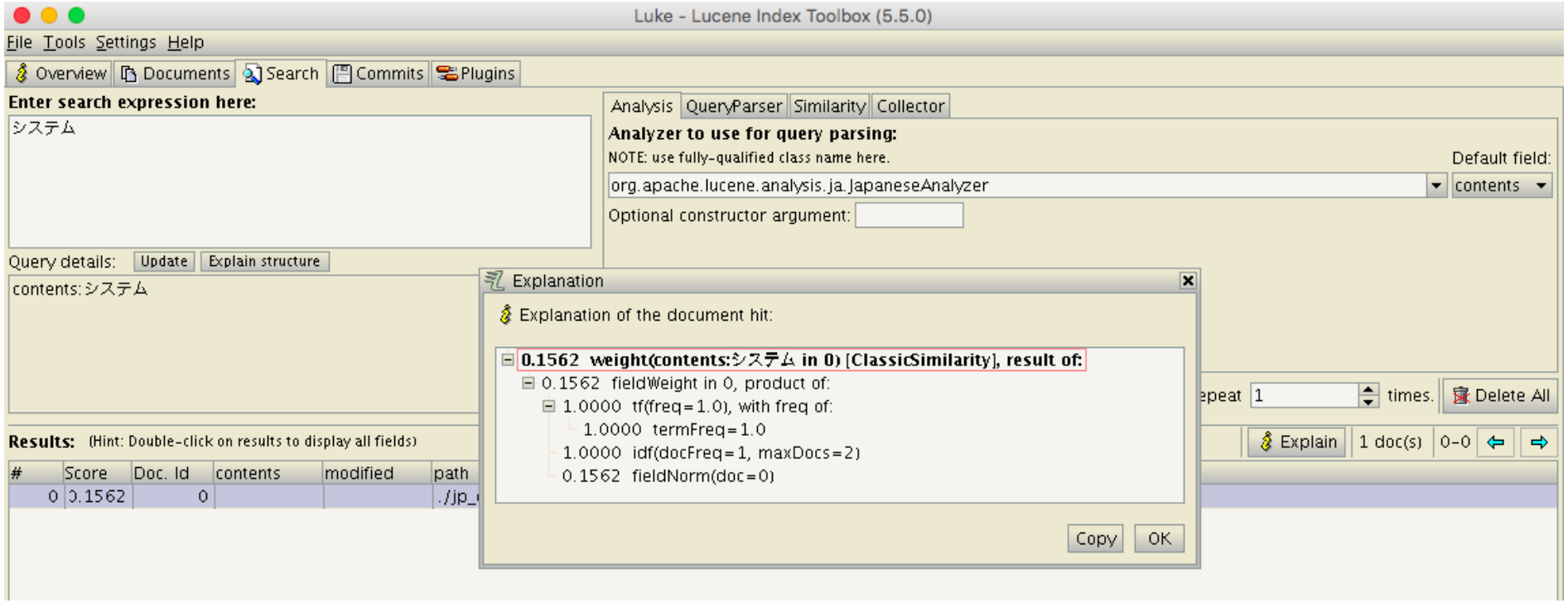
```
090     System.out.println("Indexing to directory '" + indexPath + "'...");
091
092     Directory dir = FSDirectory.open(Paths.get(indexPath));
093     Analyzer analyzer = new StandardAnalyzer();
094     IndexWriterConfig iwc = new IndexWriterConfig(analyzer);
095
096     if (create) {
097       // Create a new index in the directory, removing any
098       // previously indexed documents:
099       iwc.setOpenMode(OpenMode.CREATE);
```

検索エンジンの基礎 - Apache Lucene

- Demo用のIndexer&Searcher
  - main文を持つシンプルなJavaプログラム
  - このクラスを日本語対応させつつ、検索エンジンの理解を深めるのも良いかもしれません
    - Indexer
      - org.apache.lucene.demo.IndexFiles
      - https://lucene.apache.org/core/5_5_0/demo/src-html/org/apache/lucene/demo/IndexFiles.html
    - Searcher
      - org.apache.lucene.demo.SearchFiles
      - https://lucene.apache.org/core/5_5_0/demo/src-html/org/apache/lucene/demo/SearchFiles.html

検索エンジンの基礎 - Apache Lucene

- Luke: GUI(インデックスブラウザ)もあります
  - https://github.com/DmitryKey/luke
  - https://www.youtube.com/watch?list=PLGeM09tlguZTaS5FNoJGYEohaubtIvErS&v=fQAAzpk4oQ4#t=392



検索エンジンの基礎 - Apache Lucene

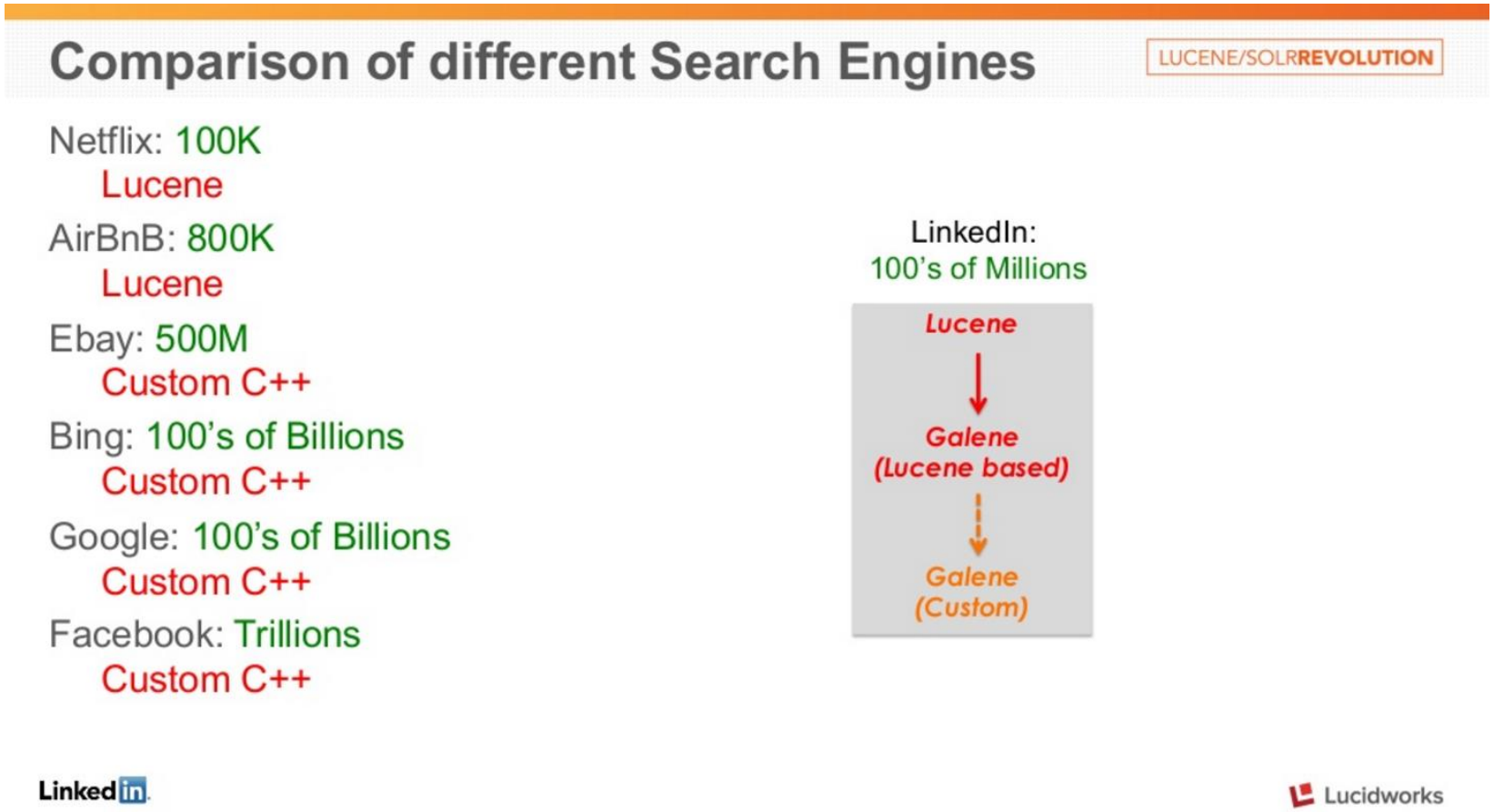
- Luceneをプロダクション環境で使っている事例
  - Lucene Solr RevolutionでのEvernoteやLinkedInの発表

Search Architecture at Evernote
https://www.youtube.com/watch?v=drOmahIie6c

Galene: LinkedIn's Search Architecture
https://www.youtube.com/watch?v=8O7cF75intk

自前で検索エンジンを構築？

- (2014年後半のスライドですが)自前で検索エンジンも一定数



http://www.slideshare.net/lucidworks/galene-linkedins-search-architecture-presented-by-diego-buthay-sriram-sankar-linkedin/8

A9 Product Search

- Amazonも自身(A9)で検索エンジンを構築しています

Product Search

If you've done a search on Amazon, you've used the A9 Product Search engine.

Our work starts long before a customer types a query. We've been analyzing data, observing past traffic patterns, and indexing the text describing every product in our catalog before the customer has even decided to search. As soon as we see the first keystroke, we're ready with instant suggestions and a comprehensive set of search results.

http://www.a9.com/whatwedo/product-search/

検索エンジンの基礎 - Apache Lucene

- Apache Lucene入門

本 › コンピュータ・IT › コンピュータサイエンス

Apache Lucene 入門 ~Java・オープンソース・全文検索システムの構築 大型本 – 2006/5/17

関口 宏司 ▾ (著)

★★★★☆ ▾ 5件のカスタマーレビュー

▸ すべての フォーマットおよびエディションを表示する

大型本
￥ 2,844 より

￥ 2,844 より 9 中古品の出品
￥ 4,190 より 1 コレクター商品の出品

http://www.amazon.co.jp/dp/4774127809

Apache Lucene入門

株式会社 ロンウイット

RONDHUIT
WWW.RONDHUIT.COM

http://rondhuit.com/lucene-for-bea-060710.pdf

検索エンジンの基礎 - Apache Lucene

- Similarity: tf-idf
  - tf-idf
    - term frequency–inverse document frequency
  - term frequency
    - ドキュメントの中に沢山そのキーワードが出てくればスコア高い
      - 大事なキーワードだとみなす
  - inverse document frequency
    - いろんなドキュメントに頻出するキーワードはスコア低い
      - ユニークなキーワードではない(＝重要ではない)とみなす
  - 検索結果のソート順に利用
    - スコアが高いものが上にくる
    - 特定のキーワードやフィールドをブーストさせることも可能

Agenda

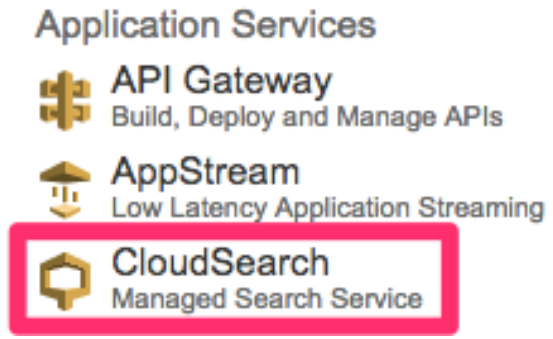
- 全文検索(Full-Text Search)
- 検索エンジンの基礎-Apache Lucene
- **AWSの検索サービスのご紹介**
  - Amazon CloudSearch
  - Amazon Elasticsearch Service
- CloudSearchとAmazon ESの比較

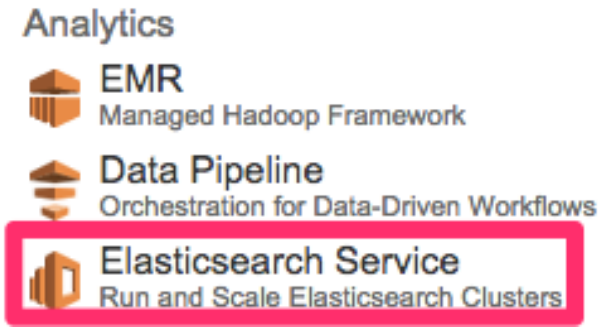
AWSの検索サービス

- Amazon CloudSearch
  - https://aws.amazon.com/jp/cloudsearch/
- Amazon Elasticsearch Service
  - https://aws.amazon.com/jp/elasticsearch-service/

Amazon CloudSearch

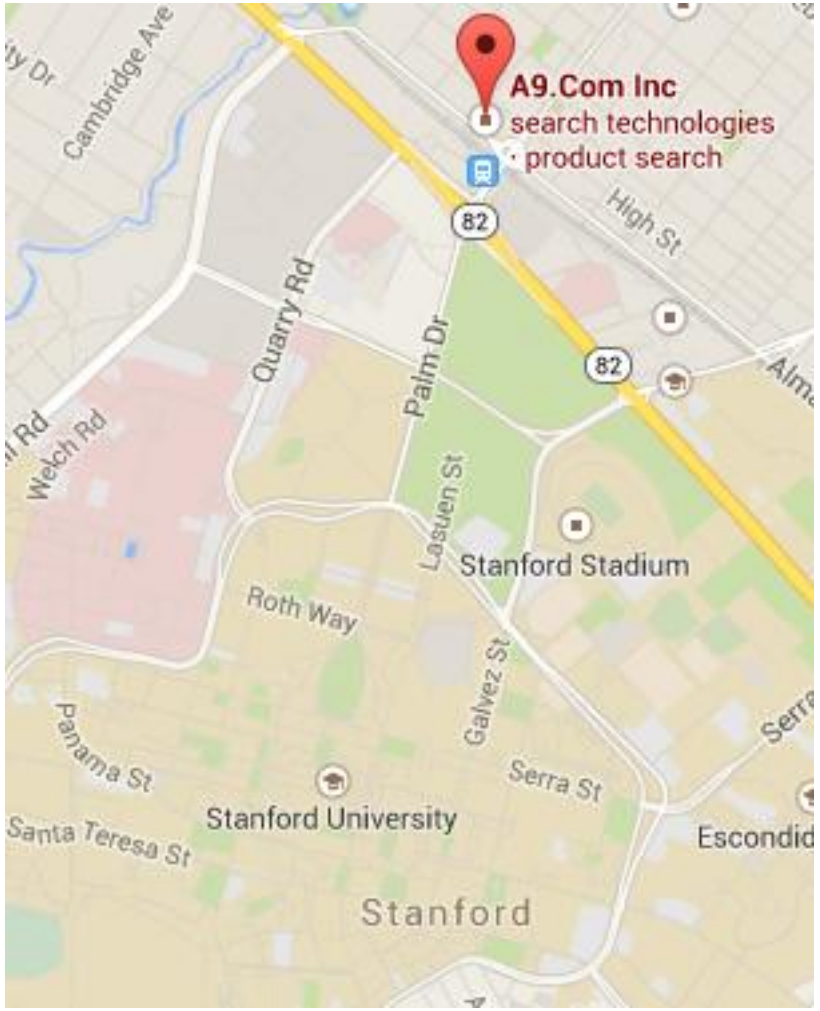
Amazon Elasticsearch Service





A9.com

- CloudSearch/Amazon ES の開発拠点はパロアルト

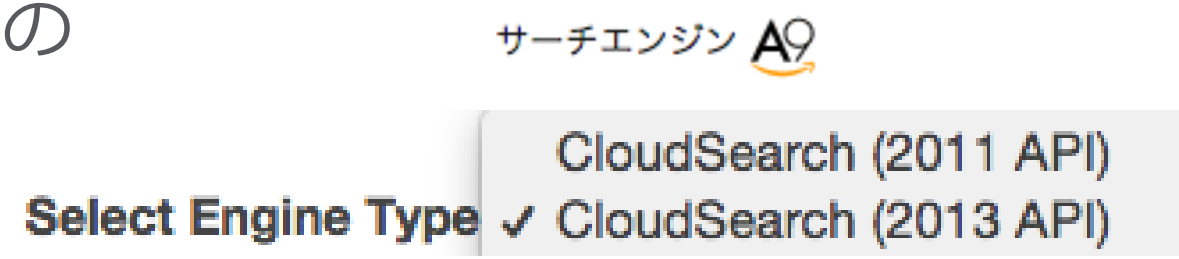


A9.com

- Amazonの商品検索もA9で作っています

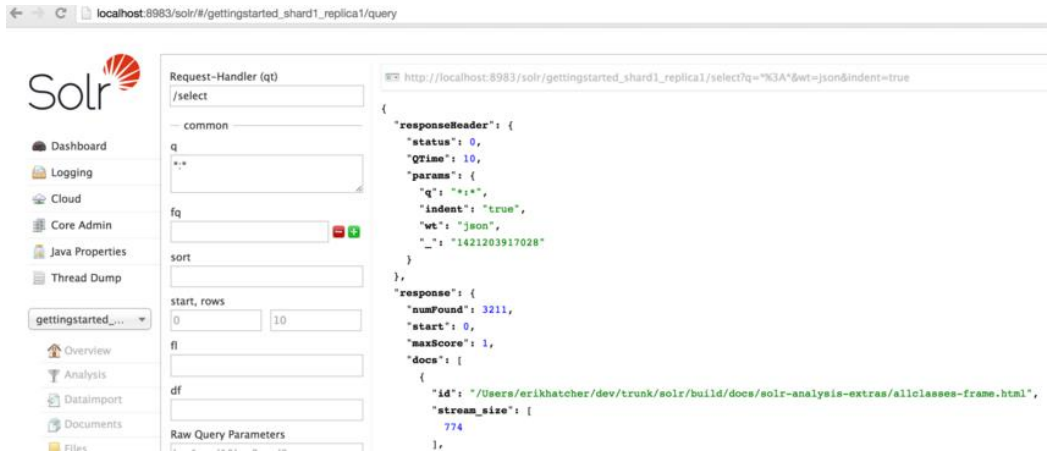
Amazon CloudSearch

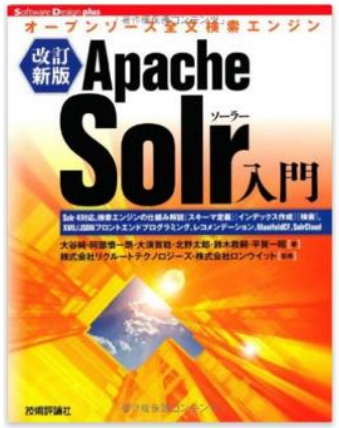
- 自動拡張するフルマネージド検索サービス
  - 2011 API
    - A9が作ったプロプライエタリな検索エンジン
    - Amazon.comで使っているもの
    - 東京リージョンは対象外
  - 2013 API
    - on top of Apache Solr
    - 多言語対応
      - 日本語の形態素解析、n-gram、カスタム辞書にも対応
    - 東京リージョンは2014年3月からサービス提供中



Amazon CloudSearch

- Apache Solr
  - 現在はApache Luceneのサブプロジェクト
  - 単なるLuceneのHTTPラッパーではなく様々な機能を持つ
    - 例えばSolrCloud
      - Zookeeperを活用したクラスタ管理
      - 分散検索・インデクシングを実現

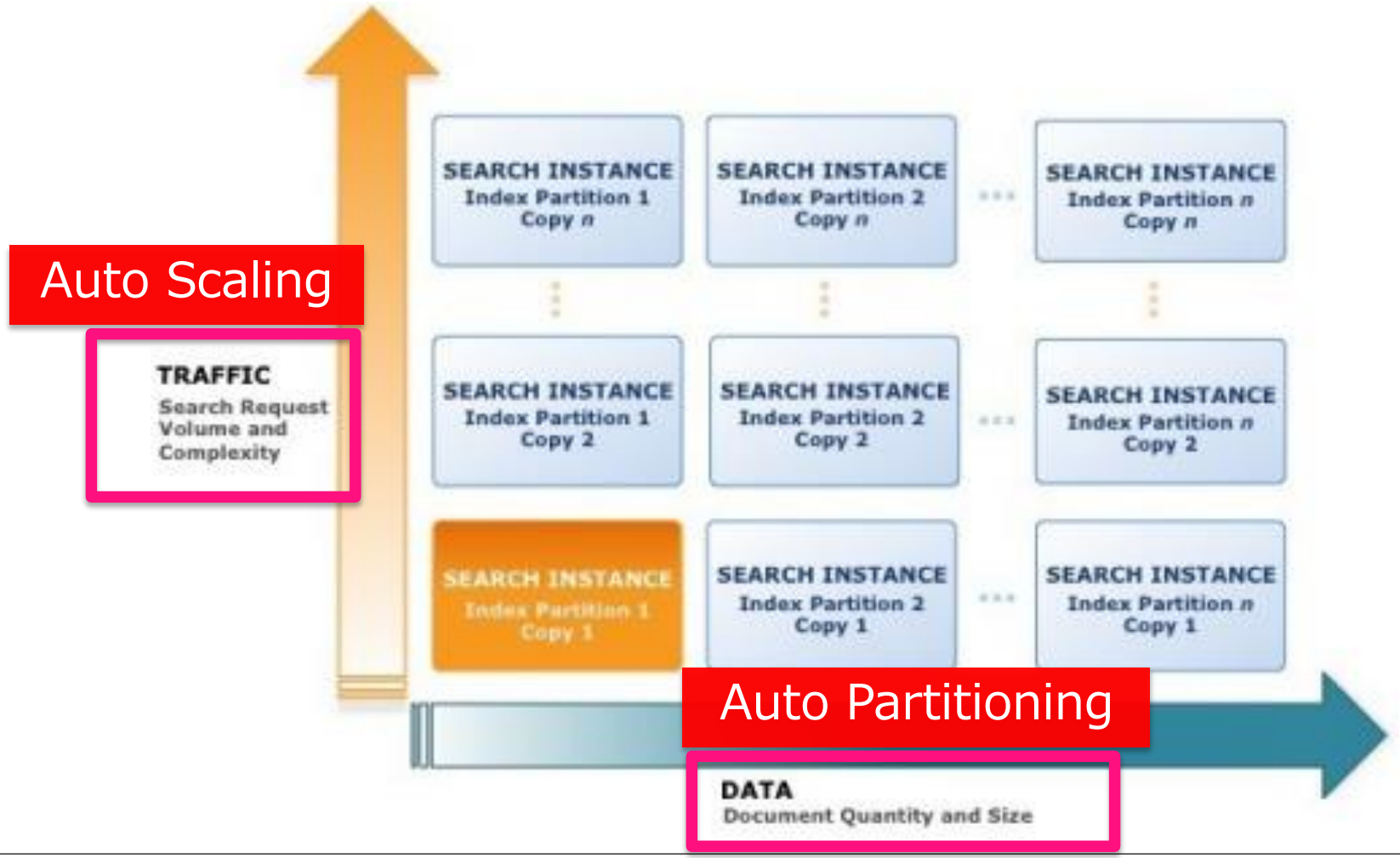




[改訂新版] Apache Solr入門 ~オープンソース全文検索エンジン (Software Design plus) 大型本 –
2013/11/29
大谷 純 (著), 阿部 慎一朗 (著), 大須賀 稔 (著), 北野 太郎 (著), & 4 その他
6件のカスタマーレビュー
すべての 2 フォーマットおよびエディションを表示する
大型本
¥ 3,888 プライム
¥ 2,457 より 10 中古品の出品
¥ 3,888 より 1 新品
住所からお届け予定日を確認 既定の住所を使用 詳細
3/23 水曜日 にお届けするには、今から22 時間 52 分以内に「お急ぎ便」または「当日お急ぎ便」を選択して注文を確定してください (有料オプション。

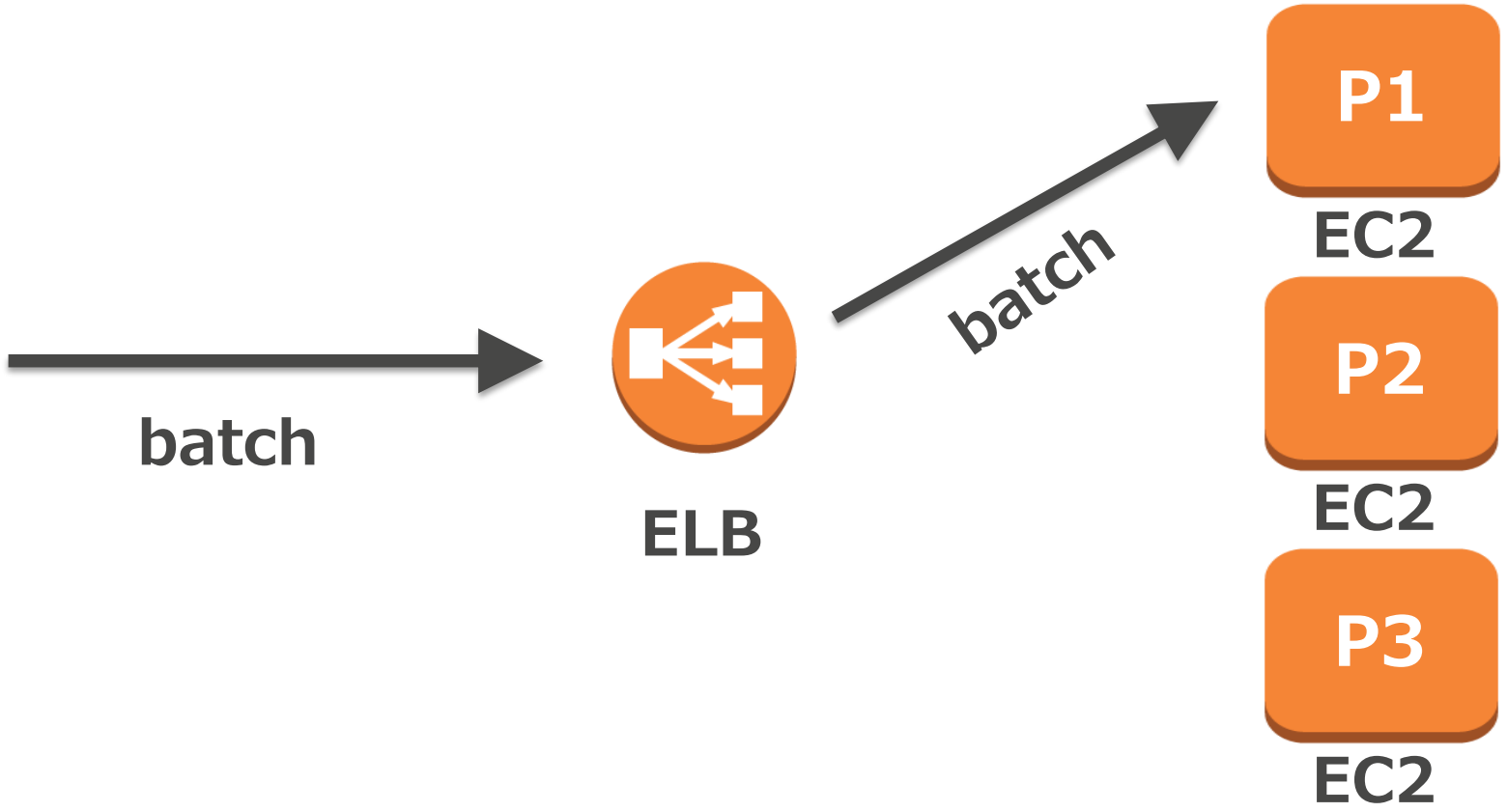
Amazon CloudSearch

- Auto Scaling / Auto Partitining



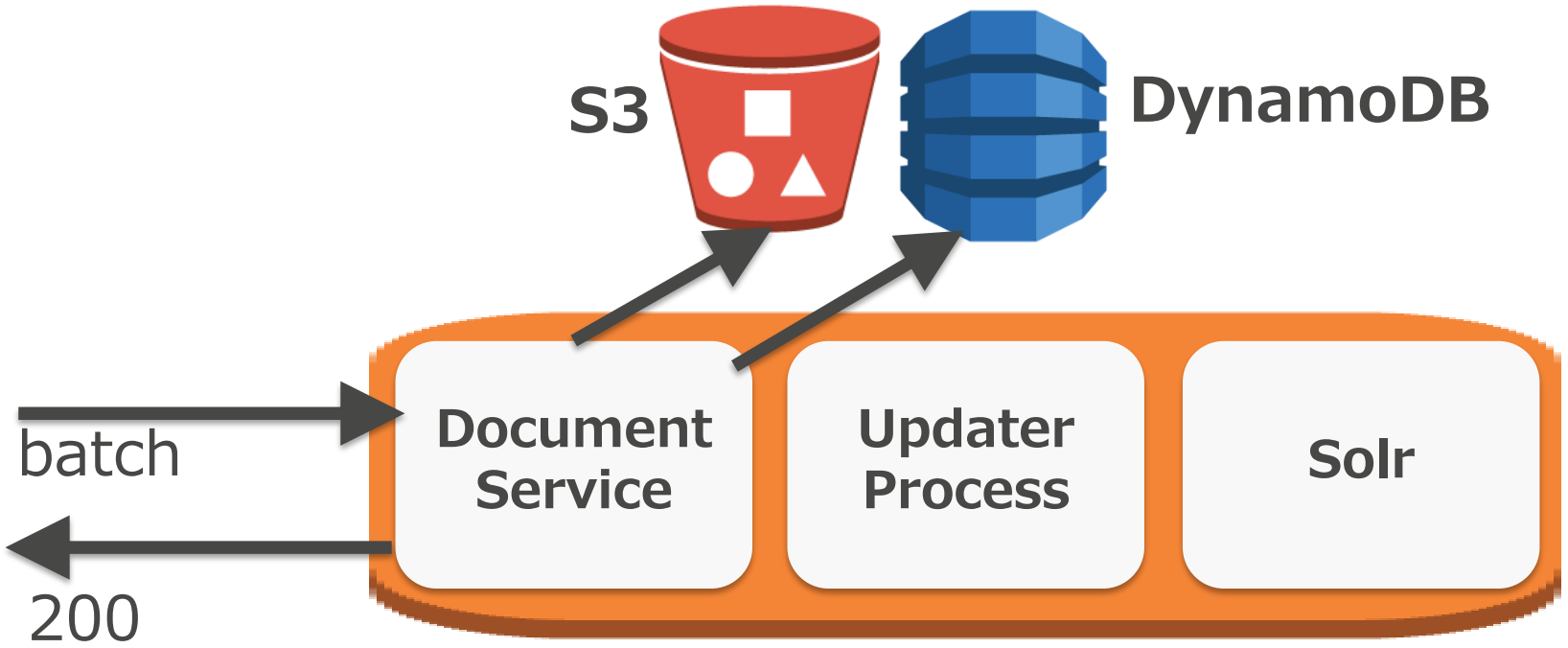
Inside Amazon CloudSearch

- Indexing
  - 全てのノードにELB経由で均等にアップロード



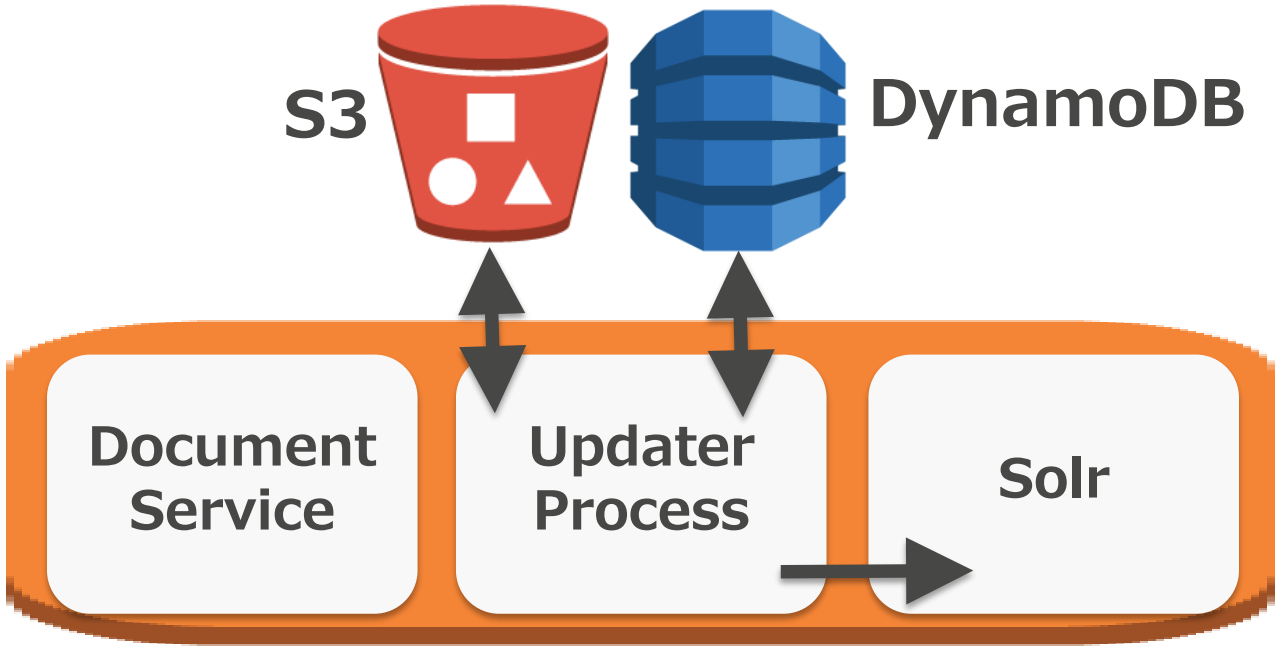
Inside Amazon CloudSearch

- Indexing
  - データを受け取ったノードは、
  - ファイルをS3に保存し、メタ情報をDynamoDBに保存した後、
  - クライアントにHTTPステータスコード200(正常終了)を返す



Inside Amazon CloudSearch

- Indexing
  - S3とDynamoDBにポーリングを行い、
  - Indexing対象があれば処理を行いS3に配置＆DynamoDBを更新
  - 自Partition担当のIndexバイナリがあればローカルのSolrへ配置



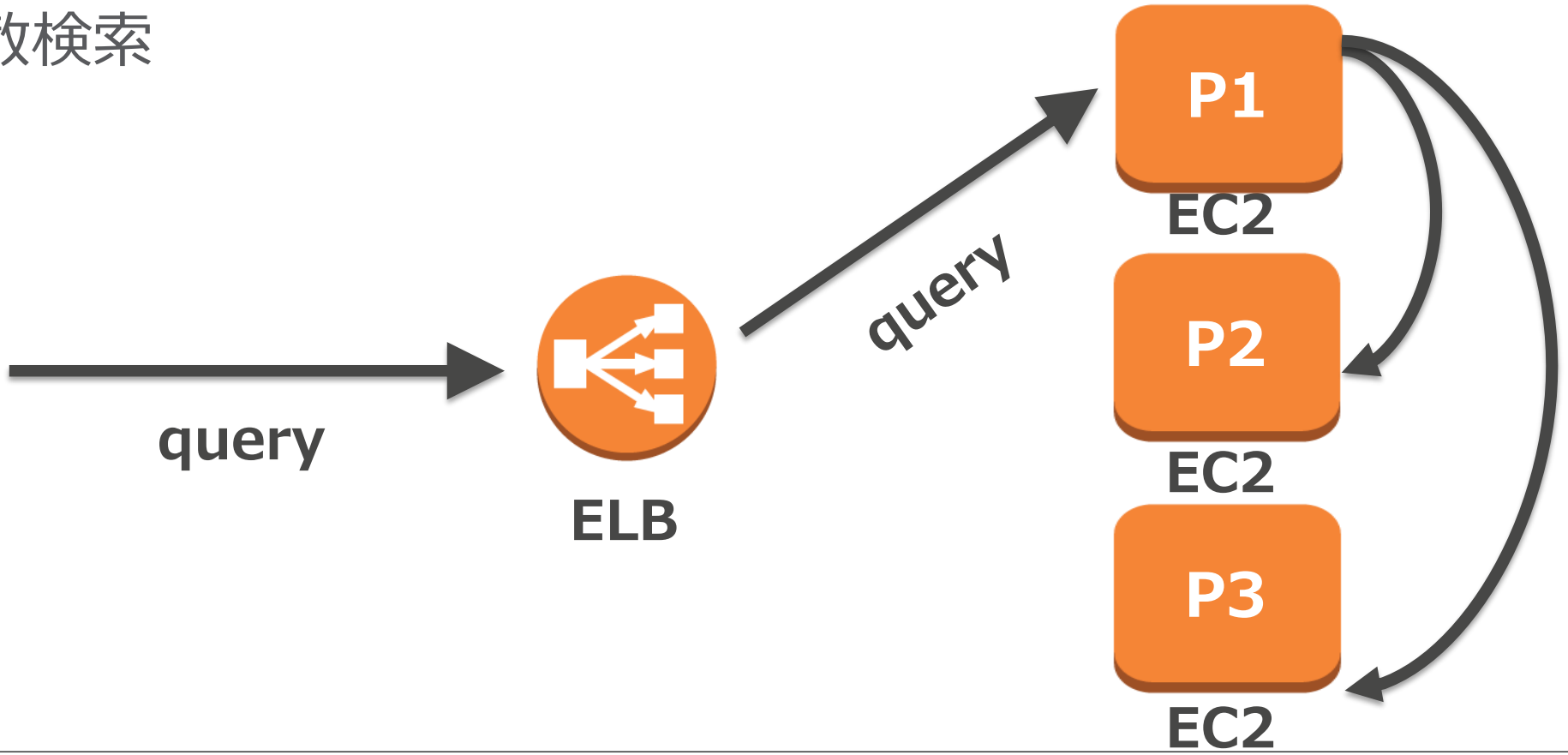
Inside Amazon CloudSearch

- Amazon Simple Storage Service (S3)

  - 高い堅牢性 99.999999999%
  - 格納容量無制限。利用した分のみ課金
  - 様々なAWSサービスと連携するセンターストレージ
- Amazon DynamoDB
  - 高い信頼性、スケーラビリティ、低レイテンシ、安定した性能を兼ね備えたNoSQLデータベースサービス
  - 必要スループットを決めるだけで利用可能。ストレージ容量は事前に決める必要がなく、自動的にプロビジョンされる

Inside Amazon CloudSearch

- Query
  - インデックス処理と同様にELBで均等に割り振り
  - 分散検索

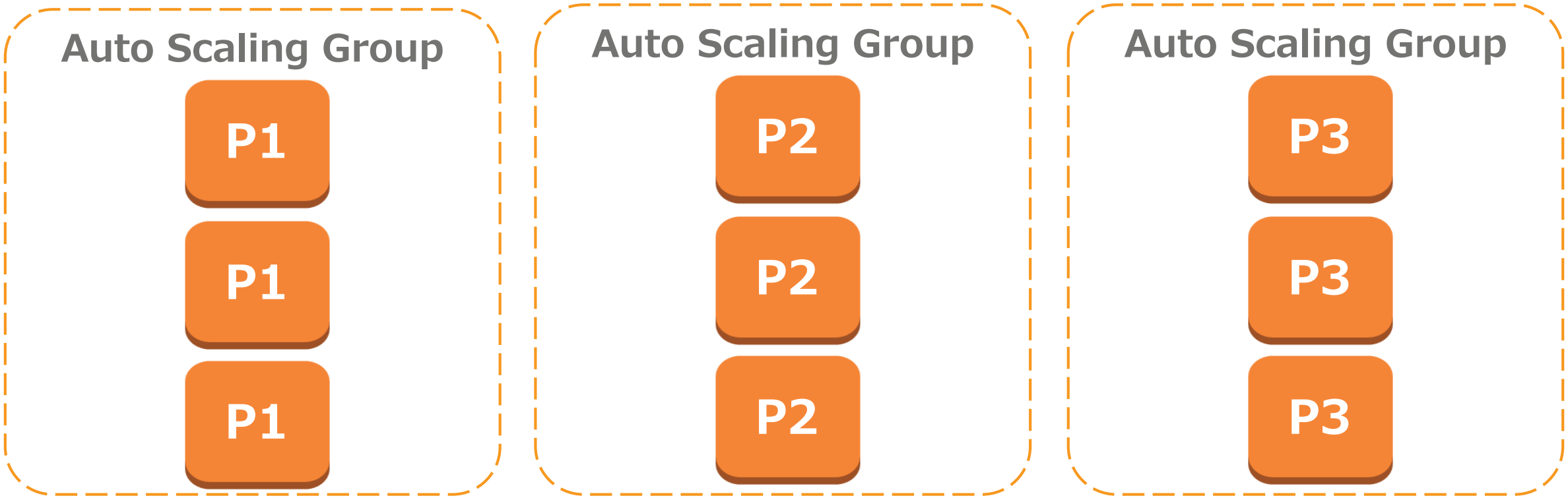


Inside Amazon CloudSearch

- Auto Scaling
  - 大量の検索リクエストをハンドリングする必要がある場合は、
  - AutoScaling(ELB + EC2)で対応

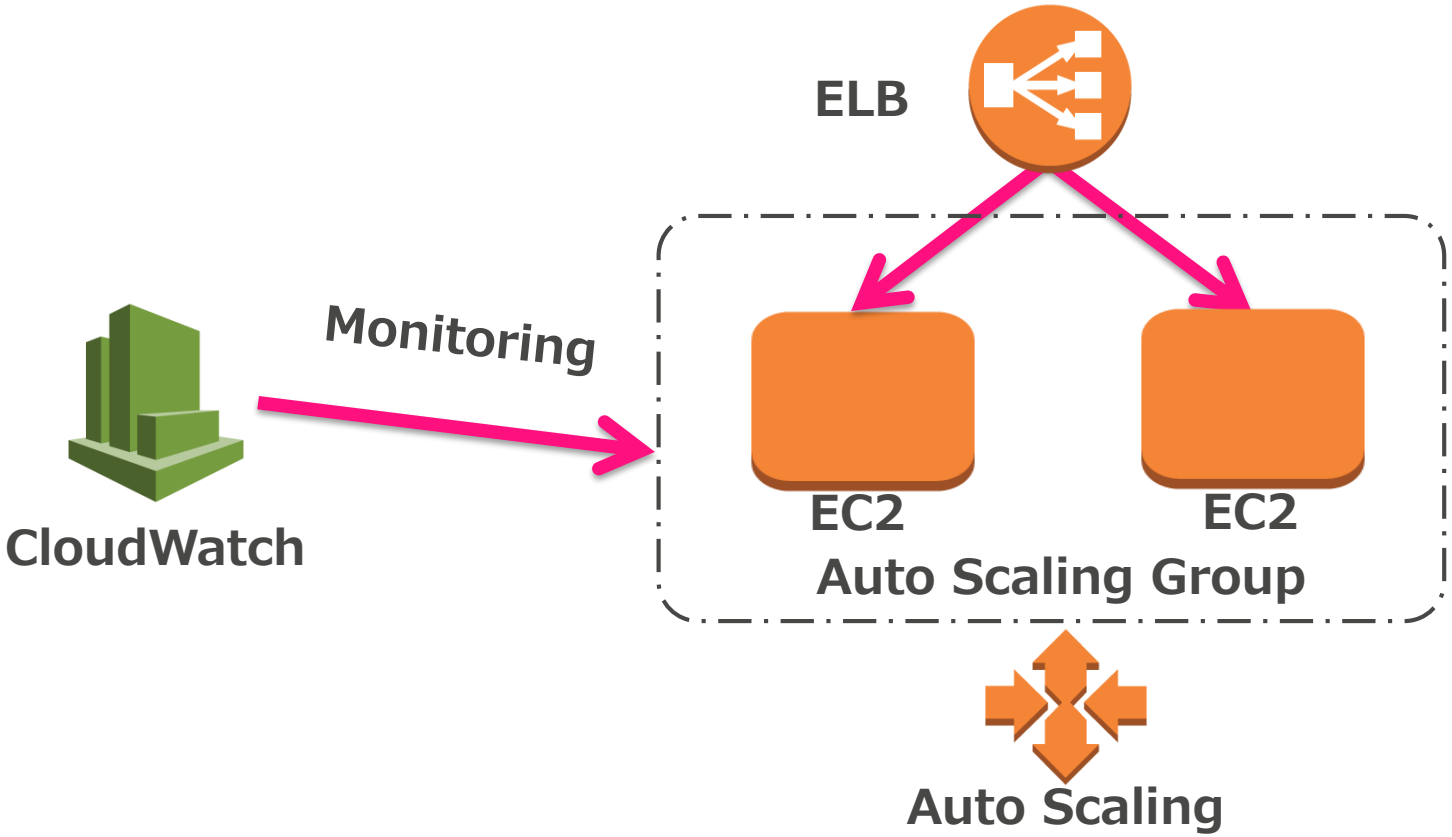
Inside Amazon CloudSearch

- Auto Scaling
  - 大量の検索リクエストをハンドリングする必要がある場合は、
  - AutoScaling(ELB + EC2)で対応



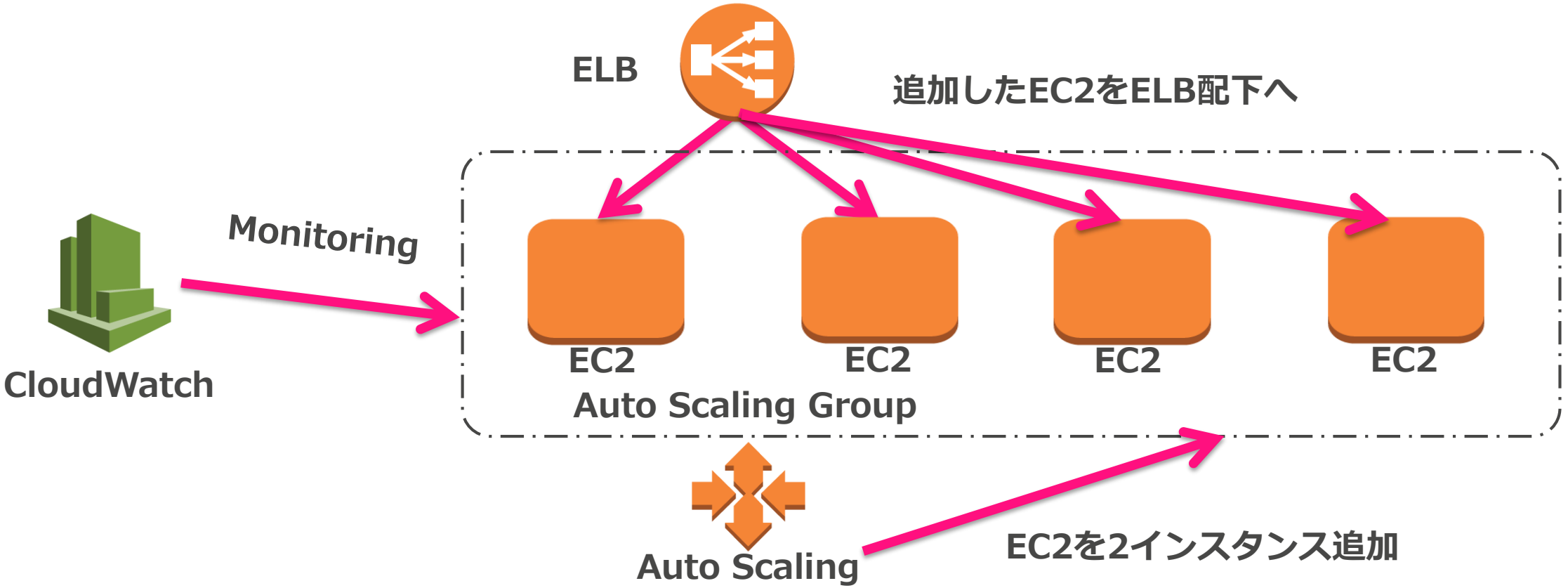
Inside Amazon CloudSearch

- Auto Scalingの仕組み
  - 例)CPU利用率が5分以上70%を超える場合、EC2を2インスタンス追加



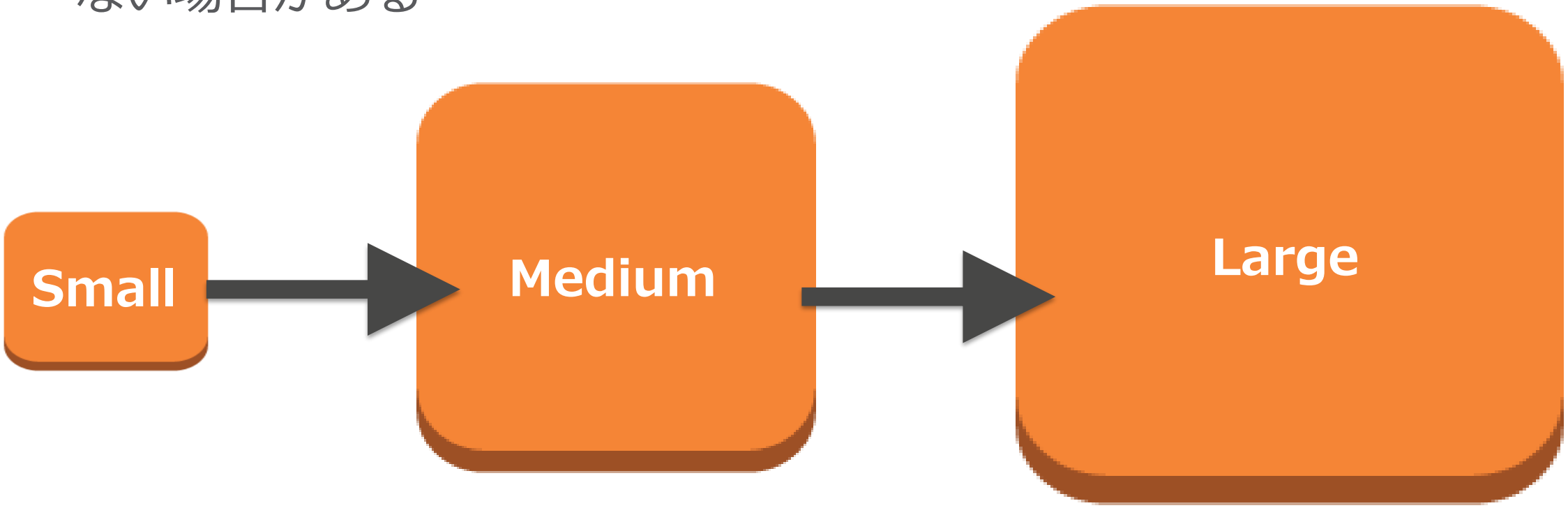
Inside Amazon CloudSearch

- Auto Scalingの仕組み
  - 例)CPU利用率が5分以上70%を超える場合、EC2を2インスタンス追加



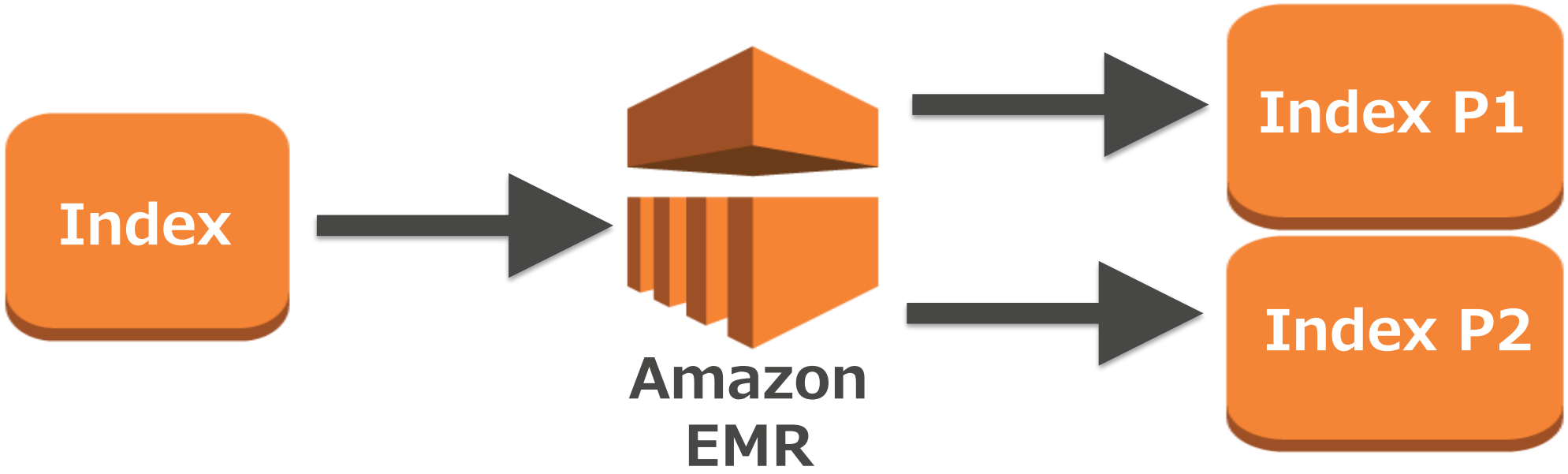
Inside Amazon CloudSearch

- Auto Partitioning
  - 最初はスケールアップ
  - ダウンタイムはないが切り替わり中はIndexing結果がすぐに反映されない場合がある



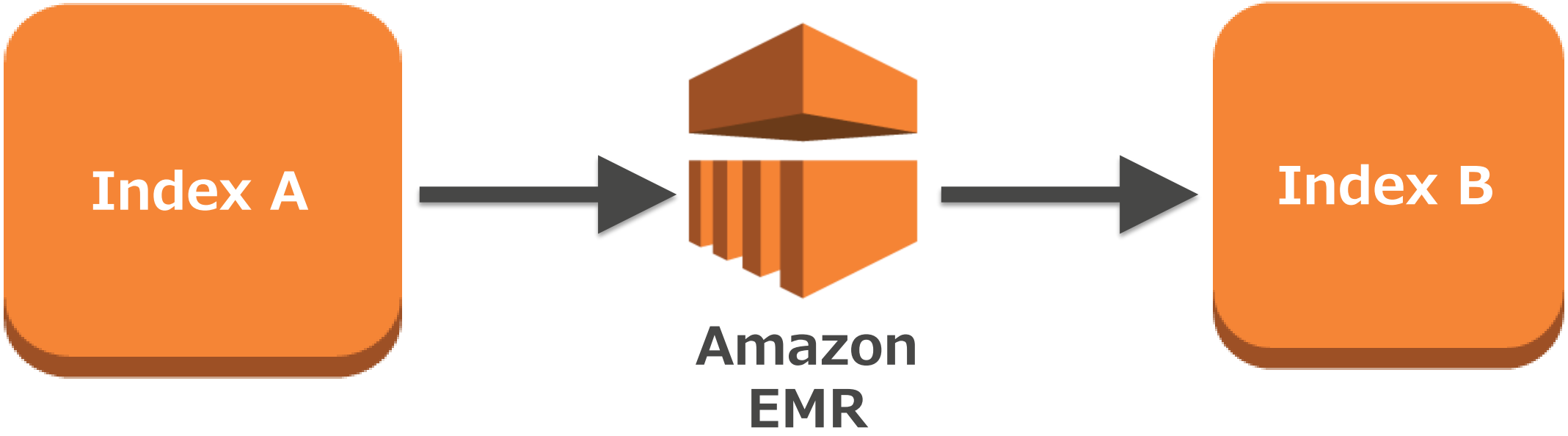
Inside Amazon CloudSearch

- Auto Partitioning
  - 1つのノードでまかないきれなくなったらスケールアウト
  - パーティションの分割処理はHadoop(EMR)で
  - ダウンタイムはないが結果整合性モデル
    - S3にインデックスファイルが置かれ各ノードがポーリング



Inside Amazon CloudSearch

- 検索ドメインの設定変更を行った場合
  - 変更に伴う処理はHadoop(EMR)で
  - ダウンタイムはないが結果整合性モデル
    - S3にインデックスファイルが置かれ各ノードがポーリング

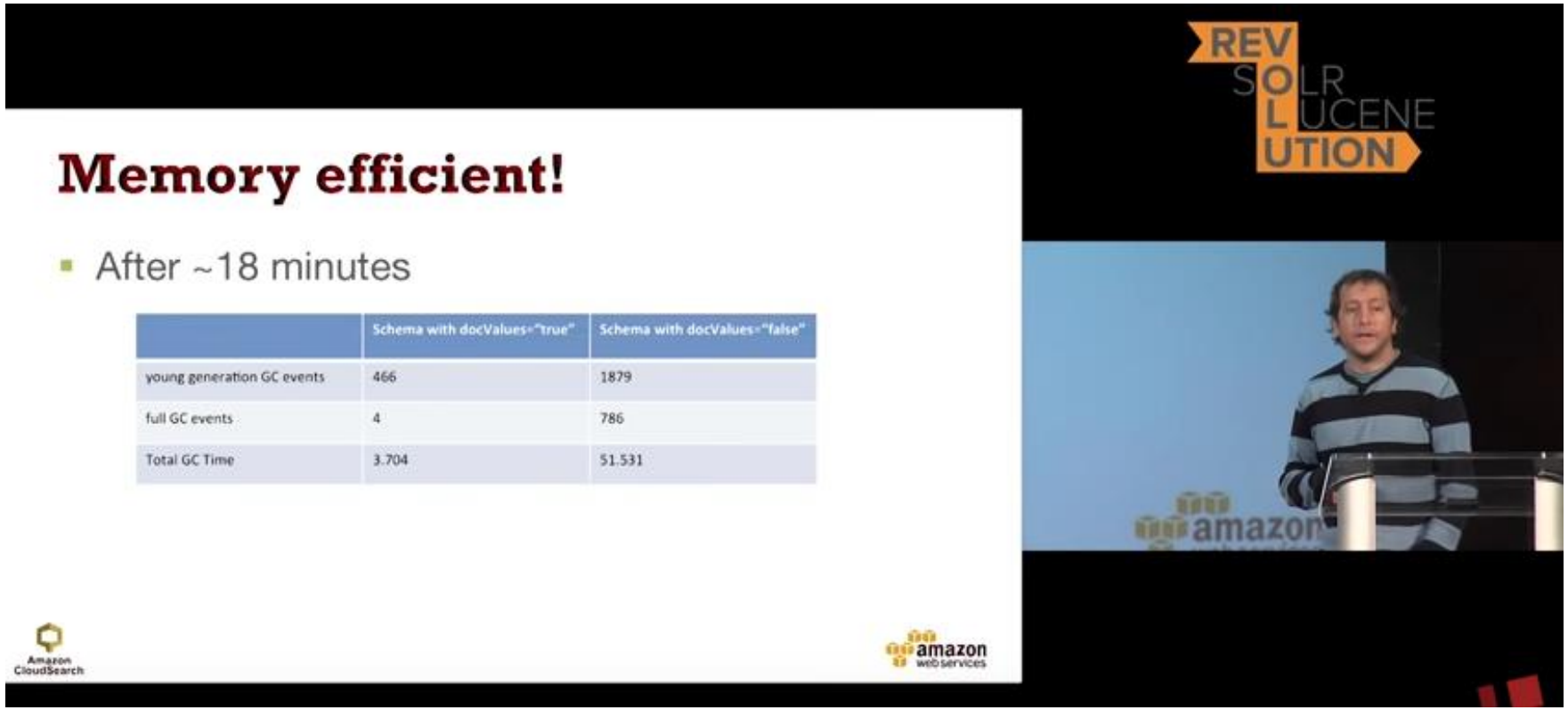


Amazon CloudSearch

- DocValues
  - CloudSearchではID,Facet,Sortといったところに使用
    - 具体的にはSolrで以下のように設定
    - <field name="foo" type="string" indexed="false" stored="false" **docValues="true"** />
  - Javaなミドルウェアでは Garbage Collection(GC)の扱いが重要
    - DocValuesはJavaVM(メモリ)からファイルシステムのキャッシュにオフロードするテクノロジー
  - Lucene 4.0で導入

Amazon CloudSearch

- DocValues
  - Lucene Solr RevolutionでのA9 Tomás Fernández Löbbeによる解説



https://www.youtube.com/watch?v=RI1x0d-yO8A

Amazon CloudSearch

- 日本発グローバル事例
  - Pixta 星さん(@NaoshiHoshi)の発表資料 @ Jaws Days 2016

http://www.slideshare.net/NaoshiHoshi/pixtacloudsearch-jaws-days-2016-lt

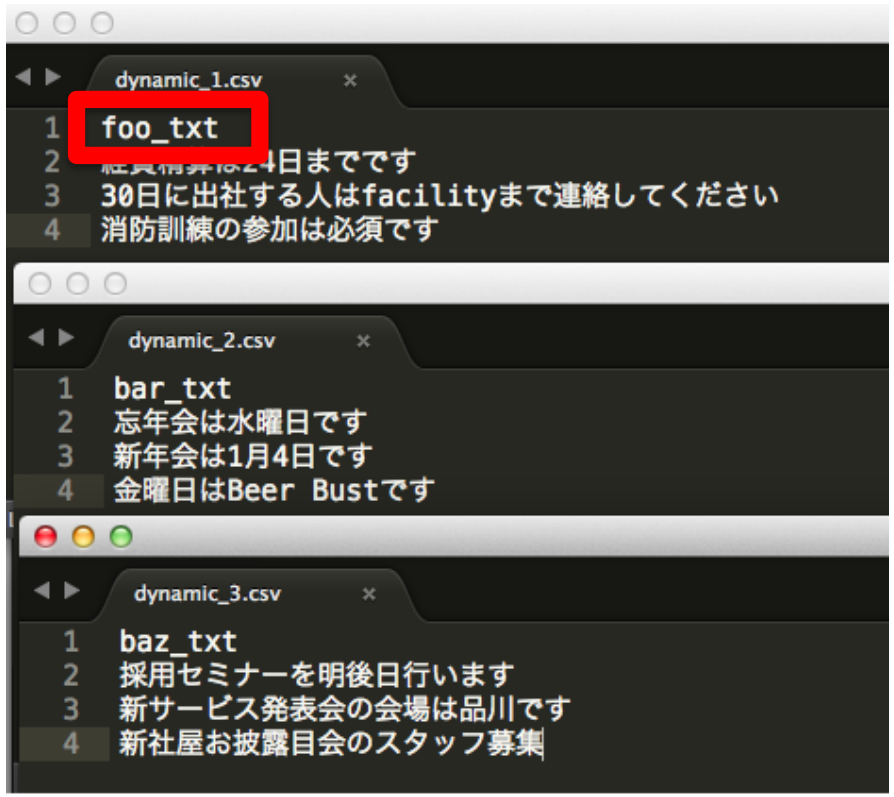
Amazon CloudSearch

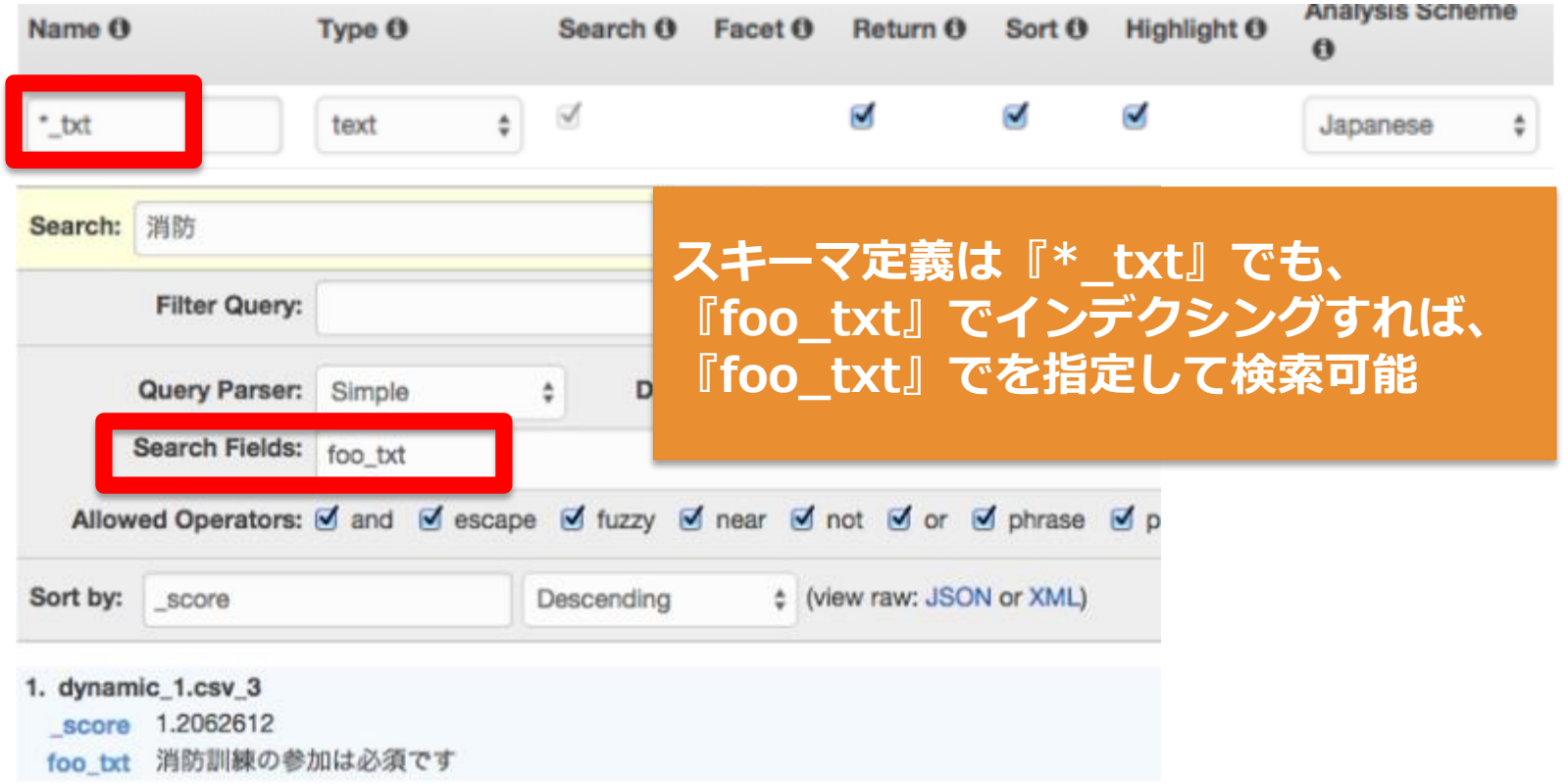
- 他にも様々な機能を備えています。詳細は以前のAWS Black Belt Tech Webinarの資料を是非ご覧ください

http://www.slideshare.net/AmazonWebServicesJapan/aws-black-belt-tech-amazon-cloudsearch

Amazon CloudSearch Update

- **Dynamic Fields のサポート – 2014年12月22日**
  - スキーマ変更せずともアプリケーション側で定義可能
    - 例) 複数の情報を *_txt というフィールドに格納





スキーマ定義は『*_txt』でも、『foo_txt』でインデクシングすれば、『foo_txt』でを指定して検索可能

Amazon CloudSearch Update

- **CloudWatchにドメインメトリクス追加 – 2015年3月5日**
  - SuccessfulRequests
    - 検索インスタンスにより正常に処理された検索リクエスト数
  - SearchableDocuments
    - 検索インデックスで利用できるドキュメント数
  - IndexUtilization
    - 検索インスタンスの検索ストレージ利用率
  - Partitions
    - 検索インデックスで利用できるパーティション数

Amazon CloudSearch Update

- **インデックスフィールド統計 – 2015年3月5日**
  - Apache SolrのStats componentと同等の機能
    - Facetが有効な数値フィールドで利用可能
  - 取得できる統計
    - Count
    - Min
    - Max
    - Mean
    - Missing
    - Stddev
    - Sum
    - sumOfSquares

Amazon CloudSearch

- **CloudSearchにいただいているご要望**
  - リアルタイムにデータを取り込んで可視化したい
    - CloudSearchは最大で5MBのバッチファイルでの連携
    - 細かいデータを高頻度でストアするのに強くない
  - Amazon CloudSearchの独自APIを習得するのに時間がかかる
    - Solrを使っているがSolrのAPIは利用できない
    - 完全マネージドでスケールと低レイテンシを同時に満たすには様々なSolrの機能をオープンにすることが難しい

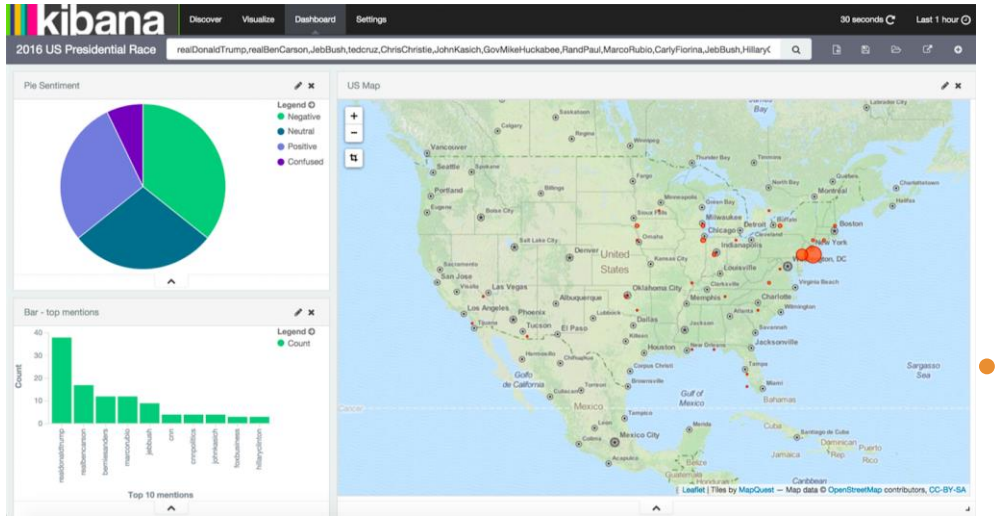
Amazon Elasticsearch Service

Amazon Elasticsearch Service

- Elasticsearchのマネージドサービス
  - AWSクラウド上で Elasticsearch を簡単に構築可能
    - Elasticsearchの分散/スケーリング機能はクラウドと相性が良い
    - インスタンスタイプと台数を選択するだけでプロビジョニング
  - デフォルトでKibanaがインストールされる
    - Management ConsoleにてURLをクリックするだけで直ぐ利用可能
  - 使った分だけの従量課金
    - ノードに利用するEC2の時間課金
    - EBSボリュームを使った場合はEBSの料金
  - 略称はAmazon ES

Amazon Elasticsearch Service

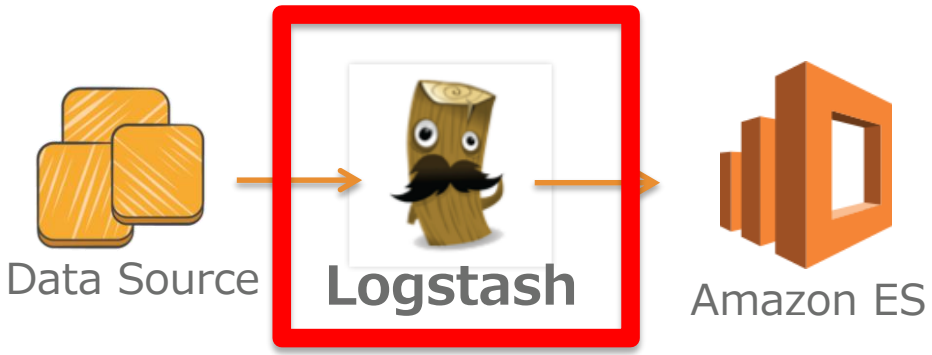
ELK(Elasticsearch, Logstash, Kibana)スタックをサポートしたマネージドAnalyticsサービス

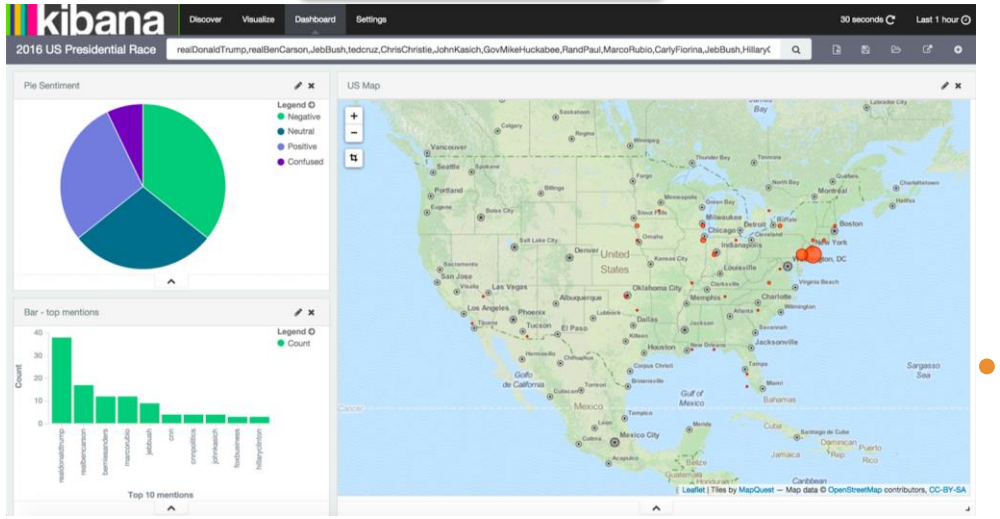


- **特徴** (https://aws.amazon.com/jp/elasticsearch-service/)
  - ElasticsearchのAPIをそのまま利用可能
  - AWSのサービスと連携した構成を簡単に構築
    例)
    - CloudWatch Logs -> Lambda -> Amazon ES
    - DynamoDB Streams -> Logstash -> Amazon ES
  - 検索ドメインを作成すると同時にKibanaが利用可能
  - 日本語解析に対応
    - Elasticsearch ICUプラグイン
    - Elasticsearch Kuromojiプラグイン
- **価格体系** (https://aws.amazon.com/jp/elasticsearch-service/pricing/)
  - Elasticsearchインスタンス時間
  - Amazon EBSストレージ

Amazon Elasticsearch Service

ELK(Elasticsearch, Logstash, Kibana)スタックをサポートしたマネージドAnalyticsサービス

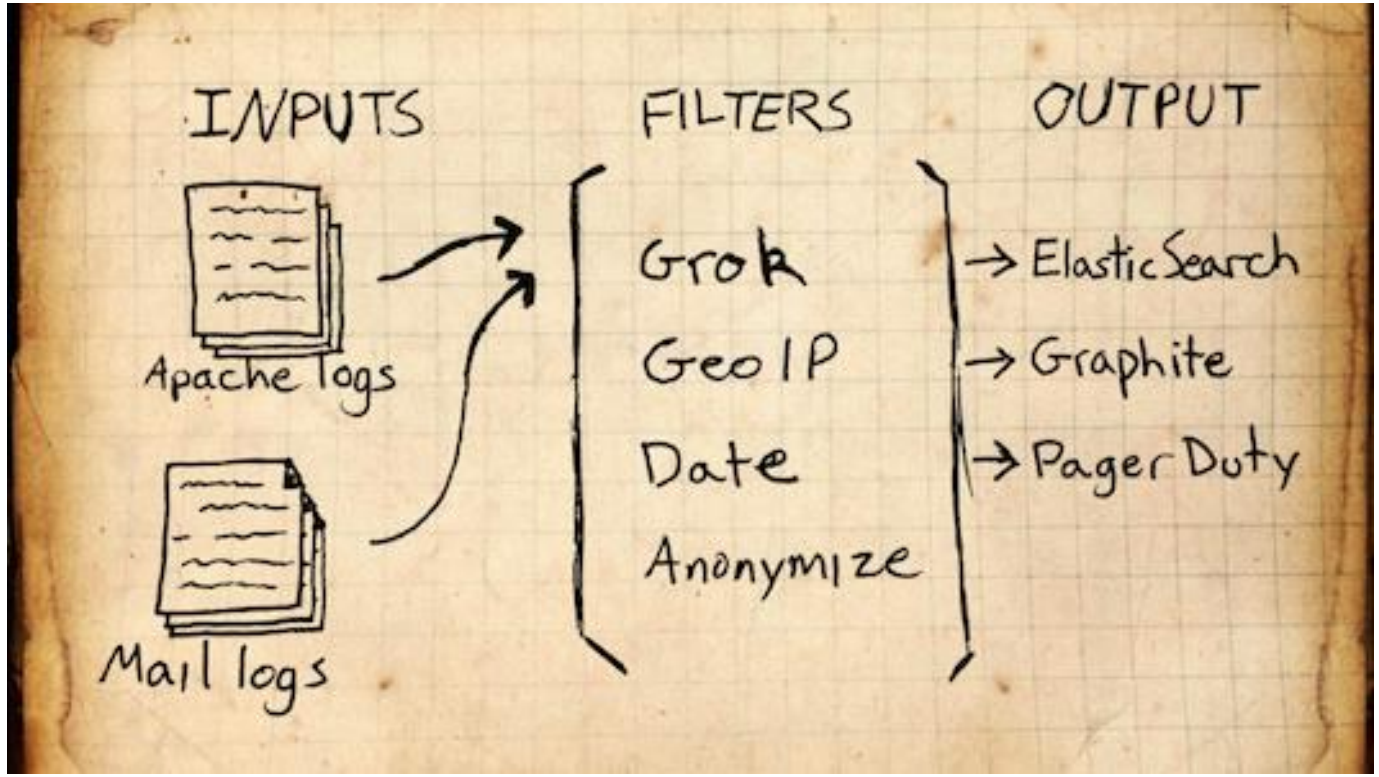




- **特徴** (https://aws.amazon.com/jp/elasticsearch-service/)
  - ElasticsearchのAPIをそのまま利用可能
  - AWSのサービスと連携した構成を簡単に構築
    例)
    - CloudWatch Logs -> Lambda -> Amazon ES
    - DynamoDB Streams -> Logstash -> Amazon ES
  - 検索ドメインを作成すると同時にKibanaが利用可能
  - 日本語解析に対応
    - Elasticsearch ICUプラグイン
    - Elasticsearch Kuromojiプラグイン
- **価格体系** (https://aws.amazon.com/jp/elasticsearch-service/pricing/)
  - Elasticsearchインスタンス時間
  - Amazon EBSストレージ

Logstash

- イベントの処理フロー



https://www.elastic.co/downloads/logstash

Logstash

- **便利なFilter機能**
  - 様々なパターンが登録されている
    - **https://github.com/logstash-plugins/logstash-patterns-core/blob/master/patterns/grok-patterns**

```
1   USERNAME [a-zA-Z0-9._-]+
2   USER %{USERNAME}
3   EMAILLOCALPART [a-zA-Z][a-zA-Z0-9_.+-=:]+
4   EMAILADDRESS %{EMAILLOCALPART}@%{HOSTNAME}
5   HTTPDUSER %{EMAILADDRESS}|%{USER}
6   INT (?:[+-]?(?:[0-9]+))
7   BASE10NUM (?<![0-9.+-])(?>[+-]?(?:(?:[0-9]+(?:\.[0-9]+)?)|(?:\.[0-9]+)))
8   NUMBER (?:%{BASE10NUM})
～略～
100 # Log Levels
101 LOGLEVEL ([Aa]lert|ALERT|[Tt]race|TRACE|[Dd]ebug|DEBUG|[Nn]otice|NOTICE|[Ii]nfo|
```

Logstash

- **便利なFilter機能**
  - 日付系

```
DATE_US %{MONTHNUM}[/-]%{MONTHDAY}[/-]%{YEAR}
DATE_EU %{MONTHDAY}[./-]%{MONTHNUM}[./-]%{YEAR}
ISO8601_TIMEZONE (?:Z|[+-]%{HOUR}(?::?%{MINUTE}))
ISO8601_SECOND (?:%{SECOND}|60)
TIMESTAMP_ISO8601 %{YEAR}-%{MONTHNUM}-
%{MONTHDAY}[T ]%{HOUR}:?%{MINUTE}(?::?%{SECOND})?%{ISO8601_TIMEZONE}?
DATE %{DATE_US}|%{DATE_EU}
DATESTAMP %{DATE}[- ]%{TIME}
TZ (?:[PMCE][SD]T|UTC)
DATESTAMP_RFC822 %{DAY} %{MONTH} %{MONTHDAY} %{YEAR} %{TIME} %{TZ}
DATESTAMP_RFC2822 %{DAY}, %{MONTHDAY} %{MONTH} %{YEAR} %{TIME} %{ISO8601_TIMEZONE}
DATESTAMP_OTHER %{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{TZ} %{YEAR}
DATESTAMP_EVENTLOG %{YEAR}%{MONTHNUM2}%{MONTHDAY}%{HOUR}%{MINUTE}%{SECOND}
HTTPDERROR_DATE %{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{YEAR}
```

Logstash

- **便利なFilter機能**
  - IPアドレスから地域を導出

```
"geoip" => {
                    "ip" => "183.60.215.50",
          "country_code2" => "CN",
          "country_code3" => "CHN",
           "country_name" => "China",
         "continent_code" => "AS",
            "region_name" => "30",
              "city_name" => "Guangzhou",
               "latitude" => 23.11670000000001,
              "longitude" => 113.25,
               "timezone" => "Asia/Chongqing",
```

Logstash

- **便利なFilter機能**
  - Apacheの一般的なログ: COMMONAPACHELOG
  - **199.72.81.55 - - [01/Nov/2015:00:00:01 -0400] "GET /yo/ HTTP/1.0" 200 624**
    - %{IPORHOST:clientip}
    - %{HTTPDUSER:ident}
    - %{USER:auth}
    - ¥[%{HTTPDATE:timestamp}¥]
    - "(?:%{WORD:verb} %{NOTSPACE:request}(?: HTTP/%{NUMBER:httpversion})?|%{DATA:rawrequest})"
    - %{NUMBER:response}
    - (?:%{NUMBER:bytes}|-)

Logstash

- **便利なFilter機能**
  - Apacheの一般的なログ: COMMONAPACHELOG
  - **199.72.81.55 - - [01/Nov/2015:00:00:01 -0400] "GET /yo/ HTTP/1.0" 200 624**

```
filter {
    grok {
        match => { "message" => "%{COMMONAPACHELOG}" }
    }
    date {
        match => [ "timestamp", "dd/MMM/YYYY:HH:mm:ss Z" ]
        locale => en
    }
}
```

@timestampを現在の時間ではなく
ログに記述された時間にする

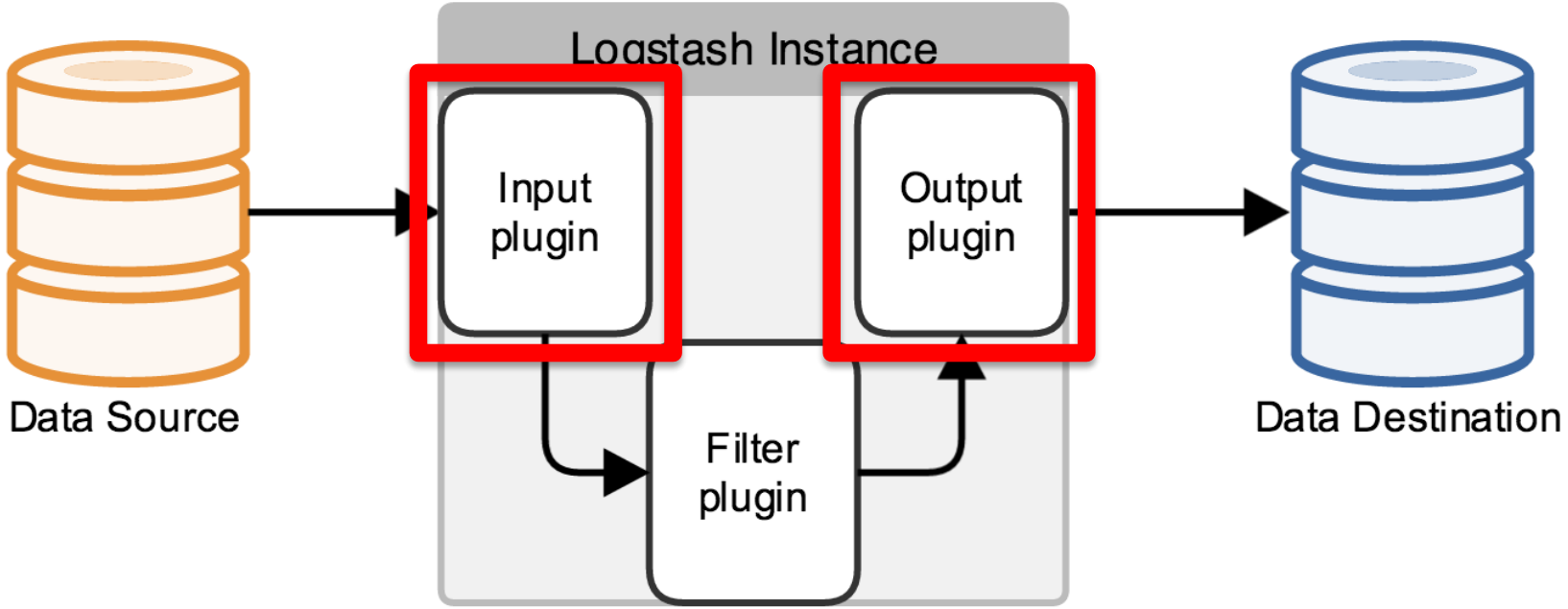
Logstash

- **便利なFilter機能**
  - Apacheの一般的なログ: COMMONAPACHELOG
  - **199.72.81.55 - - [01/Nov/2015:00:00:01 -0400] "GET /yo/ HTTP/1.0" 200 624**

```
{
       "message" => "199.72.81.55 - - [01/Nov/2015:00:00:01 -0400] ¥"GET /yo/ HTTP/1.0¥" 200 624",
      "@version" => "1",
    "@timestamp" => "2015-11-01T04:00:01.000Z",
          "host" => "ip-172-31-20-185",
      "clientip" => "199.72.81.55",
         "ident" => "-",
          "auth" => "-",
     "timestamp" => "01/Nov/2015:00:00:01 -0400",
          "verb" => "GET",
       "request" => "/yo/",
   "httpversion" => "1.0",
      "response" => "200",
         "bytes" => "624”
}
```

Amazon Elasticsearch Service

- **Logstash – AWSプラグイン**
  - AWSのクレデンシャルを使ってセキュアにAmazonESにデータ投入



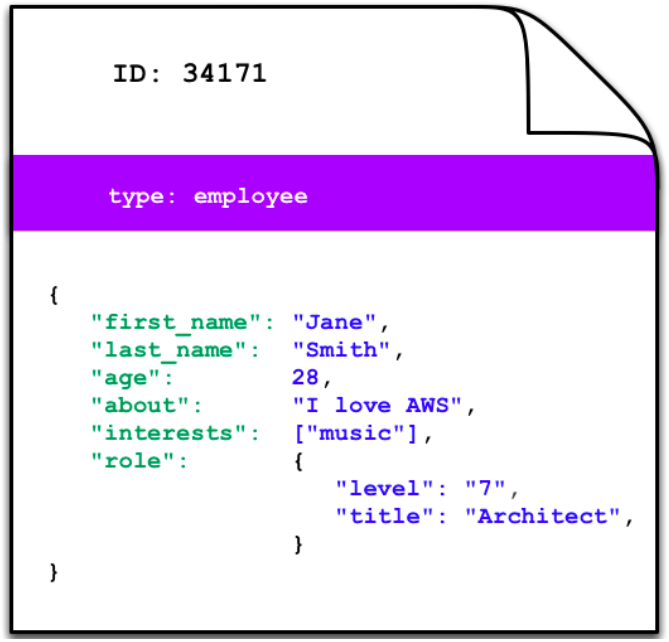
https://www.elastic.co/guide/en/logstash/current/advanced-pipeline.html

Amazon Elasticsearch Service

- **Logstash – AWSプラグイン**
  - **Input**
    - **S3 input プラグイン**
      - バケットとファイル名のパターンを指定
      - https://www.elastic.co/guide/en/logstash/current/plugins-inputs-s3.html
    - **DynamoDB input プラグイン**
      - DynamoDB Streamsのデータを読み込み
      - https://github.com/awslabs/logstash-input-dynamodb
  - **Output**
    - **Amazon Elasticsearch Searvice output プラグイン**
      - セキュアにAmazonESにインデクシング
      - https://github.com/awslabs/logstash-output-amazon_es

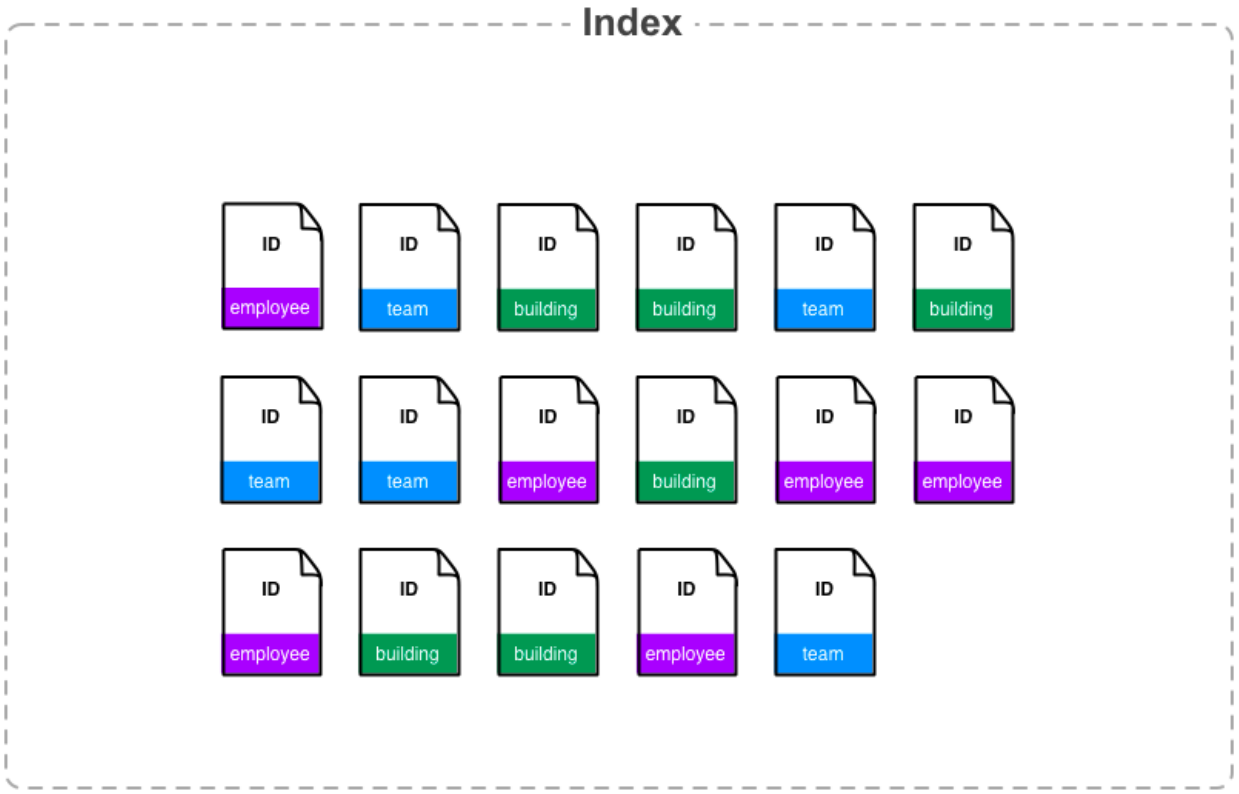
Amazon Elasticsearch Service

- ElasticsearchのTerminology
  - Document
    - ユニークなIDを持つ (RDBMSの row に近い)
    - Field を持つ (RDBMSの column に近い)
    - 同じ Field の Document の集合体を Type (RDBMSの table に近い)



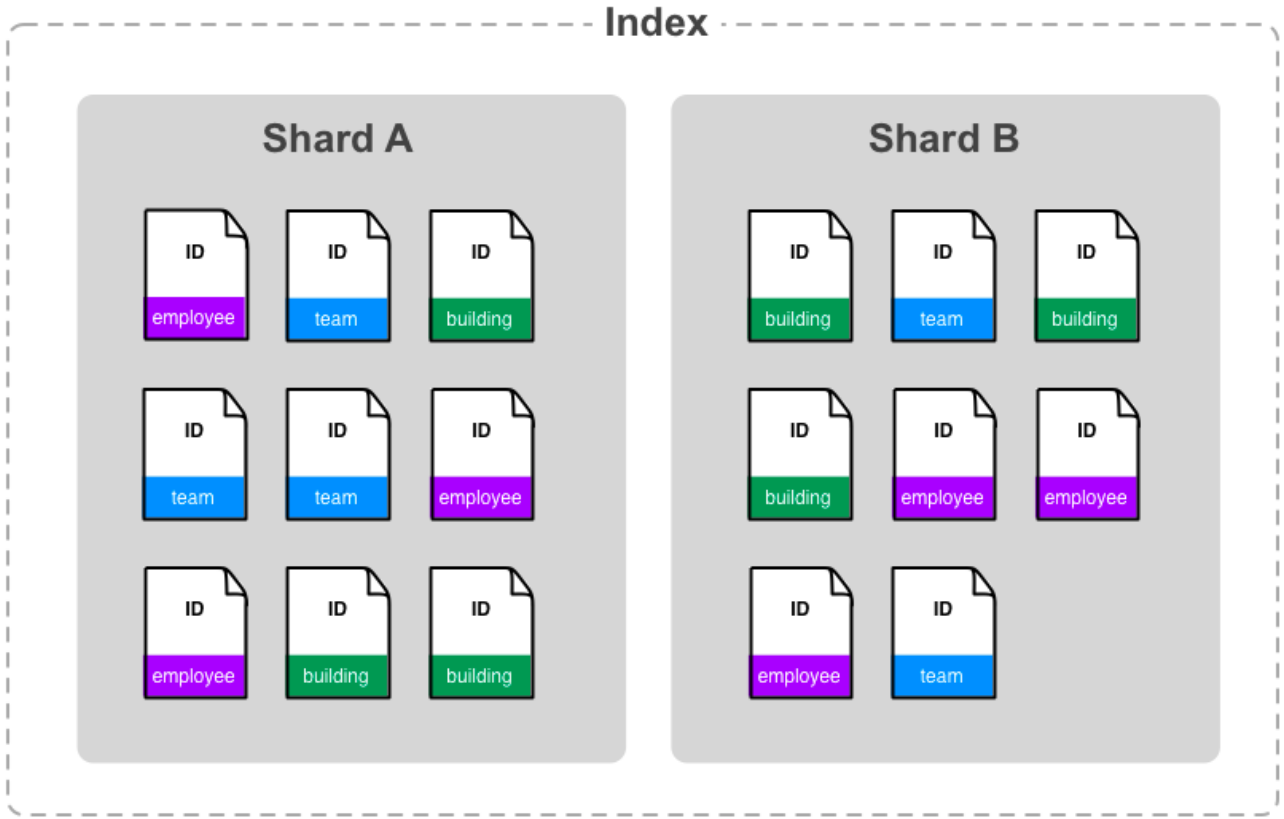
Amazon Elasticsearch Service

- ElasticsearchのTerminology
  - Index
    - Document の集合体。RDBMSの database に近い



Amazon Elasticsearch Service

- ElasticsearchのTerminology
  - Shard
    - Document は Index 内の複数の Shard に分散して配置される

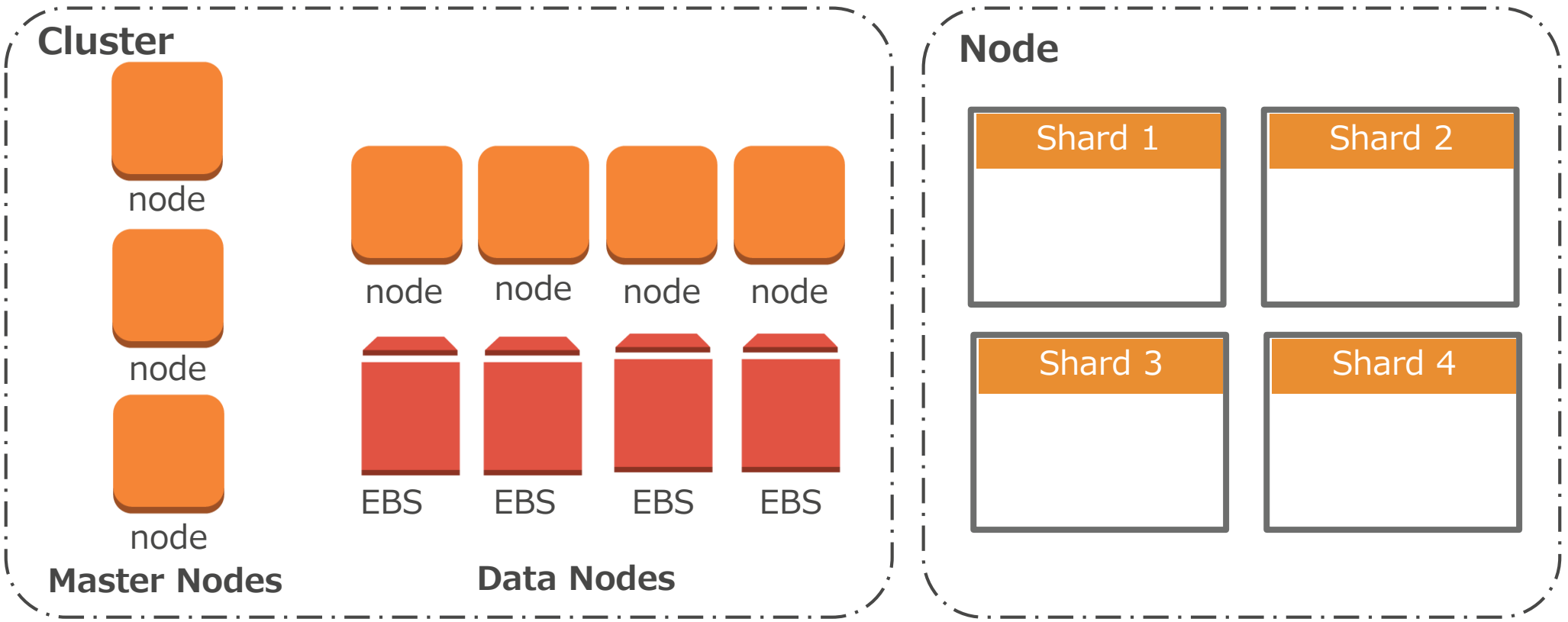


Amazon Elasticsearch Service

- ElasticsearchのTerminology
  - Node
    - 物理/仮想インスタンス
    - 複数のShardを保持
  - Cluster
    - 1つもしくは複数のノードからなる
    - 複数の Index を保持することができる
  - Amazon ESにおける domain
    - Managed Elasticsearch Cluster
    - CloudSearchは1domainに1つのschema定義のみ可能であるが、AmazonESであれば複数のIndexやTypeが定義できる

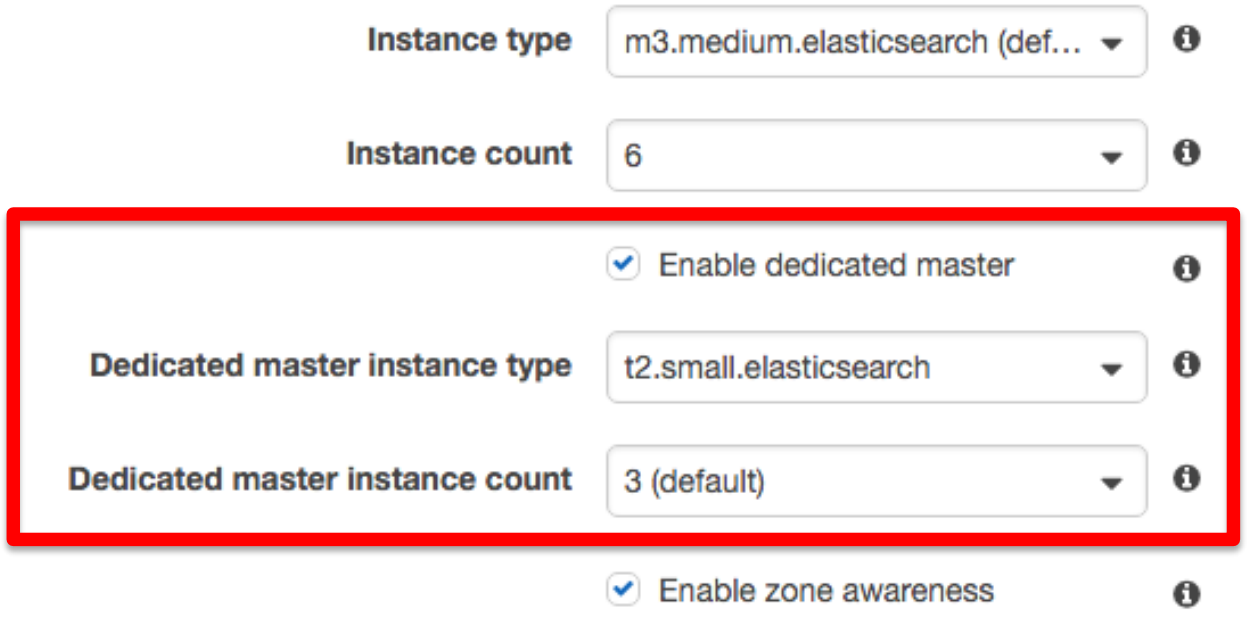
Amazon Elasticsearch Service

- Amazon ES の Deployment
  - 各ノードEBSボリュームはサイズ指定可能(2016年3月現在 最大512GB)
  - ノードの最大数は10(2016年3月現在)



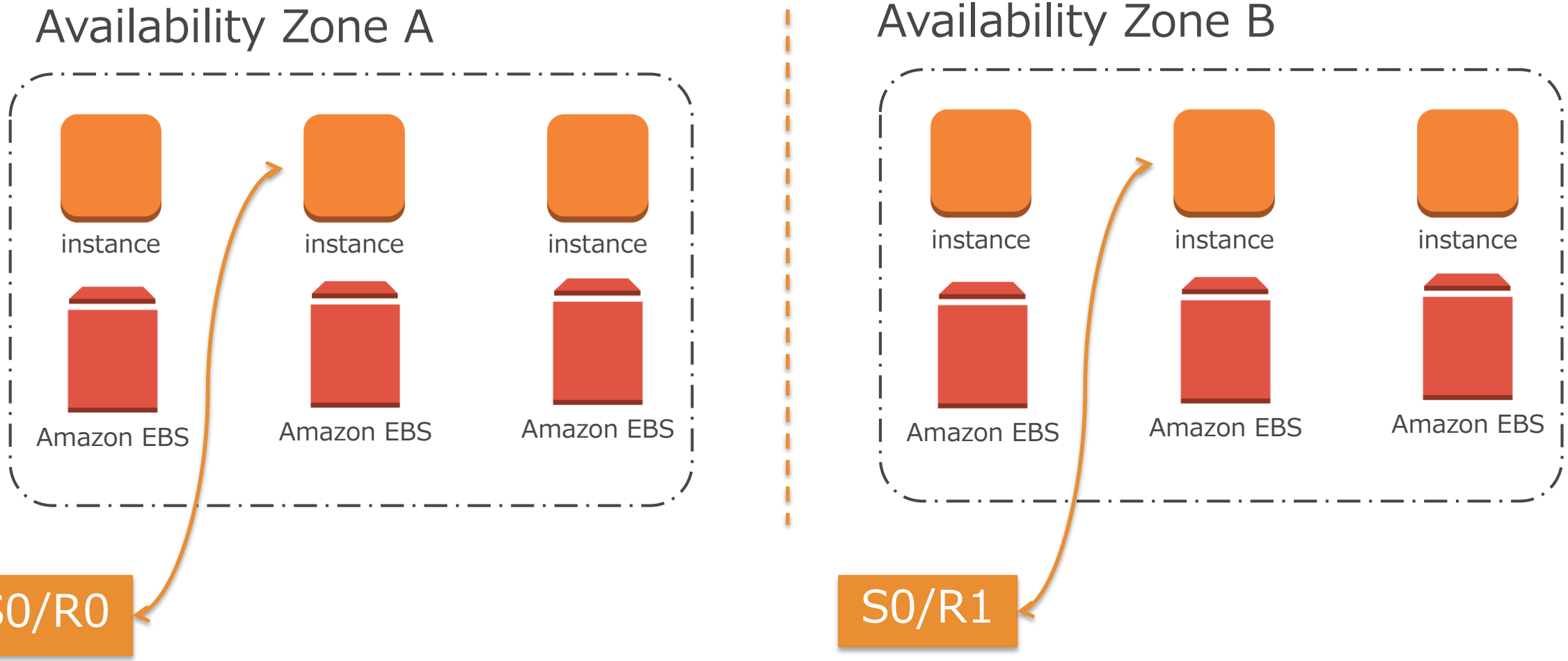
Amazon Elasticsearch Service

- dedicated master nodes
  - ノードを管理するマスター専用ノード
    - Split Brainを考慮して3台構成がオススメ
      - https://www.elastic.co/guide/en/elasticsearch/reference/1.5/modules-node.html



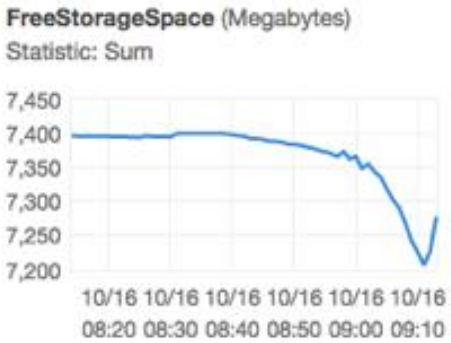
Amazon Elasticsearch Service

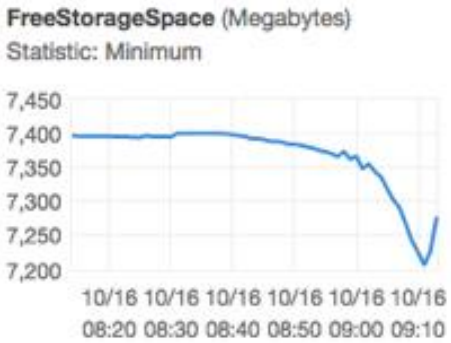
- Zone awareness - 複数のAZを使う設定



Amazon Elasticsearch Service

- Amazon ES の Deployment の注意点
  - CloudSearchのように自動的にスケーリングはしない
    - 構成は後から簡単に変更することは可能
  - プラグインを自由にインストールすることはできない
    - マネージドサービスである為、セキュリティや可用性担保の必要性
    - 商用プラグインに該当するような機能はIAMやCloudWatchで
  - 各種メトリクスを見ながらご自身で対応する必要がある





Amazon Elasticsearch Service

- Supported APIs

_alias
_aliases
_all
_analyze
_bulk
_cat
_cluster/health
_cluster/settings
for three properties(PUT only):
indices.breaker.fielddata.limit
indices.breaker.request.limit
indices.breaker.total.limit

_cluster/stats
_count
_flush
_mapping
_mget
_msearch
_nodes
_plugin/kibana
_plugin/kibana3

_percolate
_refresh
_search
_snapshot
_stats
_status
_template

Amazon Elasticsearch Service

- Mapping: Typeの定義
  - データ構造について – AWSで始めるElasticSearch(4)
  - http://dev.classmethod.jp/cloud/aws/use-elasticsearch-4-data-structure/

(左: 私、右: ブログ著者の佐々木さん)
先日、サンフランシスコで開催された
Elasticsearchのカンファレンスで
ご一緒させていただきました

Amazon Elasticsearch Service

- Mapping: Typeの定義
  - ElasticsearchはRestfulなAPIとJSONが基本

```
$ curl -XPUT https://<<DomainのURL>>/<<Index>>/<<Type>>/_mapping -d '{
>     "<<Type>>": {
>         "properties“ : {
>             "hoge"     : { "type": "string" },
>             “user”     : { “type”: “nested“ …略… },
>             "aaaa"     : { "type": "long" },
>             "bbbb"     : { "type": "boolean" },
>             "ccccc"    : { "type": "ip" },
>             "location": { "type": "geo_point" },
>             "dd_date": { "type":"date", "format":"yyyy/MM/dd"}
>         }
>     }
}‘
{"ok":true,"acknowledged":true}
```

ネストしたJSONもOK
"user" : [
  {"first" : "John", "last" : "Smith"},
  {"first" : "Alice", "last" : "White"}
]

位置情報も
"location": {"lat": 41.12, "lon": -71.34}

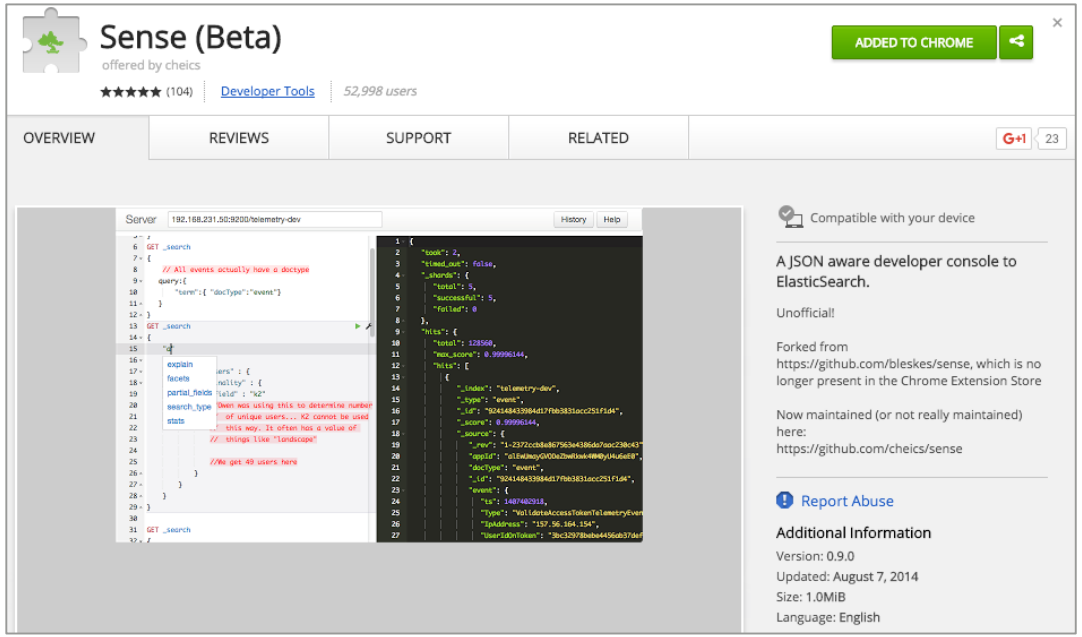
Amazon Elasticsearch Service

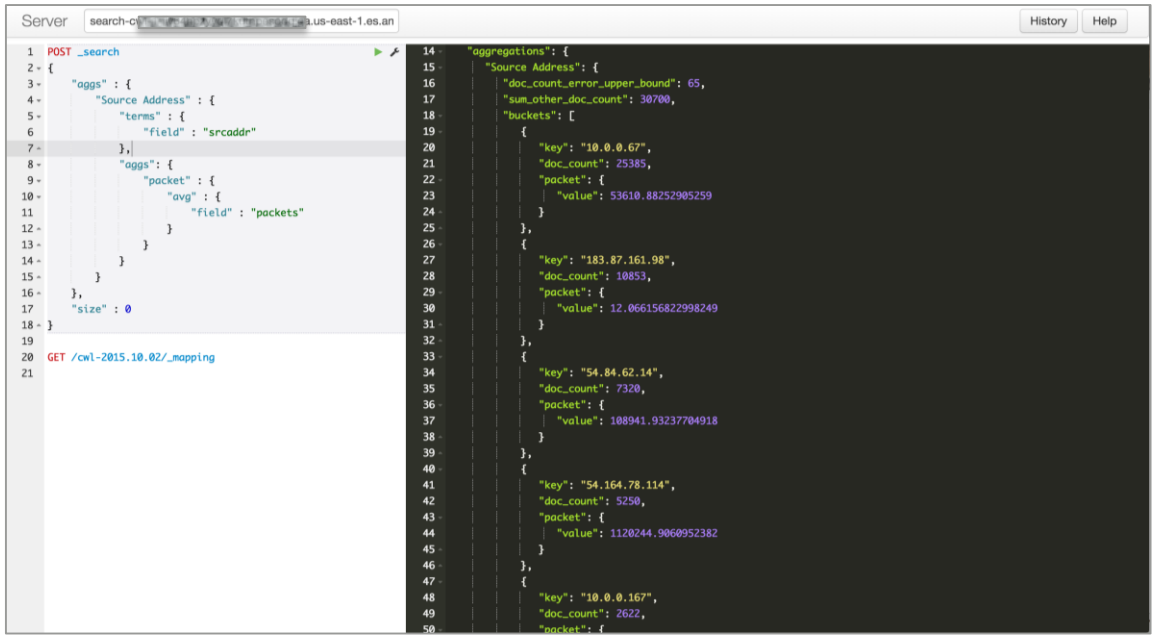
- Mapping: Typeの定義
  - Documentの登録

```
$ curl -XPUT https://<<DomainのURL>>/<<Index>>/<<Type>>/1 -d '{
>             "hoge"    :  "foo" ,
>             "user"    :  [ {"first" : "John", "last" : "Smith"}, {"first" : "Alice", "last" : "White"} ],
>             "aaaa"    :  99999999999,
>             "bbbb"    :  true,
>             "ccccc"   :  "192.168.1.1",
>             "location":  “41.12,-71.34”,
>             "dd_date":  "`/bin/date +%Y/%m/%d`"}
}‘
{"ok":true,"_index":”<<Index>>","_type":”<<Type>>","_id":"1","_version":1}
```

Amazon Elasticsearch Service

- コンソールでJSONをやりくりするのは辛い…？
  - 商用プラグインはAmazon ESでは使えない
  - SenseのChromeプラグイン
    - ※ オフィシャルなプラグインではないのでご利用は自己責任で ※





https://chrome.google.com/webstore/detail/sense-beta/lhjgkmllcaadmopgmanpapmpjgmfcfig?hl=en

Amazon Elasticsearch Service

- Amazon ESのCloudWatch Metrics

- ❑ Cluster status(Green/Yellow/Red)
- ❑ Node count
- ❑ Searchable documents
- ❑ Deleted documents
- ❑ CPU utilization
- ❑ Free storage space
- ❑ JVM memory pressure
- ❑ Automatic snapshot failures
- ❑ Master CPU utilization
- ❑ Master free storage space
- ❑ Master JVM memory pressure
- ❑ Read IOPS
- ❑ Write IOPS
- ❑ Read latency
- ❑ Write latency
- ❑ Read throughput
- ❑ Write throughput
- ❑ Disk queue depth

Amazon Elasticsearch Service

- DocValues
  - Amazon ESのElasticsearchのバージョンは1.5.2
  - Elasticsearch 2.0のリリースノートにはDoc Valuesがデフォルトになることが書かれている
    - https://www.elastic.co/guide/en/elasticsearch/reference/2.1/release-notes-2.0.0-beta1.html

Amazon Elasticsearch Service

- DocValues
  - Amazon ESのElasticsearchのバージョンは1.5.2
  - CloudWatchのJVM memory pressureを見ながらインスタンスタイプを調整していただきつつ、必要に応じてdoc_valuesを設定

```
PUT /my_index
{
  "mappings": {
    "my_type": {
      "properties": {
        "timestamp": {
          "type":       "date",
          "doc_values": true
        }
      }
    }
  }
}
```

https://www.elastic.co/blog/elasticsearch-1-4-0-beta-released

Amazon Elasticsearch Service

- 日本語解析
  - ICUとKuromojiプラグインはAmazon ESにインストール済み
    - analysis-icu(ノーマライズ)
    - analysis-kuromoji(形態素解析)
      - ユーザー辞書の追加機能は現在(2016年3月)開発中

- ICU Analysis Plugin
  - ICU Normalization Character Filter
  - ICU Tokenizer
  - ICU Normalization Token Filter
  - ICU Folding Token Filter
  - ICU Collation Token Filter
  - ICU Transform Token Filter

- Japanese (kuromoji) Analysis Plugin
  - `kuromoji` analyzer
  - `kuromoji_iteration_mark` character filter
  - `kuromoji_tokenizer`
  - `kuromoji_baseform` token filter
  - `kuromoji_part_of_speech` token filter
  - `kuromoji_readingform` token filter
  - `kuromoji_stemmer` token filter

Amazon Elasticsearch Service

- ICU: International Components for Unicode
  - ICUで出来ること => Unicodeの正規化
  - 具体的には
    - ㎏ => キログラム
    - ① => 1
  - 使い方の詳細は↓のREADMEを参照
    - https://github.com/elastic/elasticsearch-analysis-icu/blob/master/README.md
  - ICUのホームページ
    - http://site.icu-project.org/
    - http://icu-project.org/apiref/icu4j/

ICU4J

com.ibm.icu » icu4j » 56.1 under I18N Libraries

International Component for Unicode for Java (ICU4J) is a mature, Unicode and Globalization support

Amazon Elasticsearch Service

- Kuromoji
  - https://www.atilika.com/ja/products/kuromoji.html

KuromojiはJavaで書かれているオープンソースの日本語形態素解析エンジンです。

KuromojiはApache Software Foundationに寄付されており、バージョン3.6より JapaneseTokenizer としてApache LuceneとApache Solrの日本語サポートを提供していますが、単独で自然言語処理のプロジェクトにも 利用できます。

機能のまとめ

Kuromojiは下記の機能を含む日本語の形態素解析エンジンです：

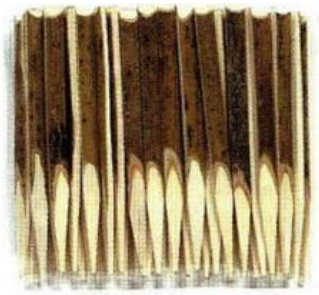
- **複合語の分割** テキストを言葉に分割（形態素）
- **品詞のタグ付け** 言葉分類の割当（名詞、動詞、助詞、形容詞など）
- **見出し化** 活用の動詞や形容詞に辞書の見出しを表示
- **読み方** 漢字の読み方を抽出

Kuromojiは下記の特徴があります。

- **実用的なパッケージング** 必要なものがすべて含まれるjarファイルとしてのパッケジング
- **検索用の設計** 検索リコールを改善するため複合プレーズを分割するモード
- **簡単に利用** 簡単な利用のため使いやすいAPIおよびMavenインテグレーション
- **実用的なライセンス** オープンソースも商業用ソフトウェアでも適用できるApache v2ライセンス

Amazon Elasticsearch Service

- Kuromoji
  - 英語: This is a pen.
    - This 主語 / is 動詞 / a 不定冠詞 / pen 名詞
  - 日本語: これはペンです。
    - これ 主語 / は 助詞 / ペン 名詞 / です 助動詞
  - 日本語はスペースで区切られていない
    - 日本語用の解析が必要
  - Kuromojiの由来
    - http://shinodogg.com/?p=3346



Amazon Elasticsearch Service

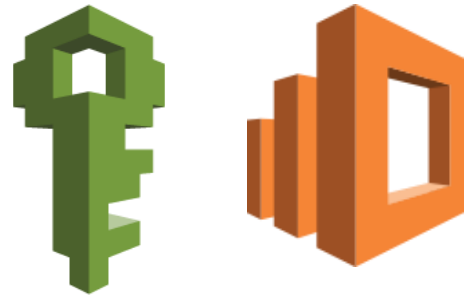
- Kuromojiのデモ
  - https://www.atilika.com/ja/products/kuromoji.html

```
import org.atilika.kuromoji.Token;
import org.atilika.kuromoji.Tokenizer;

public class TokenizerExample {
   public static void main(String[] args) {
      Tokenizer tokenizer = Tokenizer.builder().build();
      for (Token token : tokenizer.tokenize("寿司が食べたい。")) {
         System.out.println(
            token.getSurfaceForm() + "¥t" +
            token.getAllFeatures()
         );
      }
   }
}
```

```
$ java -Dfile.encoding=UTF-8 ¥
      -cp lib/kuromoji-0.7.7.jar:src/main/java KuromojiExample
寿司    名詞,一般,*,*,*,*,寿司,スシ,スシ
が      助詞,格助詞,一般,*,*,*,が,ガ,ガ
食べ    動詞,自立,*,*,一段,連用形,食べる,タベ,タベ
たい    助動詞,*,*,*,特殊・タイ,基本形,たい,タイ,タイ
。      記号,句点,*,*,*,*,。,。,。
```

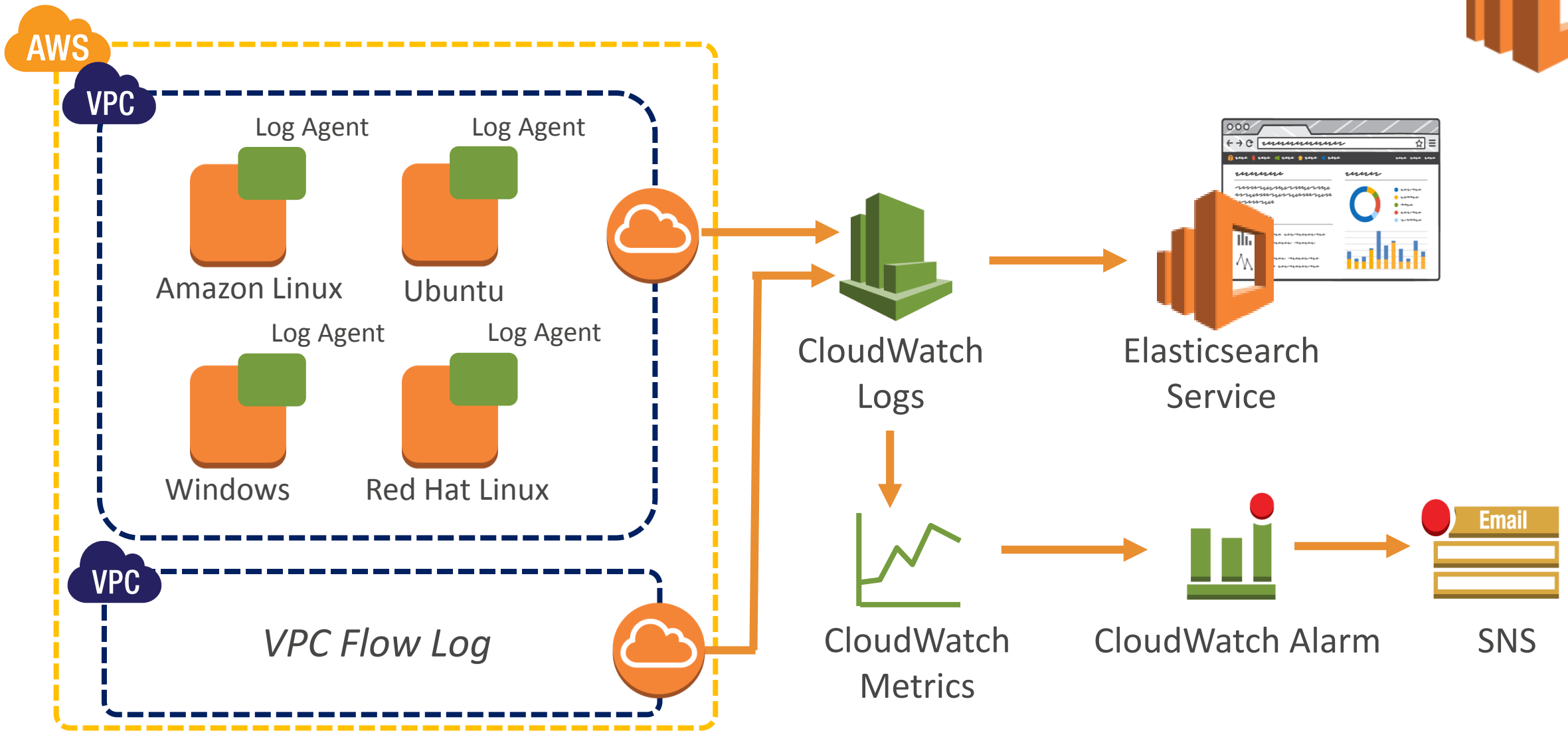
Amazon Elasticsearch Service



- IAM Integration
  - IPアドレスベースの制限
    - 例) Kibanaへのアクセスは社内のIPアドレスレンジからのみ
  - Signed requests with SigV4
    - 例) AWSのクレデンシャルを使ってセキュアにアクセス
    - Logstashプラグイン: https://github.com/awslabs/logstash-output-amazon_es

  - Fine-grainedアクセスコントロール
    - 例) ドメイン内のIndex毎にアクセス権限を分ける

CloudWatch Logs を使ったログ監視



Amazon Elasticsearch Service

- 2016年3月末まで無料のqwikLABS
  - https://qwiklabs.com/

Celebrating 10 years of AWS with unlimited free labs!

emily linde

March 14, 2016

Announcing...

10 Years of AWS, Free!, Limited time

Exciting news – to celebrate 10 years of AWS, *qwik*LABS is offering all of our AWS labs for free! Starting right now, you can take any lab (we mean any!) free of charge on *qwik*LABS.com. Then take another, and another, and another... as many as you want!

Try out that lab you've had your eye on for weeks, finish up the Quest you started last month, or earn your next badge, totally free of charge. Real, hands-on training and Amazon authorized content – take advantage today.

Introduction to Amazon Elasticsearch Service

Select

FREE

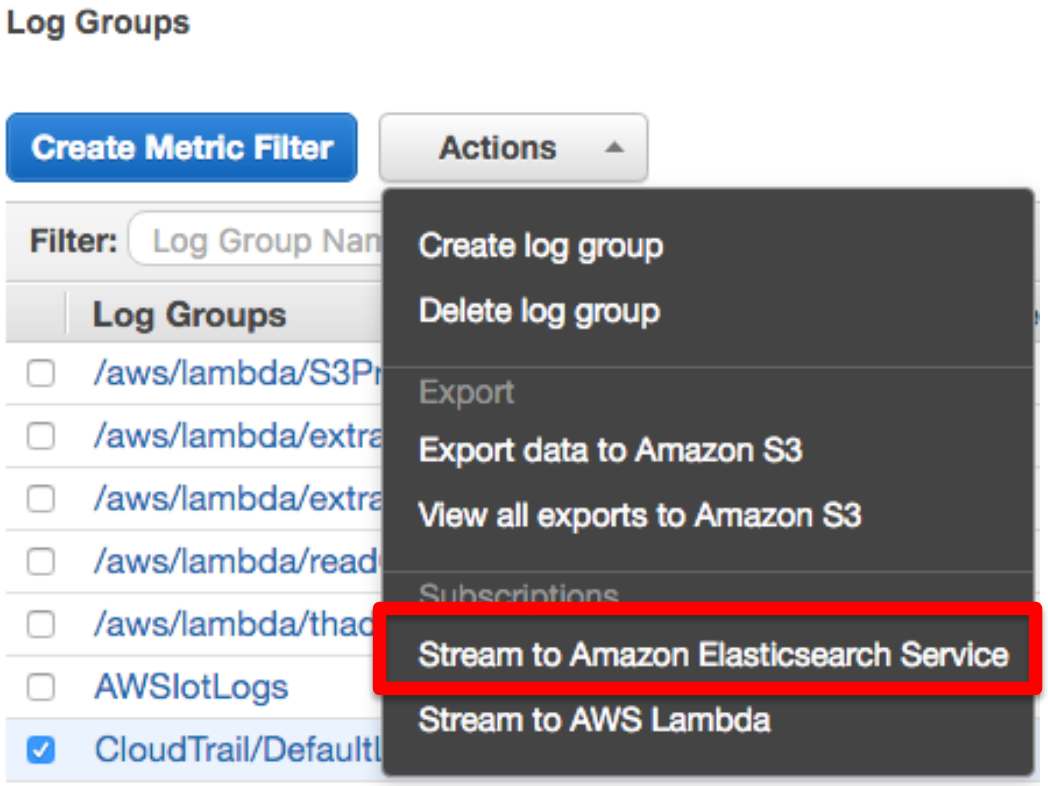
The lab will give you the basic understanding of Amazon Elasticsearch Service (ES). It will demonstrate the basic steps required to get started with Amazon ES: creating clusters, cluster node configurations, storage configurations and Identity & Access Management (IAM) Policies. As prerequisites you should have already taken the "Introduction to Amazon Elastic Compute Cloud (EC2)" and "Introduction to AWS Identity and Access Management (IAM)" labs. Previous knowledge of Kibana4 and Elasticsearch is desirable.

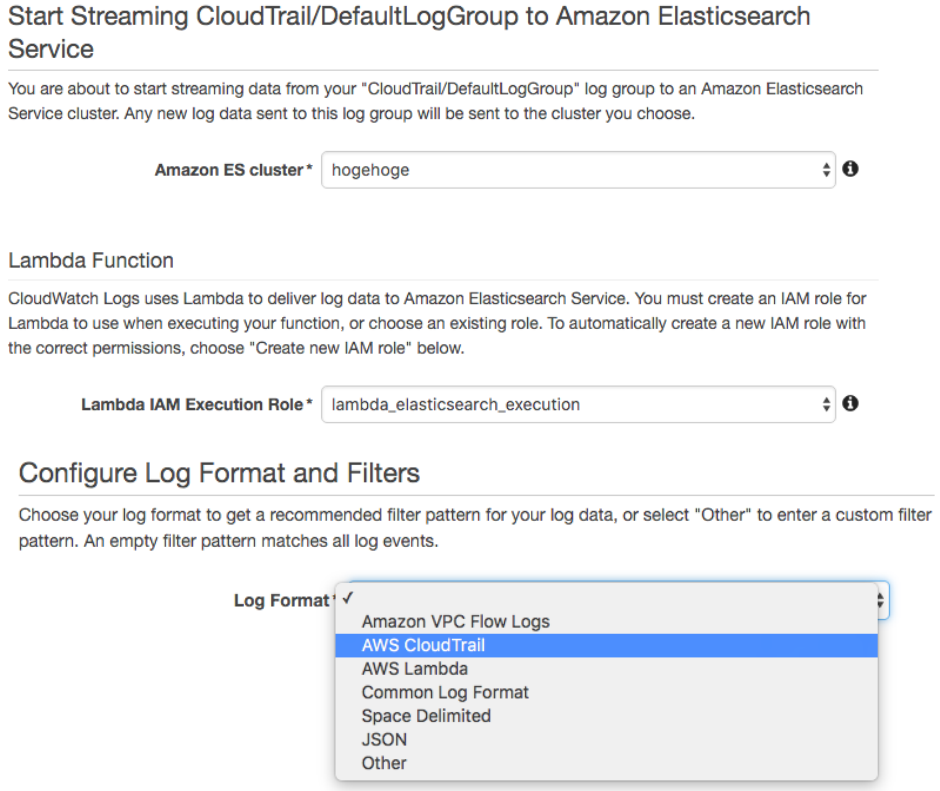
439

Duration:	00 h:30 m
Access Time:	00 h:40 m
Setup Time:	00 h:00 m
Levels:	Introductory
Tags:	Elasticsearch, free lab, Kibana, search engine, text, logfiles, query,

Amazon Elasticsearch Service

- CloudWatch Logs
  - Stream to Amazon Elasticsearch Service



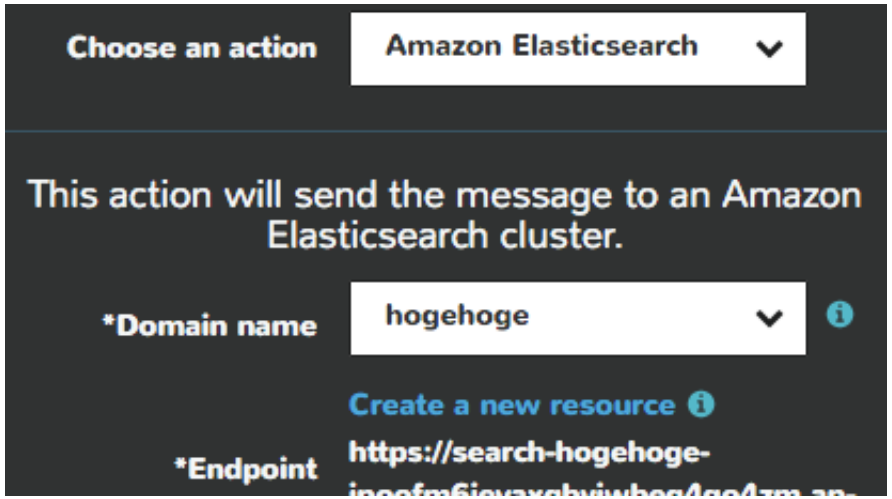


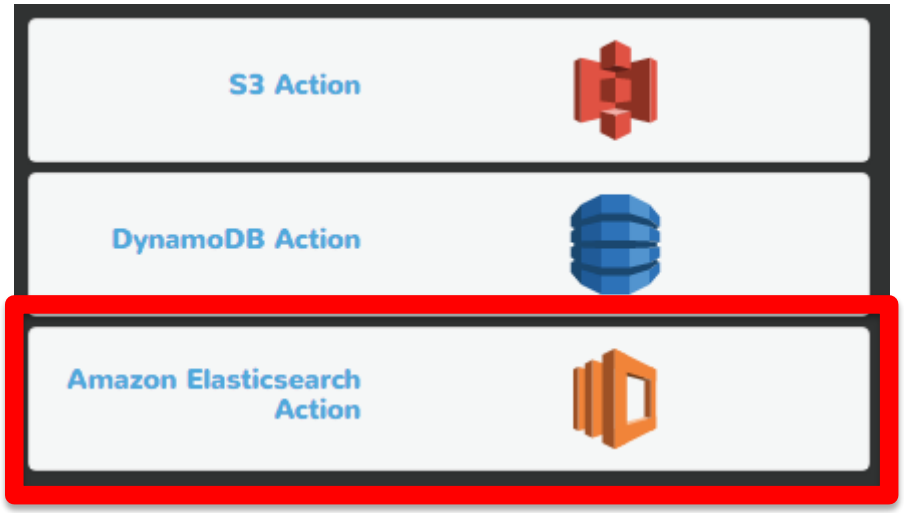
Amazon Elasticsearch Service

- AWS IoT との連携
  - http://aws.typepad.com/aws_japan/2016/03/aws-iot-update.html

【アップデート】 AWS IoT が Elasticsearch Service と CloudWatch に連携できるようになりました

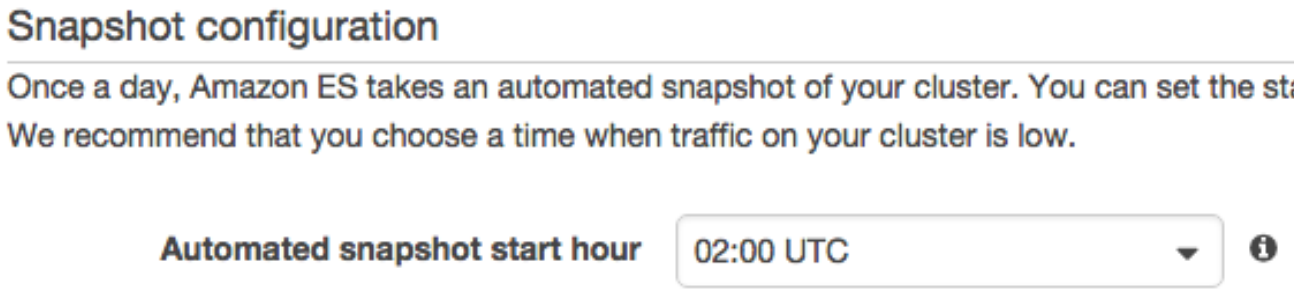
AWS IoT のルールでデバイスが生成したデータを直接、 Amazon Elasticsearch ドメインに渡すことができるようになりました。これによってデータを分析したり、データに対してフルテキストやパラメータによる検索を実行したり、 Kibana で可視化したりすることができます。この連携によって、デバイスの特定のエラーコードをフルテキスト検索したり、デバイスのパフォーマンスをリアルタイムに近い形で視覚化するような、ユースケースをサポートします。





Amazon Elasticsearch Service

- バックアップ＆リストア
  - AWSが自動で取得するもの
    - Daily: 1日1回 Automated snapshot start hourで指定
    - リストアはAWS技術サポートへお問い合わせ



  - お客さまがご自身で取得 ⇒ **_snapshot** API
    - Elasticsearchのフォーマットでお客さまのS3バケット
    - 任意のタイミングでいつでもリストア可能

Amazon Elasticsearch Service

- 今後も多くの機能追加を予定
  - お客さまからのご要望によって優先度は変わってきます
  - 是非AmazonESをご利用いただき、フィードバックいただければと思います！

Agenda

- 全文検索(Full-Text Search)
- 検索エンジンの基礎-Apache Lucene
- AWSの検索サービスのご紹介
  - Amazon CloudSearch
  - Amazon Elasticsearch Service
- **CloudSearchとAmazon ESの比較**

Amazon CloudSearch と Amazon ES の比較

	CloudSearch	Amazon ES
検索エンジン	Solr	Elasticsearch
検索エンジンのAPIを そのまま利用可能か？	X CloudSearchのAPIを利用	○ プラグイン利用等に制限あり
Auto Scaling	○ データ量/検索リクエスト量	X 構成を後から変更することは可能
大量のデータ保存	△ (低レイテンシ実現のため)インスタンスストアにデータを保持 ⇒ インスタンス数の増大に懸念	○ EBSを選択可能。1ノード最大512GBまで保持できる
ログの可視化	X CloudSearchに該当機能無し	○ Kibanaを使った可視化が可能
リアルタイムでのデータ取込	△ 最大5MBのファイルでの取込。1000回アップロード毎に0.1USD	○ RestAPI, Logstash, AWS Lambda等を使った取込

Amazon CloudSearch と Amazon ES の比較

- クラスメソッド木戸さんのブログ
  - 

  - http://dev.classmethod.jp/cloud/elasticsearch-service-vs-cloudsearch/
  - 決まった仕様の CloudSearch VS 高いカスタマイズ性の Elasticsearch
  - １対多のデータ構造対応なら迷わず Elasticsearch
  - 運用で楽がしたい場合は CloudSearch

Webinar資料の配置場所

宣伝

- AWS クラウドサービス活用資料集
  - http://aws.amazon.com/jp/aws-jp-introduction/

プロダクト別：				
Amazon S3		AWSマイスターシリーズ Re:Generate Amazon Simple Storage Service (S3)	Slideshare	PDF
Amazon Glacier		AWSマイスターシリーズ Reloaded Amazon Glacier Amazon Glacierのご紹介 機能編	Slideshare（Reloaded） Slideshare（機能編）	PDF（Reloaded） PDF（機能編）
Amazon Route 53		AWSマイスターシリーズ Re:Generate	Slideshare	PDF

公式Twitter/Facebook
AWSの最新情報をお届けします

宣伝

@awscloud_jp

もしくは
http://on.fb.me/1vR8yWm

最新技術情報、イベント情報、お役立ち情報、お得なキャンペーン情報などを
日々更新しています！