HUAWEI

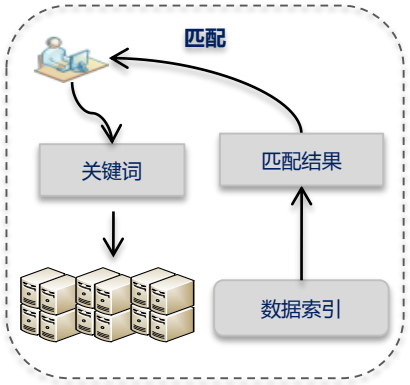
Day3 入门Elasticsearch原来如此简单

搜索的发展

由简单匹配到智能分析

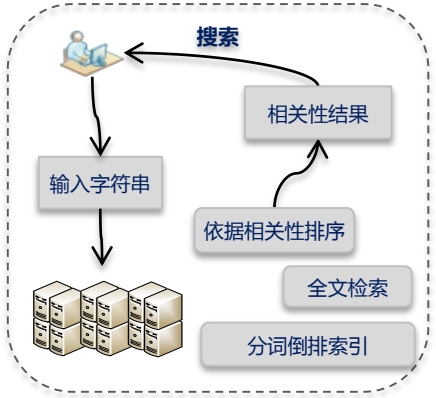
搜索技术的发展

第一代:
关键词匹配



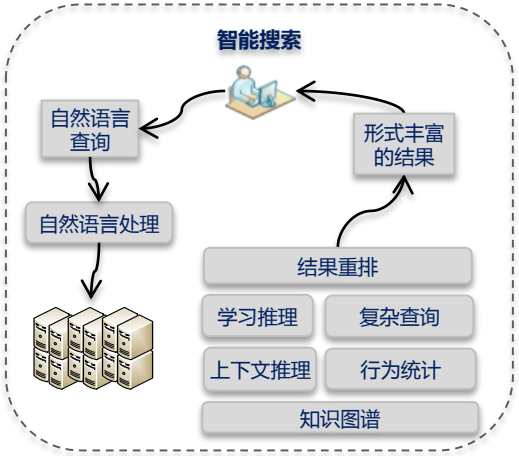
- 根据关键字段在数据中寻找匹配信息。
- 传统数据库式的匹配条件查询。

第二代:
相关性检索



- 基于分词的全文检索，通过相关性计算排序得到搜索结果。
- 可以根据用户的输入给予纠错和提示

第三代:
智能搜索



- 通过自然语言处理，学习推理，实现智能搜索。
- 用户仿佛在同一个知己对话，并得到文字、图片、音频、视频等各种相关结果。

业内实践

MySQL Egrep

elastic Solr

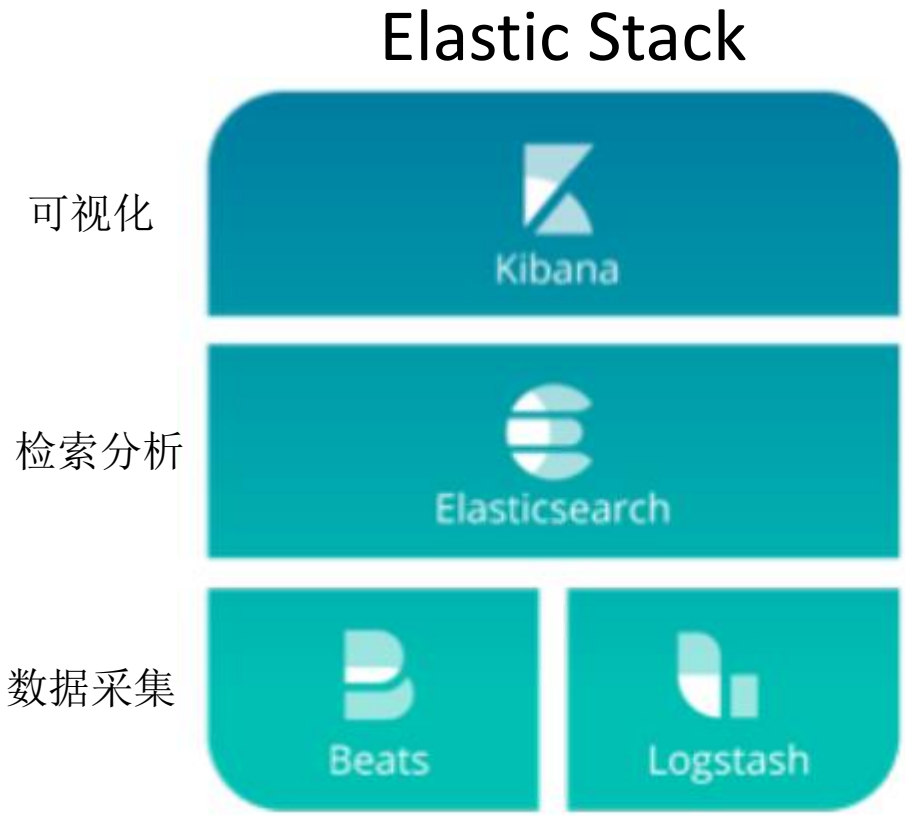
Google

Siri

HUAWEI

Elasticsearch是什么

- Elastic Stack是帮助用户实现结构化、非结构化数据从采集、检索分析到可视化的一个技术栈。
- Elasticsearch作为Elastic Stack的核心，是一个基于Lucene全文检索引擎的分布式搜索和分析软件。
- Elasticsearch采用RESTful接口，支持JSON格式。
- Elasticsearch每秒钟可处理海量事件，同时能够自动管理数据和请求在集群中的分布方式，以实现极其流畅的操作。
- Elasticsearch支持数字、文本、地理位置等多种数据格式。



云搜索服务是什么

云搜索服务（Cloud Search Service）是将**软件(Elasticsearch、Kibana)及硬件(计算、存储、网络)云化增强**后，为用户提供的**全托管搜索和分析平台**。

它可以帮助网站和APP**搭建搜索框**，提升用户寻找资料和视频的体验；还可以**搭建日志分析平台**，在运维上进行业务日志分析和监控，在运营上进行流量分析等等。

通过云搜索服务，可以快速搭建Elastic Stack，学习使用Elasticsearch相关技术。

基本概念

术语	描述
集群	云搜索服务是以集群为单位进行组织，一个集群代表一个独立运行的搜索服务，由多个节点构成。
节点	一个节点就是一个运行的Elasticsearch实例。
Index	用于存储Elasticsearch的数据，类似关系型数据库的Database。
Type	类似关系型数据库中的表，用于区分不同的数据，1个索引里面可以包含若干个文档类型。每个文档必须设定它的文档类型。
Document	Elasticsearch存储的实体，是可以被索引的基本单位，相当于关系型数据库中的行。
Mapping	用来约束字段的类型，可以根据数据自动创建。相当于数据库中的Schema。
Field	组成文档的最小单位。相当于数据库中的Column。
Shard	索引可以存储数据量超过1个节点硬件限制的数据。为满足这样的需求，Elasticsearch提供了一个能力，将一个索引拆分为多个，称为Shard。当您创建一个索引时，您可以根据实际情况指定Shard的数量。每个Shard托管在集群中的任一节点中，且每个Shard本身是一个独立的、全功能的“索引”。
Replica	Shard下的实际存储索引的一个副本。可以理解为备份Shard。副本的存在可以预防单节点故障。使用过程中，您可以根据业务情况增加或减少Replica数量。

Elasticsearch中数据的逻辑结构

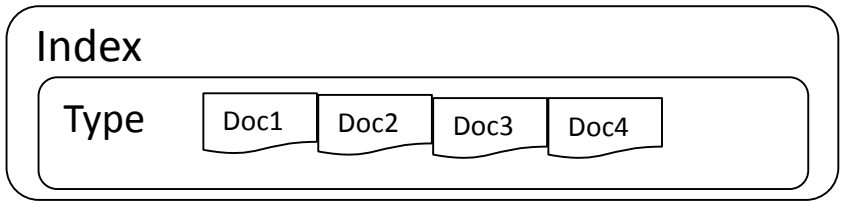
Document是Elasticsearch索引和搜索的原子单位，它是包含了一个或多个Field的容器，用JSON格式进行表示。每个Field拥有一个名字及一个或多个值。

Document由一个ID代表，被组织在Type中，Type又被组织在Index中。因此，Index + Type + DocID 可以唯一确定一个Document。

一个文档

```
{
    "name":         "张三",
    "age":          30,
    "birth_date":   "1990-01-01",
    "hobby":        ["爬山", "唱歌", "看书"]
}
```

一个索引

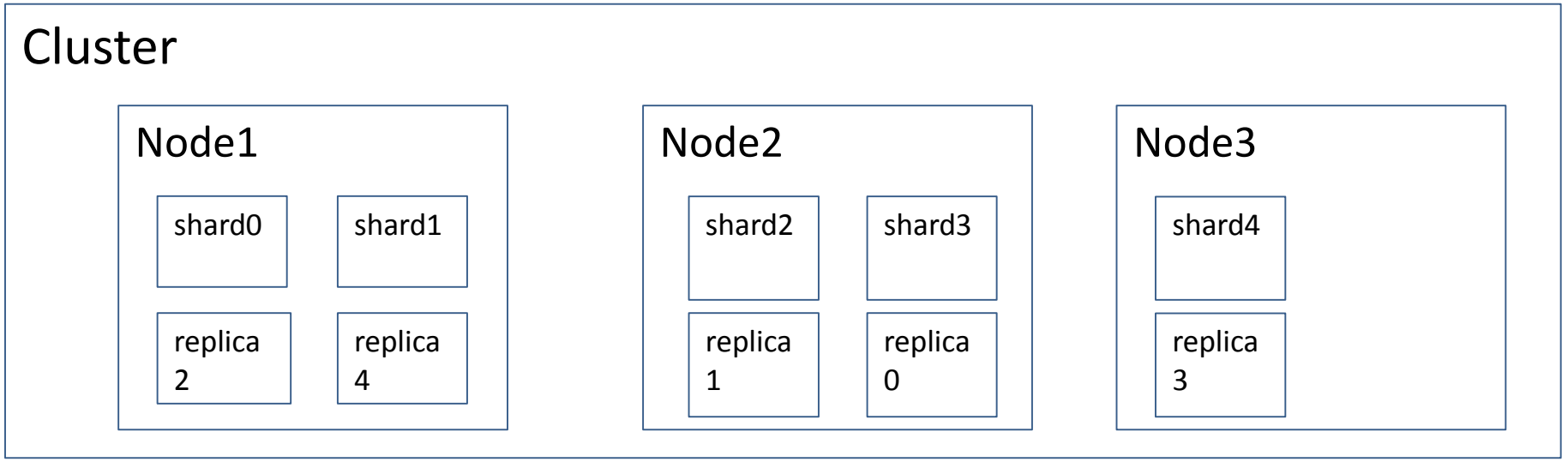


Elasticsearch中数据的物理结构

Elasticsearch支持分布式，因此，云搜索服务中的Elasticsearch物理形态是以集群的形式存在。集群由一个或者多个Node组成。

每个Node上，会分布不同数量的Shard。每个Shard中管理索引的一部分数据。

Shard分为主和副两种类型。习惯上把副Shard成为Replica。



Elasticsearch的查询语言DSL

Elasticsearch支持很多查询方式，最常用的就是DSL，也是采用JSON的格式描述。

```
GET _search
{
  "query": {
    "bool": {
      "must": [
        { "match": { "title":   "Search"        }},
        { "match": { "content": "Elasticsearch" }}
      ],
      "filter": [
        { "term":  { "status": "published" }},
        { "range": { "publish_date": { "gte": "2015-01-01" }}}
      ]
    }
  }
}
```

关键技术　分词

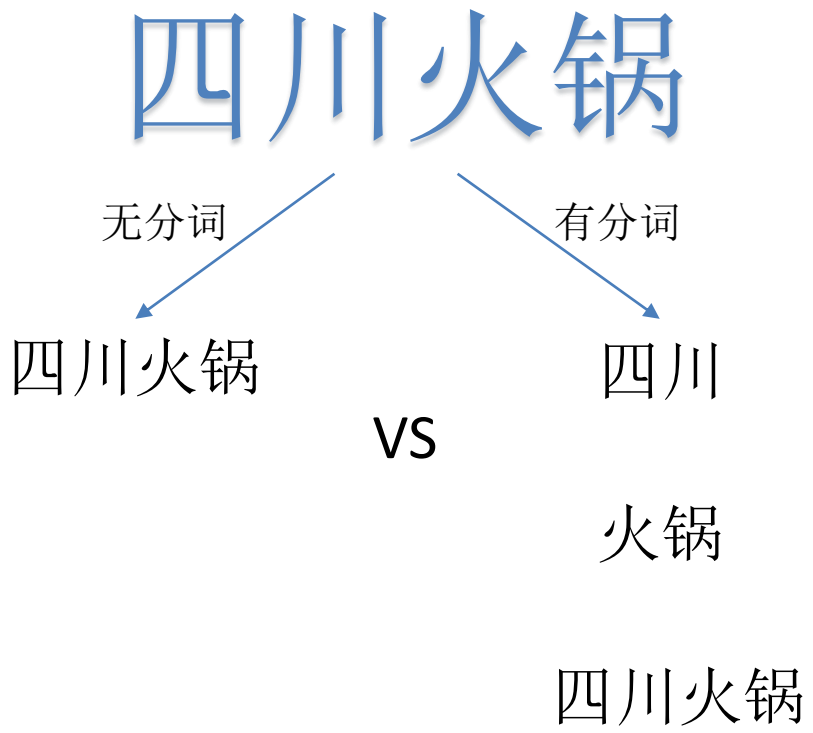
举例：
在做中文搜索时，小刘发现组合词检索在数据库是很难完成的。

例如，当用户在搜索框输入“四川火锅”时，数据库通常只能把这四个字去进行全部匹配。可是在文本中，可能会出现“推荐四川好吃的火锅”，这时候就没有结果了。

原因：
传统数据库不支持分词。“四川火锅”只能当做一个字符序列，在数据记录中进行匹配。

关键技术：
分词就是将连续的字序列按照一定的规范重新组合成词序列的过程。比如，“四川火锅”就可以分词得到“四川”、“火锅”、“四川火锅”三个词。

效果：
例如当小刘输入“四川火锅”时，Elasticsearch会自动做下面两件事
（1） 将“四川火锅”分词成“四川”、“火锅”和“四川火锅”
（2） 查找包含这三个词的文档
“推荐四川好吃的火锅”就可以被查找出来了！



关键技术　倒排索引

举例：
小刘在做测试时，发现当数据库中的文档数仅仅上百万条时，关键词查询就比较慢了。如果一旦到企业级的数据，响应速度就会更加不可接受。

原因：
在数据库做模糊查询时，如LIKE语句，它会遍历整个索引，同时进行字符串匹配。

例如，当小刘在数据库查询“手机”时，数据库会在每一条索引中去匹配“手机”这两字是否出现。实际上，并不是所有索引记录都包含“手机”。正排索引查询时不仅遍历了所有数据，还做了很多无用功。

这并不高效，而且随着数据量的增大，消耗的资源和时间都会线性的增长。

关键技术：
倒排索引是分词和文档之间建立起的映射关系。其将文本内容分词后产生的关键词作为KEY，文档标号列表作为VALUE。
通过倒排索引进行搜索，就是通过关键词查询对应的文档编号。

效果：
当小刘搜索“手机”时，从倒排索引就可以立即查到文档F，G，H。这样就不用花多余的时间在其他文档上了，因此检索速度得到了数量级的提升。上亿条的数据也是毫秒级得到结果。

正排索引

KEY	VALUE
A	手机市场
B	市场
C	用户市场
D	手机
E	用户
F	手机用户

VS

倒排索引

KEY	VALUE
手机	A, D, F
市场	A, B, C
用户	C, E, F

关键技术　相关性打分

举例：

在用数据库做搜索时，结果经常会出现一系列文档。小刘不禁思考：

- 到底什么文档是用户真正想要的呢？
- 怎么才能把用户想看的文档放在搜索列表最前面呢？

数据库只能依据某个字段排序，但是原始数据中并不存在用户的输入和文档匹配程度的字段

原因：

数据库并不支持相关性计算。查询的结果只有两个结果，命中或者不命中。

例如，当用户搜索“云搜索服务”的时候，“华为云搜索提供搜索服务”应该要比“百度云盘内容搜索服务”更相关，应该排在前面。

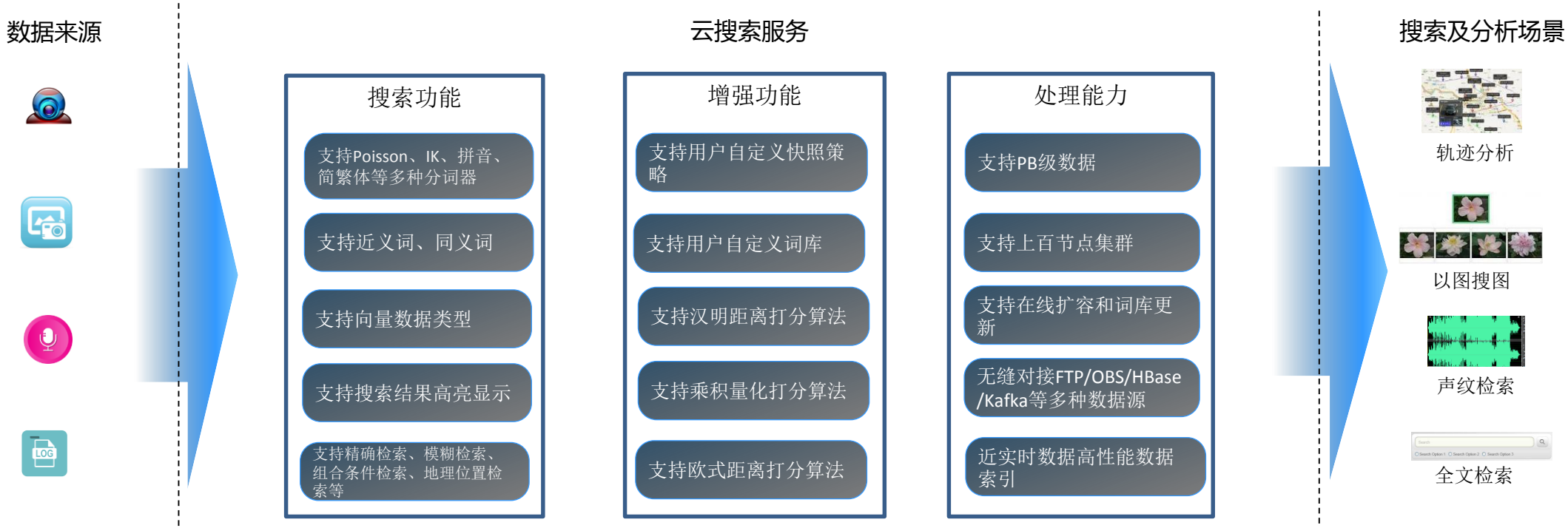
关键技术：

Elasticsearch支持相关度评分，针对匹配到的文档，根据一定算法，对匹配程度进行打分。这样在返回结果就会根据分数由高到低排列。分数越高，意味着和查询语句越相关。

效果：

小刘搜索“云搜索服务”的时候，“华为云云搜索提供搜索服务”排在了“百度云盘内容搜索服务”前面。

云搜索服务的功能及场景



- ✓ **100% 兼容**Elasticsearch APIs
- ✓ **PB级数据**处理，支持扩容至上百节点集群
- ✓ **无需编程**对接FTP/OBS/HBase/Kafka等多数据源
- ✓ **零业务中断**，支持在线扩容、词库热更新
- ✓ **增强算法**可支持图像、音频搜索

精彩预告

通过今天的学习，我们了解了Elasticsearch中的关键技术，简单了解了Elasticsearch的DSL语言。

明天，我们会手把手教大家搭建一个网站搜索框，帮助大家理解Elasticsearch在整个网站搜索流程中的位置，以及怎么实现同义词搜索。

Elasticsearch常见问题梳理

详情查看：https://support.huaweicloud.com/usermanual-es/es_01_0007.html

HUAWEI

Thank You.