地理形状

地理形状（ Geo-shapes ）使用一 与地理坐 点完全不同的方法。我 在 算机屏幕上看到的 形并不是由完美的 的 成的。而是用一个个 的着色像素点画出的一个近似 。地理形状的工作方式就与此相似。

的形状——比如点集、 、多 形、多多 形、中空多 形——都是通 geohash 元 ``画出来''的， 些形状会 化 一个被它所覆 到的 geohash 的集合。

NOTE 上， 型的 格可以被用于 geo-shapes： 使用我 之前 的 geohash ， 外 有一 是 四叉 。四叉 与 geohash 似，除了四叉 个 只有 4 个元，而不是 32 。 不同取决于 方式的 。

成一个形状的 geohash 都作 一个 元被索引在一起。有了 些信息，通 看是否有相同的 geohash 元，就可以很 易地 个形状是否有交集。

geo-shapes 有以下作用：判断 的形状与索引的形状的 系； 些 系 可能是以下之一：

`intersects`
: 的形状与索引的形状有重 （ ）。

`disjoint`
: 的形状与索引的形状完全 不 重 。

`within`
: 索引的形状完全被包含在 的形状中。

Geo-shapes 不能用于 算距 、排序、打分以及聚合。

映射地理形状

与 `geo_point` 型的字段相似， 地理形状也必 在使用前明 映射：

```
PUT /attractions
{
  "mappings": {
    "landmark": {
      "properties": {
        "name": {
          "type": "string"
        },
        "location": {
          "type": "geo_shape"
        }
      }
    }
  }
}
```

需要考 修改 个非常重要的 置： 精度 和 距 差 。

精度

精度 （ precision ）参数 用来控制生成的 geohash 的最大 度。 精度 9 ，等同于尺寸在 5m x 5m 的geohash 。 个精度可能比 需要的精 得多。

精度越低，需要索引的 元就越少， 索 也会更快。当然，精度越低，地理形状的准 性就越差。 需要考 自己的地理形状所需要的精度——即使 少1-2个等 的精度也能 来明 的消耗 收益。

可以使用距 来指定精度 —— 如 <code>50m</code> 或 <code>2km</code>—不 些距 最 也会 成 的[geohashes]等 。

距 差

当索引一个多 形 ，中 区域很容易用一个短 geohash 来表示。麻 的是 部分，些地方需要使用更精 的 geohashes 才能表示。

当 在索引一个小地 ， 会希望它的 界比 精 。 些 念碑一个 着一个可不好。当索引整个国家 ， 就不需要 高的精度了。 差个50米左右也不可能引 争。

距 差 指定地理形状可以接受的最大 率。它的 是 0.025 ， 即 2.5% 。也就是 ，大的地理形状（比如国家）相比小的地理形状（比如 念碑）来 ，容 更加模糊的 界。

0.025 是一个不 的初始 。不 如果我 容 更大的 率， 地理形状需要索引的 元就越少。

索引地理形状

地理形状通 GeoJSON 来表示， 是一 放的使用 JSON 的二 形状 方式。 个形状都包含了形状 型— <code>point</code>, <code>line</code>, <code>polygon</code>, <code>envelope</code> —和一个或多个 度点集合的数 。

CAUTION	在 GeoJSON 里， 度表示方式通常是 度 在前， 度 在后。

如，我 用一个多 形来索引阿姆斯特丹 姆广 ：

```
PUT /attractions/landmark/dam_square
{
    "name" : "Dam Square, Amsterdam",
    "location" : {
        "type" : "polygon", ①
        "coordinates" : [[ ②
          [ 4.89218, 52.37356 ],
          [ 4.89205, 52.37276 ],
          [ 4.89301, 52.37274 ],
          [ 4.89392, 52.37250 ],
          [ 4.89431, 52.37287 ],
          [ 4.89331, 52.37346 ],
          [ 4.89305, 52.37326 ],
          [ 4.89218, 52.37356 ]
        ]]
    }
}
```

① type 参数指明了　　度坐　集表示的形状　型。

② lon/lat 列表描述了多　形的形状。

上例中大量的方括号可能看起来　人困惑，不　　上 GeoJSON 的　法非常　　：

1. 用一个数　表示　　度 坐　点：

   ```
   [lon,lat]
   ```

2. 一　坐　点放到一个数　来表示一个多　形：

   ```
   [[lon,lat],[lon,lat], ... ]
   ```

3. 一个多　形（　　polygon　　）形状可以包含多个多　形；第一个表示多　形的外　廓，后　的多形表示第一个多　形内部的空洞：

   ```
   [
     [[lon,lat],[lon,lat], ... ],  # main polygon
     [[lon,lat],[lon,lat], ... ],  # hole in main polygon
     ...
   ]
   ```

参　{ref}/geo-shape.html[Geo-shape mapping documentation] 了解更多支持的形状。

地理形状

{ref}/query-dsl-geo-shape-query.html[geo_shape　　]不　常的地方在于，它允　我　使用形状来做

，而不　是坐　点。

个例子，当我　的用　　出阿姆斯特丹中央火　站　，我　可以用如下方式，　出方　　1km 内的所有地　：

```
GET /attractions/landmark/_search
{
  "query": {
    "geo_shape": {
      "location": { ①
        "shape": { ②
          "type":   "circle", ③
          "radius": "1km",
          "coordinates": [ ④
            4.89994,
            52.37815
          ]
        }
      }
    }
  }
}
```

① 使用 location 字段中的地理形状。

② 中的形状是由 shape 的内容表示。

③ 形状是一个半径　1km 的　形。

④ 安姆斯特丹中央火　站入口的坐　点。

的，　（或者　器　——　工作方式相同）会从已索引的形状中与指定形状有交集的部分。此外，可以把　relation　字段　置　disjoint　来与指定形状不相交的部分，或者　置　within 来　完全落在　形状中的。

个例子，我　可以　阿姆斯特丹中心区域所有的地　：

```
GET /attractions/landmark/_search
{
  "query": {
    "geo_shape": {
      "location": {
        "relation": "within", ①
        "shape": {
          "type": "polygon",
          "coordinates": [[ ②
              [4.88330,52.38617],
              [4.87463,52.37254],
              [4.87875,52.36369],
              [4.88939,52.35850],
              [4.89840,52.35755],
              [4.91909,52.36217],
              [4.92656,52.36594],
              [4.93368,52.36615],
              [4.93342,52.37275],
              [4.92690,52.37632],
              [4.88330,52.38617]
            ]]
        }
      }
    }
  }
}
```

① 只匹配完全落在　　形状中的已索引的形状。

②　个多　形表示安姆斯特丹中心区域。

在　　中使用已索引的形状

　于那些　常会在　　中使用的形状，可以把它　索引起来以便在　　中可以方便地直接引用名字。以之前的阿姆斯特丹中部　例，我　可以把它存　成一个　型　`neighborhood` 的文　。

首先，我　　照之前　置 `landmark`　的方式建立映射：

```
PUT /attractions/_mapping/neighborhood
{
  "properties": {
    "name": {
      "type": "string"
    },
    "location": {
      "type": "geo_shape"
    }
  }
}
```

然后我　索引阿姆斯特丹中部　　的形状：

```
PUT /attractions/neighborhood/central_amsterdam
{
  "name" : "Central Amsterdam",
  "location" : {
      "type" : "polygon",
      "coordinates" : [[
        [4.88330,52.38617],
        [4.87463,52.37254],
        [4.87875,52.36369],
        [4.88939,52.35850],
        [4.89840,52.35755],
        [4.91909,52.36217],
        [4.92656,52.36594],
        [4.93368,52.36615],
        [4.93342,52.37275],
        [4.92690,52.37632],
        [4.88330,52.38617]
      ]]
  }
}
```

形状索引好之后，我　就可以在　　中通　index，type 和 id 来引用它了：

```
GET /attractions/landmark/_search
{
  "query": {
    "geo_shape": {
      "location": {
        "relation": "within",
        "indexed_shape": { ①
          "index": "attractions",
          "type":  "neighborhood",
          "id":    "central_amsterdam",
          "path":  "location"
        }
      }
    }
  }
}
```

① 指定　indexed_shape　而不是　shape　，Elasticesearch　就知道需要从指定的文　和　path　　索出　　的形状了。

阿姆斯特丹中部　个形状没有什　特　的。同　地，我　也可以在　　中使用已　索引好的　姆广　。　个　　可以　出与　姆广　有交集的　近点：

```
GET /attractions/neighborhood/_search
{
  "query": {
    "geo_shape": {
      "location": {
        "indexed_shape": {
          "index": "attractions",
          "type":  "landmark",
          "id":    "dam_square",
          "path":  "location"
        }
      }
    }
  }
}
```